Une approche cognitive du langage mathématique afin d'assister les activités de manipulation de formules

Gérald Moulis

N° d'ordre : 71

Année 1991

THÈSE

présentée en vue de
l'obtention du titre de

DOCTEUR

de

L'ECOLE NATIONALE SUPERIEURE
DE L'AERONAUTIQUE ET DE L'ESPACE

SPECIALITE : INTELLIGENCE ARTIFICIELLE

par

Gérald MOULIS

UNE APPROCHE COGNITIVE DU LANGAGE MATHEMATIQUE AFIN D'ASSISTER LES ACTIVITES DE MANIPULATION DE FORMULES

Soutenue le 28 juin 1991 devant la Commission d'Examen :

MM.	R.	CUPPENS	Président
	L.	GOUZENES	Examinateurs
	D.	LACOMBE	
Mme.	D.	PASTRE	
M.	M.	SAMUELIDES	

"Thèse préparée au sein du Département d'études et de recherches en Informatique (DERI) du CERT / ONERA"

REMERCIEMENTS

Je remercie Madame et Messieurs les membres du jury de leur intérêt scientifique. En particulier, je remercie chaleureusement :

Monsieur Roger Cuppens, professeur de l'Université Paul Sabatier, de m'avoir fait l'honneur de présider ce jury. Il s'est dépensé pour étudier cette thèse et m'aider dans mes recherches ; son enthousiasme et sa rigueur sont une source de motivation exemplaire,

Monsieur Manuel Samuelides, professeur de l'ENSAE, de s'être engagé par intérêt scientifique sur un sujet déjà défini afin d'y prodiguer ses conseils avisés. Son sens de l'engagement lui a permis de planifier et de sanctionner l'avancement de l'étude même dans les passages difficiles,

Monsieur Laurent Gouzènes, chercheur puis consultant industriel, d'être l'instigateur de ce sujet d'un intérêt scientifique et épistémologique majeur. Ses techniques d'analyse et son action de structuration finale ont contribué à achever cette étude du Langage Mathématique,

Monsieur Daniel Lacombe, professeur de l'Université Paris VII et pourfendeur de l'illogisme et de l'antilogicisme de ses contemporains. Son style et son brio oratoire sont en accord avec la qualité de sa vivante contribution à l'étude du Langage Mathématique,

Madame Dominique Pastre, maître de conférences du LAFORIA, pour sa modestie, son ouverture et sa rigueur scientifique. L'activité mathématique ne serait pas si difficile si Dominique Pastre n'en avait pas exploré avec talent et persévérance les facettes de l'heuristique et du raisonnement. Son enthousiasme et son soutien m'ont particulièrement aidé,

Monsieur Laurent Chaudron, chercheur au CERT, pour l'originalité de sa pensée et le pouvoir de conviction de ses justifications. Son esprit de synthèse lui a permis de progresser là où d'autres rompent l'engagement en attendant de s'y retrouver en masse. Qu'il trouve ici, lui et sa famille, l'expression de mon amitié ainsi que le témoignage de son soutien permanent,

Cette thèse n'aurait pas vu le jour sans le financement répété de la DRET (contrats EAR n° 86/1428/DRET/DS/SR et 88/1525/DRET/DS/DR). Elle n'aurait pas eu lieu sans l'engagement renouvelé du responsable du GIA, Monsieur François-Régis Valette, son accueil originel comme responsable du DERI, et sans l'action du responsable du DERI Monsieur René Jacquart. Elle aurait manqué d'exemples sans l'amabilité des auteurs des ouvrages scientifiques référencés et de leurs éditeurs. Citons aussi le remarquable travail de Monsieur Laurent Fallot et la gracieuse exploitation de MAD pour servir de repère dans une problématique différente,

Des chapitres de cette thèse ont de plus bénéficié de la lecture, des conseils et des encouragements de Guy Boy, Jean-Michel Hufflen, Michel Lemoine et Véronique Royer. J'ai de plus bénéficié de l'accueil de Bernard Senach, Norber Kajler et l'équipe Sysiphe de l'INRIA-Sophia. J'ai enfin bénéficié de l'apport documentaire de Anne Condamine, Frédérique Molines et Jacques Virbel (IRIT) ; des publications et contributions des linguistes-mathématiciens Colette Laborde et François Lo Jacomo, et de l'admirable bibliothèque du MLAD et de L'IREM de Toulouse,

Je tiens à remercier personnellement mes collègues ou amis qui m'ont encouragé, en particulier mais non exclusivement Nora Boulaya, son mari toujours disponible et efficace, Kioumars Yazdanian et son réseau de Macs, Sandrine Moulin, Paul Depasse, Gérard Verfaillie, Nathalie Mathé, l'équipe du GIA, celle des stagiaires de thèse, secrétaires et ingénieurs du DERI, et l'édition du CERT qui effectue avec efficacité l'essentiel du tirage des thèses.

A tous ceux qui me sont chers, à Marie-Thérèse et Céline :

"Dieu bénit l'homme

Non pour avoir trouvé, mais pour avoir cherché."

Victor Hugo : Les Contemplations

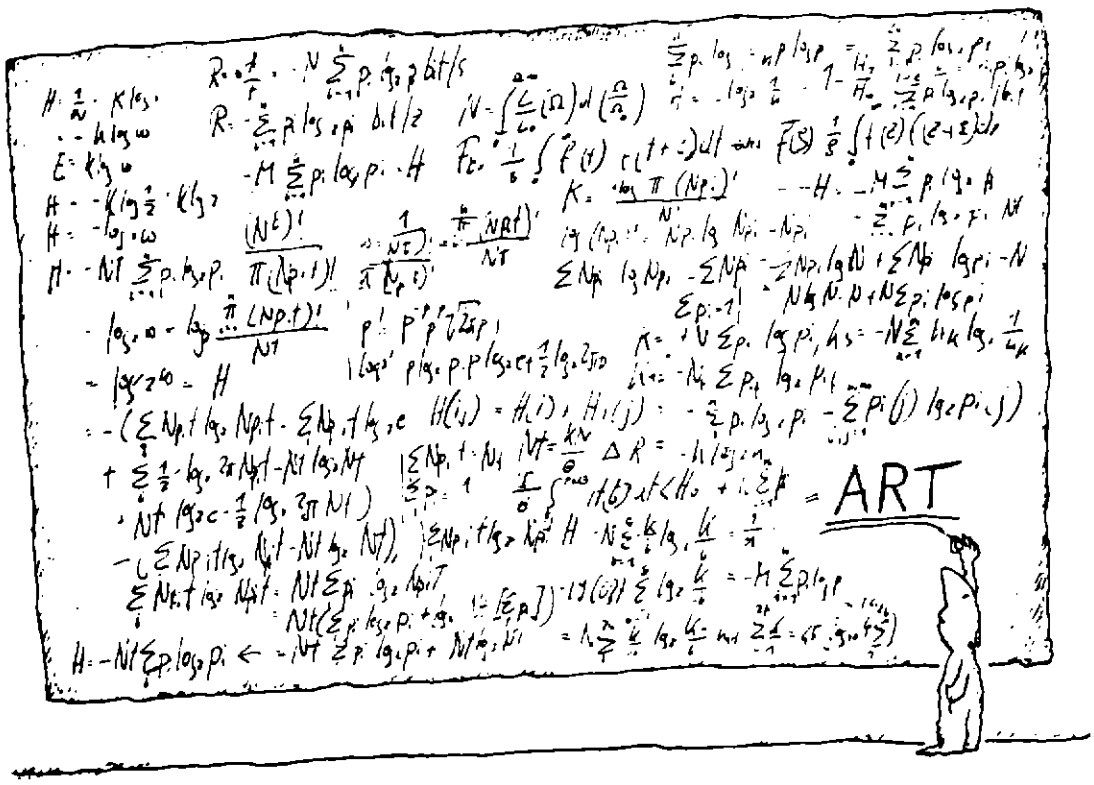


Illustration : "Définition" de Magi Wechsler,
dans : "Le langage du changement ; éléments de communication thérapeutique" de Paul Watzlawick

PLAN

PARTIE I : LE LANGAGE MATHÉMATIQUE

1. Le travail du mathématicien
2. Introduction à la modélisation du langage
3. Regards sur l'usage du Langage Mathématique
4. Analyse linguistique d'écrits mathématiques
5. L'usage des preuves mathématiques

PARTIE II : LES MATHÉMATIQUES ASSISTÉES

1. L'assistance au mathématicien
2. Étude et modélisation de preuves
3. Assistance lors de la construction de preuves
4. Fonctionnalités d'un système d'assistance
5. Problématique de l'assistance au travail

PARTIE III : LES TECHNIQUES INFORMATIQUES

1. État des lieux des systèmes informatiques
2. Les manipulations de formules
3. Modèles informatiques pour une mise en œuvre ergonomique
4. Le traitement du Langage Mathématique
5. Organisation des connaissances
6. Opérationnalisation des formules

CONCLUSION

1. Une étude de l'activité mathématique
2. Présentation chronologique
3. Notre contribution vers un AMI
4. Défis et perspectives des MAO
5. Bilan

INTRODUCTION

ORIENTATIONS, EXPLORATIONS POUR DEMAIN

Le développement des outils mathématiques joue un rôle essentiel dans de nombreux domaines et en particulier dans toutes les branches de l'informatique de formalisation et de compréhension : en modélisation et représentation de la connaissance et du raisonnement, en automatique, dans les problèmes d'éléments finis, de la complexité algorithmique etc... Ce développement est capital et la DRET apportera son soutien en particulier, mais non exclusivement, pour encourager les transferts vers les techniques informatiques.

[La Recherche 90]

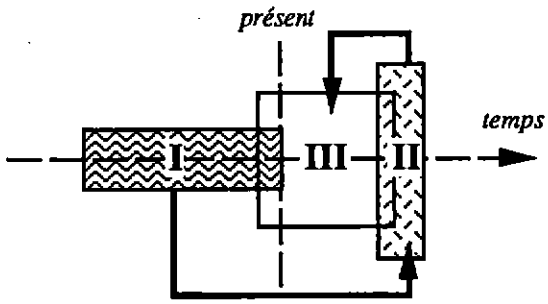
L'organisation globale de "l'entreprise des mathématiques" se subdivise en plus de 3000 secteurs ! Ces secteurs d'activité ne se mesurent pourtant pas via leurs chiffres d'affaires mais par leurs professionnels attitrés (30 000 aux USA) ou par leur **production de 200 000 théorèmes par an**[1] ! Les mathématiques ne sont pas pour autant réservées à une élite, puisque chaque personne est susceptible de produire des modèles mathématiques, de manipuler des formules, de résoudre des équations ou d'utiliser l'une des multiples fonctions standard.

Malgré le gigantisme des proportions, les mathématiques veulent regrouper l'ensemble des informations produites en un savoir universellement disponible. La bibliothèque de la Brown University compte par exemple 60 000 volumes sans prétendre à l'exhaustivité, ni à la validité du contenu... Le stockage, l'organisation, le recensement ou la vérification des connaissances relève ainsi de **procédés artisanaux** employés depuis des millénaires. Bien que les technologies actuelles ont développé les moyens de diffusion et de validation communautaire des connaissances, le **problème épistémologique** demeure : représenter et effectuer un contrôle objectif d'un savoir si bien défini et cohérent.

La progression rapide du traitement de la connaissance offre des perspectives pour y remédier. Encore faut-il imaginer et proposer des solutions pour définir où et comment intervient l'apport de l'informatique. Nous avons donc développé notre étude à partir de l'interrogation :

Peut-on fournir au mathématicien des outils informatiques adaptés à son activité ?

Nous qualifions ce domaine d'application de Mathématiques Assistées par Ordinateur ou MAO. Cette interrogation est le fondement de la constitution d'un domaine pour lequel peu de réflexions sont disponibles et dont l'état actuel et futur reste à définir. Cette interrogation se décompose en trois sous-problèmes, dont la répartition temporelle est schématisée par le **cycle de développement** ci-dessous :



I) Quelle est la nature de l'activité mathématique ?

II) Que peut-on souhaiter ?

III) Que peut-on proposer ?

figure 0.1

Pour résumer notre réponse, nous dirons que ce qui caractérise l'activité observable du mathématicien est la production de démonstrations. Les outils adaptés à son activité sont donc des **outils d'assistance au cycle de vie des démonstrations** : définition du problème, construction et vérification de la preuve, présentation écrite et diffusion. Les capacités de raisonnement servent alors à réaliser des tâches élémentaires plutôt que la résolution d'un problème général.

[1] Ces nombres de 1980 sont tirés de "l'Univers Mathématique", un livre de Philip J. Davis et Reuben Hersh qui explore avec originalité et profondeur la diversité de l'expérience mathématique [Davis 85].

La solution que nous proposons est donc centrée sur l'**assistance**. La démarche globale adoptée dans notre étude correspond à la décomposition précédente et structure notre thèse en trois parties :

I) Quelle est la nature de l'activité mathématique ?
- analyse de l'activité mathématique ;
- étude du Langage Mathématique.

II) Que peut-on souhaiter ?
- construction de preuves (opérations mathématique) et démonstrations (forme textuelle) ;
- approche d'assistance privilégiant l'interaction.

III) Que peut-on proposer ?
- nos solutions pour un environnement ergonomique ;
- nos propositions pour les appliquer aux manipulations de formules.

L'activité mathématique, c'est-à-dire l'activité du mathématicien se modélise d'abord par son outil de mise en œuvre : le Langage Mathématique. Notre étude de l'activité mathématique cible le Langage Mathématique selon deux axes : scientifique et informatique (figure 0.2).

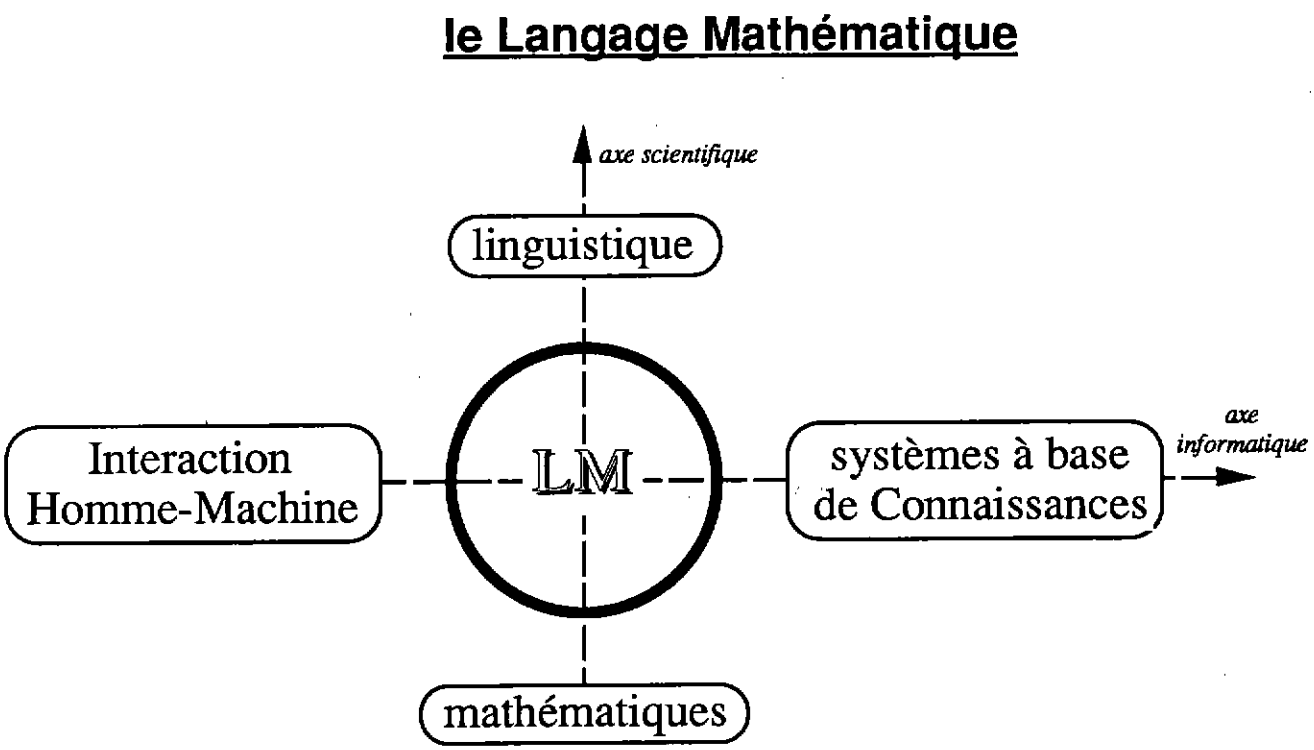


figure 0.2

La partie I étudie l'axe scientifique qui relie linguistique et mathématiques. Elle étudie les paramètres de l'activité mathématique pertinents pour la réalisation d'une assistance, indépendamment de toute considération informatique.

L'axe informatique du langage débute en partie II : il privilégie l'assistance par rapport à l'automatisation, grâce à l'Interaction Homme-Machine (IHM). Une fois les souhaits futurs précisés, la partie III propose des éléments d'un système de manipulation et de dialogue utilisant des connaissances mathématiques et langagières. La partie I est donc utilisée en partie II, et elles servent toutes deux à supporter la partie III.

Au delà des mathématiques, toute **discipline scientifique** est concernée par cette problématique où il n'est pas seulement question d'IHM mais de l'Interaction d'un Scientifique avec un Système Scientifique. D'ailleurs, si la matière de notre travail est constituée d'écrits de mathématiciens, les textes des biologistes, chimistes, informaticiens, etc. offrent un contenu similaire de "langage scientifique naturel".

Le thème clé de cette étude est ainsi **l'assistance à une activité mathématique transposée sur ordinateur**. Mais pour transposer une activité, il faut la connaître finement. Or très peu de travaux existent en la matière. Si l'on sait offrir des outils mathématiques, on ne sait guère ce qu'est l'activité du mathématicien, on ne connait pas les mécanismes de construction d'une preuve ni même les lois qui régissent les démonstrations.

L'assistance repose alors sur une étude détaillée de l'usage des mathématiciens. Une bonne part de notre étude consiste alors à préciser la nature et le rôle du langage dans les mathématiques. Cela permet de formuler et proposer des solutions informatiques différentes et originales.

Cette étude s'intéresse prioritairement à la pratique écrite d'un mathématicien qui maîtrise son activité, et cherche à assister ses productions mathématiques. L'axe scientifique analyse l'activité mathématique telle que la pratique les mathématiciens, sans se polariser sur une activité de raisonnement trop impalpable malgré des études nombreuses. Elle reste cantonnée aux mathématiques enseignées dans nos préparations scientifiques de premier cycle.

D'un point de vue scientifique, le Langage Mathématique présente trois **caractéristiques** fondamentales qui le distingue des langages opératifs qui fleurissent par besoin dans des domaines d'expertise restreints :

- la présentation et les concepts du langage résultent d'une maturation millénaire ;
- l'usage du langage reflète la conduite de l'activité cognitive du mathématicien ;
- les productions mathématiques sont "universelles", cohérentes et vérifiables.

L'un des acquis majeurs de l'analyse de l'activité mathématique est que les connaissances usuellement modélisées restent superficielles et se cantonnent à des situations stéréotypées. Les solutions proposées se limitent alors à quelques domaines restreints, sans s'intéresser directement aux problèmes généraux posés par l'usage des formules, du langage et la modélisation d'un domaine.

Vu ces lacunes, rien d'étonnant que les systèmes conçus choisissent une mécanisation simplificatrice au détriment du naturel et de l'explicabilité. C'est pourquoi cette étude ne se polarise pas sur une activité de résolution de problèmes et n'a aucune ambition didactique directe. En mettant l'accent sur l'expression des tâches et les productions langagières, elle reconsidère l'exploitation des outils mathématiques existants.

Nous proposons ***une approche cognitive*** pour :

- observer et modéliser l'activité mathématique telle qu'elle est pratiquée par les mathématiciens.
- répertorier et étudier les éléments de communication et de mise en œuvre de l'activité du mathématicien : le ***Langage Mathématique***.
- définir les structures et les connaissances que les mathématiciens utilisent dans leur activité.

L'approche cognitive impose donc de procéder par étapes à partir d'une analyse de l'activité, en oubliant les contraintes de faisabilité immédiate. Une caractérisation possible de l'immaturité des MAO est que chaque étape pose un problème dont une résolution détaillée est hors de portée de cette étude. Nous ne pouvons qu'indiquer quelques jalons indispensables aux besoins descriptifs et convictifs de cette étude. Deux exemples immédiats permettent d'appréhender les problèmes de modélisation des mathématiques :

- Prendre les premières pages d'un traité de mathématiques et chercher à organiser les définitions, notations, théorèmes et propriétés qu'il propose.
- Choisir un traité d'exercices et chercher à représenter le texte d'une démonstration, les déductions qu'elle exprime et les opérations nécessaires à leur réalisation.

Cette approche étudie les énoncés produits, les concepts avec lesquels le mathématicien bâtit les énoncés, ainsi que les procédures et les paramètres d'usage de ces énoncés. Elle cherche à utiliser des composantes du langage pour dialoguer avec un système, pour construire des productions, pour représenter des connaissances et pour les exploiter. Afin d'orienter notre étude vers un but concret d'intérêt général, nous avons choisi pour l'axe informatique de :

Concevoir un environnement ergonomique de manipulation de formules

Notre étude du Langage Mathématique est ainsi dirigée par une finalité informatique :

- Nous cherchons une ***assistance*** au travail scientifique du mathématicien.
- Nous nous intéressons particulièrement aux ***activités de manipulation de formules*** car elles forment une part essentielle de l'activité mathématique.

Il s'agit de transposer l'activité mathématique papier-crayon en une activité similaire effectuée sur un support informatique. Cela signifie qu'un tel système est conçu en fonction de l'usager, dans une recherche d'interaction homme-machine. L'usager contrôle donc la conception, et le système concrétise ses manipulations et s'efforce de lui apporter une aide pertinente. Un problème majeur est donc celui de la représentation et de l'exploitation algorithmique des abstractions proposées au mathématicien.

La modélisation des activités de manipulation de formules établit les bases du langage : une formule utilise des concepts mathématiques qu'il faut classifier, organiser et désigner ; elle les combine en une structure qu'il faut spécifier, construire et visualiser. De plus, les manipulations de formules utilisent des connaissances et sont elles-même exploitées dans des structures plus vastes comme la construction d'une preuve ou la résolution d'un problème.

Les systèmes de manipulation de formules classiques n'ont pas les moyens de reproduire l'activité mathématique naturelle. Ils peuvent simplement mimer quelques aspects indispensables de la tâche à laquelle ils sont destinés. Notamment, ils n'ont pas :

- les connaissances mathématiques et langagières pour le faire ;
- la possibilité d'utiliser des libertés de notations, de présentation ou de guidage ;
- une architecture informatique apte à utiliser des connaissances ;

Pour répondre à ces aspirations, nous proposons un **Atelier Mathématique Intégré** (AMI) dont nous définissons les principes, des fonctionnalités et l'architecture. Nous présentons plus particulièrement les fonctionnalités d'un module, baptisé SOFTMATH, destiné aux manipulations de formules.

Nous avons ainsi répondu au problème épistémologique initial, relatif aux procédés artisanaux utilisés en mathématiques. La construction mathématique (et non simplement textuelle) de démonstrations sur un support mathématique constitue le fer de lance d'outils de Mathématiques Assistées par Ordinateur. La problématique développée ici pour les mathématiques s'applique d'ailleurs pour l'Assistance au Travail Scientifique grâce à un "Système Scientifique".

Cette étude est qualitative et ne comprend PAS DE REALISATION INFORMATIQUE MAJEURE. Son intérêt réside donc uniquement dans son approche, dans les problèmes abordés et dans leur traitement. Nous pouvons ainsi présenter une **synthèse originale** des enjeux et problèmes techniques soulevés par l'assistance au mathématicien. Afin d'alléger la lecture, nous employons le "présent de l'indicatif" pour décrire les actions du système AMI lors des spécifications illustrant son utilisation.

Le contenu détaillé des trois parties correspond à la figure 0.1 (page xi) et suit la démarche globale adoptée. Conformément à ce découpage, la partie III repose sur une étude approfondie des techniques informatiques utilisées dans les systèmes actuels. Elle exploite d'ailleurs l'expérience de nos deux réalisations informatiques (génération et analyse). Notre objectif est que la partie III, qui résulte des besoins dégagés partie I et II, analyse les moyens informatiques de les satisfaire.

Partie I : Le Langage Mathématique.

Nous définissons les composantes du **travail du mathématicien** utiles pour modéliser son activité (ch. 1). Puis, nous étudions les possibilités actuelles de **modélisation du langage** et présentons les limites d'une formalisation logique (ch. 2). Cet état des lieux préliminaire nous révèle que le rôle et l'utilisation du Langage Mathématique restent à déterminer.

Nous explorons alors l'**usage du langage** par le mathématicien, en le comparant notamment à l'usage du langage tel qu'il peut être pratiqué pour un système logique (ch. 3). Les approches et les objectifs étant ainsi distingués, nous nous démarquons par une étude de l'activité du mathématicien.

L'**analyse linguistique d'écrits mathématiques** nous permet d'analyser en quoi l'usage du langage reflète la conduite de l'activité cognitive du mathématicien (ch. 4). L'étude approfondie de quelques textes nous permet d'établir l'urgence du développement d'une linguistique des mathématiques.

Cette analyse du langage débouche en particulier sur une modélisation susceptible de capturer l'**usage des preuves mathématiques** (ch. 5). Cet objectif est essentiel pour proposer un support informatique capable de rendre compte de l'activité du mathématicien.

Partie II : Les Mathématiques Assistées

L'étude précédente a permis de dégager un besoin considéré comme essentiel, tant pour le mathématicien que pour les Sciences Cognitives[2] impliquées dans ces recherches : la construction de preuves et démonstrations. La seule alternative réaliste est alors une **assistance au mathématicien** (ch. 1).

Nous définissons alors, sur des exemples mathématiques, un cadre plus restreint pour **l'étude et la modélisation de preuves** (ch. 2). Dans ce cadre, nous étudions les possibilités informatiques d'**assistance lors de la construction de preuves** (ch. 3). Nous présentons enfin les **fonctionnalités d'un système d'assistance** valables dans un cadre plus général. Nos propositions sont validées par des exemples réels de démonstrations mathématiques.

Ces souhaits d'un système ergonomique pour le mathématicien nécessitent le développement d'une **problématique de l'assistance au travail** centrée sur l'interaction avec un environnement de travail naturel. Cette problématique est aussi applicable pour d'autres domaines scientifiques (ch. 4).

Partie III : Les techniques informatiques

Forts de notre étude de l'activité du mathématicien et de nos besoins prospectifs, nous présentons l'**état des lieux des systèmes informatiques** actuels (ch. 1). Les Mathématiques effectivement Assistées par Ordinateur sont un objectif à atteindre, et certains thèmes fédérateurs doivent être étudiés.

L'un des thèmes concerne les **manipulations de formules** intégrant les aspects mathématiques et textuels (ch. 2). Un observateur extérieur pourrait résumer la production du mathématicien par ses manipulations continuelles de formules. Il est donc essentiel d'atteindre une ergonomie soignée et une grande expressivité de manipulation de formules pour notre approche d'assistance. Nous définissons des spécifications pour ces manipulations.

L'intégration de systèmes informatiques et la gestion informatique des formules constituent alors nos propositions de **modèles informatiques pour une mise en œuvre ergonomique** (ch. 3). Les spécificités informatiques du **traitement du Langage Mathématique** sont ainsi détaillées pour le cas des formules (ch. 4).

Un autre thème important concerne l'**organisation des connaissances** dans un domaine conceptuel aussi riche que les mathématiques (ch. 5). Une étude des besoins et des propositions permet d'évaluer le travail de modélisation mathématique et informatique qui reste à effectuer pour prendre en compte un domaine réaliste. L'étude linguistique de traités mathématiques permettrait de faire progresser la description des domaines mathématiques.

Le dernier thème abordé concerne l'**opérationnalisation des formules**, c'est-à-dire l'utilisation d'identités mathématiques à partir de leur saisie en Langage Mathématique (ch. 6). A partir d'exemples concrets, nous proposons des mécanismes et une expertise générale, susceptibles de sélectionner et d'utiliser les formules pour faciliter la résolution de problèmes.

Face au transfert vers un support informatique de l'activité des mathématiciens, il n'y avait jusqu'à présent que des réponses limitées. Cette étude présente une analyse de ce problème, propose une solution globale, et définit des repères nécessaires à l'élaboration de solutions à venir. Notre apport est donc délibérément qualitatif, même si des spécifications informatiques sont produites. L'approche développée convient particulièrement à l'Assistance au Travail Scientifique.

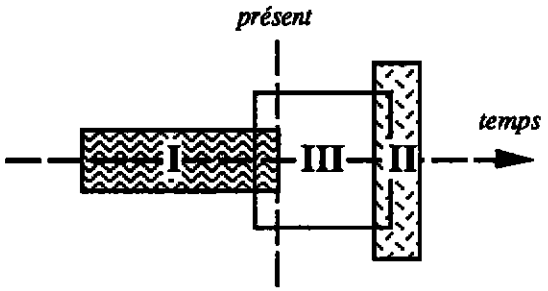
Cette étude peut être abordée avec des objectifs différents. Aussi notre conclusion rassemble des résultats importants en abordant différents points de vue, sans se cantonner uniquement à notre thème d'étude sur l'assistance au mathématicien. Les ramifications déjà présentes lors du développement de notre démarche traduisent le caractère ouvert et interdisciplinaire de cette étude.

---

[2] Le Réseau Européen pour les Sciences Cognitives (RESCO), regroupant des universitaires et des industriels toulousains, a adopté dans ses statuts la définition suivante :
Les Sciences Cognitives consistent en : "la représentation, l'acquisition et l'utilisation des connaissances pour l'analyse, la conception et l'évaluation des interactions entre agents intelligents naturels ou artificiels".

PARTIE I

Le Langage Mathématique



1. LE TRAVAIL DU MATHEMATICIEN

1.1. NOTRE APPROCHE DES MATHEMATIQUES

1.1.1. Le langage en Mathématiques

> *Toute progression géométrique* — a : b : c : d : e : f : g : h : i : k, *donne cette proportion : le premier terme est au second, comme la somme de tous les termes, moins le dernier, est à la somme de tous les termes moins le premier ; c'est-à-dire,* a : b :: s - k : s - a.

Cet énoncé, extrait du cours de mathématiques de Bossut [Bossut M. DCC. LXXXII.], illustre les progrès réalisés en mathématiques pour les concepts, les notations et les critères d'intérêt. Cet énoncé est toujours vrai mais l'intérêt de ce résultat est mineur : il n'est plus enseigné ni mémorisé par les étudiants. L'évolution des mathématiques s'est donc accompagnée d'un déplacement des centres d'intérêt du mathématicien.

L'évolution du langage rend en effet cette "recette" inutile. Cette évolution ne s'est pas produite par une révolution des mathématiques : la notion de progression géométrique existe toujours et cette recette s'exprimerait par une phrase similaire de nos jours. Par contre, l'évolution du langage s'est traduite par une évolution de l'écriture, accompagnée d'une introduction de concepts comme celui de sommation. C'est cette quête d'abstraction qui constitue la véritable révolution.

Les mathématiques évoluent avec leur langage. Elles se sont attachées à développer un système de représentation formel et précis. Les concepts "inventés et découverts" se cristallisent dans le langage qui évolue parallèlement. Réciproquement, l'introduction et l'affinement d'un système de notations permet aux concepts mathématiques de s'enrichir ou d'être repensés en favorisant l'abstraction. Le langage contribue ainsi en retour à la dynamique de développement des mathématiques.

L'**hypothèse de "Shapir-Whorf"**, énoncée par Shapir (1921) et développée par Benjamin Lee Whorf (1956) traduit cela : elle exprime que la structure de chaque langue naturelle détermine une perception du monde correspondante.

Les mathématiciens ont ainsi développé et affiné leur outil de travail au fur et à mesure que la conceptualisation et la formalisation de théories progressaient. La diffusion culturelle des mathématiques permet alors au langage de transcender en partie les barrières des langues naturelles.

Contrairement à une idée répandue, ce n'est pas l'aspect synthétique des notations qui rend désuet l'énoncé précédent. La formule

$$\frac{a_0}{a_1} = \frac{\sum_0^{n-1} a_i}{\sum_1^{n} a_i}$$

n'est pas plus facile à comprendre ou à mémoriser que la phrase "historique". Au contraire, cette formule constitue un niveau d'abstraction supérieur dont la phrase est une sorte d'interprétation : comprendre cette formule consiste à reconstituer la phrase.

Ce qui rend l'énoncé désuet est que l'évolution du langage l'a rendu inutile ! Pourquoi retenir en effet un résultat qui coule de source ? En exprimant le caractère géométrique de la suite $\left(q^i a_0\right)_{i\geq 0}$ on obtient :

$$\frac{\sum_0^{n-1} a_i}{\sum_1^{n} a_i} = \frac{\sum_0^{n-1} q^i a_0}{\sum_1^{n} q^i a_0} = \frac{\sum_0^{n-1} q^i a_0}{\sum_1^{n} q\, q^{i-1} a_0} = \frac{\sum_0^{n-1} q^i a_0}{q \sum_0^{n-1} q^i a_0} = \frac{1}{q} = \frac{a_0}{q\, a_0} = \frac{a_0}{a_1}$$

L'intérêt des notations est que leur aspect synthétique permet au mathématicien de raisonner indépendamment de leur sens. **L'évolution historique du langage a renforcé son utilité.** C'est l'utilité des manipulations de la sommation et de l'expression d'une suite géométrique qui permettent de retrouver ce résultat simplement.

1.1.2. Regards sur l'activité mathématique

Le terme *mathématicien* est employé dans cette thèse pour désigner un comportement communautaire interne aux mathématiques, et non pour désigner un professionnel de la construction interne des mathématiques. Lorsqu'une personne est en phase d'acquisition d'un savoir-faire, nous utilisons le terme *apprenant*. Face à un système informatique, nous qualifions d'*usager* un mathématicien, un ingénieur, ou un étudiant qui mène une activité de conception mathématique ou de manipulation d'un formalisme mathématique ; nous utilisons sinon le terme d'*analyste* pour la modélisation d'une activité, ou de *développeur* pour sa mise en œuvre.

Nous employons le terme *activité mathématique* pour regrouper les activités du mathématicien relatives à son exploitation des mathématiques. Nous parlons d'*une* activité mathématique lorsque nous précisons sa nature. Le mathématicien utilise dans son activité des représentations mentales sur lesquelles il raisonne. Nous qualifions de *données mathématiques* les ingrédients qui constituent ces représentations. Le *langage* sert à décrire et à partager certaines de ces données entre plusieurs mathématiciens, mais d'autres données sont personnelles ou n'ont pas d'équivalent dans le langage. Certaines données partagées et manipulées sont appelées *objets mathématiques* (par ex. le nombre π). Les données permettant de décrire d'autres données partagées sont appelées *concepts mathématiques* (par ex. la structure de groupe). Nous faisons appel au sens commun pour étendre ces "définitions" lorsque surviennent d'autres exemples.

La *psychologie cognitive* est la partie de la psychologie qui étudie la façon dont l'homme traite la connaissance. Elle a pour but de construire des modèles et des théories permettant d'expliquer ou de prédire son comportement cognitif. L'ergonomie étudie l'adéquation de l'homme à son environnement de travail. L'*ergonomie cognitive* étudie ici l'adéquation d'un système informatique à la conduite d'une activité.

Ces disciplines introduisent la distinction entre *tâches* et *activité* : l'activité correspond à ce qui se fait, tandis que les tâches décrivent ce qui est à faire.

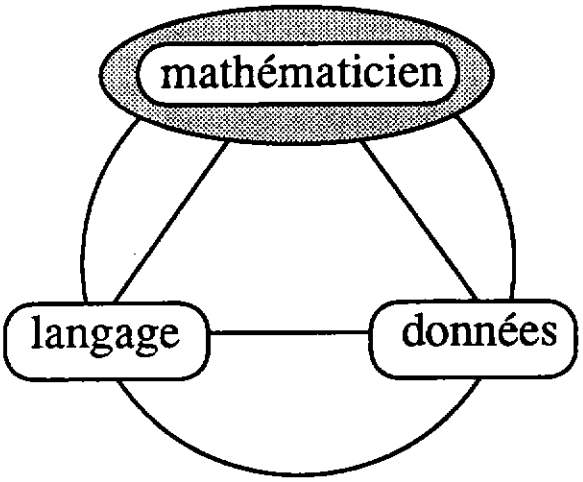
L'activité peut se décomposer en une *activité cognitive* et une activité observable externe dite *activité manifeste* ou comportement. Il y a deux types d'activités selon que la situation est connue ou nouvelle : celles qui sont fondées sur des règles et des procédures existantes en mémoire, et des activités de résolution de problème. Les connaissances intervenant lorsque la situation est connue sont qualifiées de *connaissances de surface* par opposition aux *connaissances profondes* qui font intervenir une compréhension plus fine des causes et des rouages de la situation.

Les tâches permettent soit de modéliser un raisonnement dans une activité de résolution de problème, soit d'expliquer le comportement observé d'un agent. Une tâche est alors un concept unificateur de traitement de l'information qui discrétise le processus continu qui produit l'activité.

Il est particulièrement pertinent de **différencier les tâches et l'activité en fonction de l'agent qui les décrit.** En effet, il y a toujours un décalage de communication entre ce qui est demandé et ce qui est compris. De plus, il y a aussi un décalage entre la façon dont un agent décrit ce qu'il fait et ce qu'il fait réellement. Cela s'explique partiellement par la distinction entre description et utilisation : un agent dispose d'une description et se construit des compétences d'utilisation. Cet écart est souvent important car la description relève souvent d'un modèle général qui ne permet pas de décrire finement une activité, voire simplement de la décrire selon un point de vue local.

Pour un observateur extérieur idéal, il y donc une *tâche prescrite* en début de chaîne et une *tâche effective* en sortie. La synthèse précédente suggère que les biais possibles sont importants. Elle montre de plus que **pour comprendre l'activité mathématique, il ne suffit pas de demander au mathématicien comment il la décrit.** Nous nous attachons néanmoins à étudier l'activité mathématique afin de la décrire et d'en trouver les tâches effectives. Il faut donc pour cela étudier d'autres sources que les verbalisations des mathématiciens.

Afin de mieux cerner cette activité mathématique, il est bon d'en introduire les acteurs, ou plus simplement les paramètres influents :



Les acteurs de l'activité mathématique

figure 1.1

Nous qualifions d'acteurs le langage et les données que le mathématicien utilise, car leurs caractéristiques sont essentielles pour **permettre une activité mathématique exploitable par un mathématicien.** Voyons en quoi une altération de ces acteurs modifierait aussi l'activité mathématique.

Nous avons esquissé précédemment le rôle historique du langage dans l'activité mathématique. D'après l'hypothèse de Shapir-Whorf, revenir aux usages langagiers du 18ème siècle reviendrait à dénaturer l'activité mathématique actuelle. Par ailleurs, un mathématicien ne saurait fonctionner correctement sans une forme visuelle adéquate ou sans les facilités de manipulation que lui procure son langage usuel. Chercher à altérer le langage ne peut se faire que par évolution du langage pratiqué dans la communauté mathématique. Le rôle actif du langage se mesure alors par sa résistance au changement, qui doit être argumenté et validé par une amélioration de la pratique de l'activité mathématique.

Les données mathématiques jouent aussi un rôle actif dans l'activité. Un révélateur est la conception platonicienne de certains objets de manipulation mathématique. Leur donner une existence propre n'est-il pas le meilleur moyen de prôner leur autonomie ? Un deuxième argument est que les données intègrent certaines parties du langage (une formule est par exemple un objet sur lequel le mathématicien va raisonner). Ce moyen d'attribuer une forme tangible aux données leur donne aussi une existence tangible et un rôle actif. Enfin, lorsque certaines données intuitives sont absentes ou masquées, le mathématicien perd le sens de ses actions et n'assure plus le guidage opportun de son activité. La présence de certaines données est donc un aspect important de l'activité mathématique.

1.1.3. Quelques approches de l'activité mathématique

> Je dirai que j'ai trouvé la démonstration de tel théorème dans telles circonstances ; ce théorème aura un nom barbare, que beaucoup d'entre vous ne connaîtront pas ; mais cela n'a pas d'importance : ce qui est intéressant pour le psychologue, ce n'est pas le théorème, ce sont les circonstances.
>
> Henri Poincaré

Ce qui est intéressant pour notre analyse, ce n'est pas le théorème, ce sont les moyens mis en œuvre pour le trouver, le vérifier, le représenter et l'utiliser. C'est pourquoi nous nous attacherons ici à **décrire l'activité mathématique humaine** en relation avec sa principale trace observable : le langage, ainsi que les connaissances qu'il permet d'exprimer. Notre but est de modéliser la part observable de cette activité mathématique plutôt que de recenser un corpus de théorèmes mathématiques.

L'importance des acteurs une fois mise en avant, il nous reste à préciser la nature de l'activité mathématique. Hormis dans quelques micro-domaines abordés pour des besoins informatiques, il n'existe pas à notre connaissance de modèle satisfaisant de l'activité mathématique. Le modèle traditionnel s'inspire de la logique pour formaliser les déductions. Nous en discutons ultérieurement les limites. Les mathématiciens les perçoivent intuitivement puisqu'ils n'adhèrent guère aux idéaux logicistes.

Par ailleurs, l'activité mathématique est souvent présentée comme une activité de résolution de problèmes, un point c'est tout. Ce cliché est réducteur et préjudiciable ! Cette limitation n'est pas représentative du travail des mathématiciens. Une majorité du temps des mathématiciens est consacrée à leur propre apprentissage, à l'étude de textes et de traités, à l'enseignement, à l'argumentation verbale ou écrite, ou tout simplement à la réflexion. Lorsque, au cours de ces activités, un problème mathématique est posé, c'est souvent un problème déjà résolu montré à titre d'anecdote, d'exemple, d'illustration de méthode, de vérification du raisonnement, etc.

Ce qui compte alors chez les mathématiciens, ce n'est pas uniquement leur capacité de résolution, mais aussi leur capacité d'utilisation du langage et de compréhension intuitive des concepts mathématiques :

Le langage est la clé de l'activité mathématique

alors que la résolution de problème n'est qu'une des facettes de la créativité des mathématiciens. Malheureusement, les outils mathématiques disponibles sont essentiellement des outils de calcul. Le mathématicien a essentiellement besoin d'outils de communication, mais les connaissances actuelles ne permettent pas de proposer mieux que des traitements de texte spécialisés.

Cette conception de l'activité mathématique permet d'évaluer les tentatives effectuées par ailleurs pour la modéliser. Le trièdre des acteurs de l'activité mathématique (figure 1) va nous servir de référentiel pour évaluer quelques approches proposées. Bien que nous reprendrons par ailleurs certains des résultats des dites approches, nous nous attachons ici à en apprécier leurs limites.

Par exemple les traitements de texte permettent au mathématicien l'usage du langage sans s'occuper des données. Ils relient donc exclusivement l'axe gauche du trièdre. Le traitement du Langage Naturel améliore cet axe en introduisant des données. Toutefois il n'existe pas de système spécifique aux mathématiques car la simple reconnaissance d'une phrase est bien insuffisante pour exploiter les connaissances du mathématicien. Un système de traitement du langage est insuffisant si les données ne sont pas conçues pour un usage mathématique.

Le calcul numérique est une caricature de l'axe droit reliant le mathématicien aux données. Il cherche à déterminer empiriquement les propriétés d'un objet en laissant de côté toute recherche d'abstraction. Le calcul symbolique y remédie mais n'exploite qu'un part minime des représentations du mathématicien, et n'est possible que sur des domaines de données limités. La puissance de calcul des ordinateurs est majoritairement inutilisable faute d'être intégrée à une activité mathématique complète.

L'intérêt pratique du calcul nous amène à distinguer entre **faire des mathématiques** et **exploiter des résultats mathématiques** :

- Exploiter des résultats mathématiques consiste à considérer les mathématiques comme un produit clé en main. Les mathématiques forment alors un outil universel permettant la constitution de modèles adaptés à une grande classe d'applications.
- faire des mathématiques consiste à s'intéresser au développement de l'outil universel lui-même afin de le compléter et de l'améliorer. Cela consiste bien souvent à vérifier certains rouages ou produire des extensions de l'outil à l'aide de ce même outil universel.

Le paradoxe de l'outil universel fait que les mathématiques sont à la fois "objet" de l'étude et "outil" pour la réaliser. Ainsi, tout résultat mathématique est à la fois un outil utilisé pour construire de nouveaux résultats, et un objet d'étude qu'il a fallu construire et vérifier pour agrandir les fonctionnalités globales de l'outil universel.

Faire des mathématiques, et donc l'activité mathématique, se définit alors d'après l'objectif poursuivi et non d'après les moyens utilisés. Nous considérons que les calculs ou les raisonnements produits lors de l'activité mathématique sont destinés à être décrits, communiqués, expliqués et vérifiés. La conduite des calculs et raisonnements prend alors plus d'importance que la production d'un résultat.

La formalisation logique et les outils de vérification de preuve peuvent alors prétendre à modéliser l'activité mathématique. Comme ils omettent volontairement le rôle et l'action du mathématicien, ils se situent sur l'axe horizontal du trièdre reliant le langage aux données. Malheureusement, cette schématisation a profondément atteint la nature du langage et atomisé les concepts mathématiques. Elle perd notamment tout support des représentations intuitives et s'aliène ainsi un guidage éventuel du mathématicien.

Les outils de démonstration automatique de théorèmes basés sur les connaissances des mathématiciens sont une modélisation partielle des données de l'activité mathématique. Même s'ils cherchent à imiter l'activité du mathématicien, ils se privent de l'interaction créative entre le mathématicien, son langage et ses données. Leur tentative d'autonomie les rend de plus inaptes à servir d'assistance au mathématicien.

Citons enfin des approches différentes visant à étudier indirectement l'activité mathématique. C'est le cas de la pédagogie et de la psychologie qui s'intéressent principalement au mathématicien. La linguistique quant à elle, a traditionnellement peu d'intérêt pour les activités scientitiques et même les procédés langagiers du mathématicien restent à explorer. Ces disciplines n'ont qu'une vue très partielle sur les autres sommets du trièdre mais apportent des perspectives indispensables à la prise en compte d'une activité mathématique globale.

1.1.4. Une étude des Mathématiques à partir du langage

Nous voulons :

modéliser l'activité mathématique

afin de fournir des outils informatiques pour assister le travail du mathématicien.

Plus spécifiquement nous voulons **assister le cycle de production de démonstrations** :
- définition d'un problème ;
- construction et vérification de la preuve du problème ;
- présentation textuelle et diffusion d'une démonstration.

Notre objectif principal d'assistance est alors de **faciliter les productions langagières.**

Nous nous proposons d'étudier cette activité mathématique à partir du langage qui lui sert d'expression. Le langage permet en effet d'appréhender la composante sociale du mathématicien. Cette composante est partagée par la communauté mathématique à un moment donné et est relativement indépendante d'un mathématicien particulier. De plus, le langage véhicule les connaissances et les modes de traitement des mathématiciens. Le langage va ainsi de pair avec les données et son changement implique une autre conceptualisation et une autre organisation des données de travail des mathématiciens.

Cette étude relève des sciences humaines associées à des préoccupations informatiques. Cette approche peut donc faire l'objet d'une problématique **interdisciplinaire** autour de l'activité mathématique par rapprochement de ces trois thèmes - informatique, mathématique et sciences humaines. L'activité mathématique peut alors être indifféremment examinée pour ses fondements, pour ses caractéristiques dynamiques, par les connaissances qu'elle manipule, ou par l'usage du langage. Ces deux derniers points nous concernent plus particulièrement.

Nous qualifions cette approche de *naturelle* car :
- elle résulte de l'observation et de la modélisation de l'activité du mathématicien dans le milieu naturel du développement et de la pratique de cette activité.
- elle respecte l'observation des trois acteurs (mathématicien, langage et données) dans une transcription sur un support informatique.
- elle utilise enfin les capacités de traitement informatique au service du mathématicien, selon le paradigme d'un assistant-mathématicien.

Cette approche naturelle s'inscrit dans la lignée des Sciences Cognitives (linguistique, psychologie cognitive, ergonomie cognitive, etc.). Elle adopte ainsi des choix inusuels :
- ne pas imposer un degré d'automatisation a priori ;
- produire une modélisation fine de l'activité du mathématicien de façon à savoir reproduire des tâches élémentaires ;
- faire au plus clair pour le mathématicien, et non faire au plus simple pour une théorie.

Les caractéristiques humaines de traitement sont différentes de celles facilement exhibées par une machine. En particulier l'activité mathématique humaine recherche une concision du langage et exploite de

nombreuses informations implicites. L'informatisation de ces capacités requiert une mise à niveau de la machine tant pour les connaissances utilisées que pour leurs processus d'exploitation. Les tâches supportées par l'assistance se limitent à manipuler les parties observables de l'activité mathématique (des formules jusqu'aux démonstrations) en exploitant les processus mathématiques de construction (par ex. les déductions).

Pour l'exemple d'une construction de preuve, il s'agit de laisser l'initiative au mathématicien et de bénéficier de connaissances "naturelles" pour construire des étapes pertinentes de la voie de résolution humaine. Cette approche autorise plusieurs degrés d'automatisation et d'explication. L'approche opposée propose un système automatique de production de preuves qui utilise des points de passage différents de ceux du mathématicien pour produire une preuve illisible, même quand il réussit...

Les problèmes de notre approche interdisciplinaire sont que :
- les mathématiciens n'ont pas conscience des potentialités d'assistance ;
- la plupart des problèmes soulevés sont inédits et il n'existe guère de modèles exploitables ;
- la modélisation de l'activité mathématique se distingue des solutions informatiques habituelles.

Nous proposons pour cela une approche ascendante de l'assistance commençant par la maitrise progressive des tâches humaines élémentaires. Des tâches sont qualifiées d'élémentaires lorsque leur exécution fait partie de processus considérés comme secondaires, car partagés par tous les usagers hormis quelques débutants. De telles tâches sont par exemple l'analyse d'une formule, le rappel des propriétés d'un objet, l'application d'un théorème ou la description textuelle de ces dites opérations. Nous qualifions de *manipulation élémentaire* toute action de manipulation de formule ou d'un ensemble de connaissances, qui puisse s'exprimer simplement dans un langage d'expression de manipulations issu de la pratique des mathématiques.

Notre étude de l'activité mathématique va ainsi préciser ces tâches élémentaires et mettre en valeur leur rôle dans les aspects créatifs. Quoique originale, cette **approche ascendante sous forme d'une assistance à la conception** est nécessaire et motivée par la volonté de respecter la liberté du concepteur-mathématicien dans ses tâches peu automatisables. L'usage du Langage Mathématique usuel est pour nous déterminant pour la conduite d'une activité mathématique naturelle. Bien manipuler les formules et les phrases du Langage Mathématique constitue en effet un prérequis indispensable à l'assistance au mathématicien.

En résumé, cette partie étudie le Langage Mathématique afin de déterminer les moyens d'assistance possibles de l'activité mathématique humaine. Elle ne s'intéresse pas directement aux mécanismes de l'activité mathématique tels qu'ils sont usuellement étudiés en logique, en démonstration automatique de théorèmes, en calcul formel, ou en pédagogie des mathématiques. Au contraire, elle effectue une **analyse de l'activité mathématique naturelle en vue de la conception d'un outil d'assistance.**

1.2. UNE CLASSIFICATION DES ACTIVITES DU MATHEMATICIEN

1.2.1. Caractérisation de l'activité mathématique

L'activité mathématique est avant tout un travail de conception.

Nous définissons une *activité de conception* comme une activité où le savoir-faire et les aptitudes de l'homme sont déterminants. Une telle activité est essentiellement conceptuelle, par opposition aux activités de guidage, contrôle ou diagnostic qui sont contraintes par les états d'un système. Ces activités centrées sur un système dynamique complexe relèvent d'une théorie d'assistance à l'opérateur faisant une large part à la prévention des erreurs [Boy 88, Mathé 90]. Par opposition, les concepts manipulés dans une activité de conception sont indépendants des spécificités d'un système matériel et résultent souvent d'une maturation historique.

L'activité mathématique se définit de plus comme un *travail scientifique*. Ce travail est le fruit d'une activité de réflexion dirigée vers sa réalisation. Il doit être construit, justifié, argumenté, reproductible, etc. Il conserve néanmoins une composante artistique certaine. Le travail mathématique partage enfin une propriété fondamentale avec le travail de l'écrivain : **le langage est la trace ultime de l'activité qui l'a produit.** Rien d'étonnant dès lors que la liberté d'expression de la forme et du contenu soit une composante essentielle de l'activité mathématique.

Il y a ainsi une distinction entre l'objet de l'activité et le langage qui lui sert d'expression. Pourtant, cet outil indispensable qu'est le langage nécessite aussi des capacités de traitement de la part du mathématicien. Cette distinction entre la recherche mathématique et le langage qui lui sert d'expression se répercute alors dans les tâches gérées par le mathématicien. Le mathématicien mêne alors de front deux activités (figure 1.2) :

- *l'activité purement mathématique* ;
- *l'activité paramathématique* ;

Par opposition à l'activité purement mathématique, l'activité paramathématique traite de l'usage du langage et se préoccupe uniquement des entités langagières. Cette partition permet de distinguer (sans pour autant y souscrire) entre ce qui est souvent considéré comme un monde platonicien immatériel, impersonnel et intemporel dans lequel opèrent les mathématiques, et un monde pratique qui concerne les moyens d'investigations et les productions matérielles.

Notre abord de l'activité mathématique étudie principalement l'activité paramathématique. Elle est en effet mal connue car elle fait l'objet d'automatismes d'usage du langage. Elle est de plus largement consommatrice de tâches élémentaires pour lesquelles on peut espérer apporter une assistance notable.

Partition de l'activité mathématique

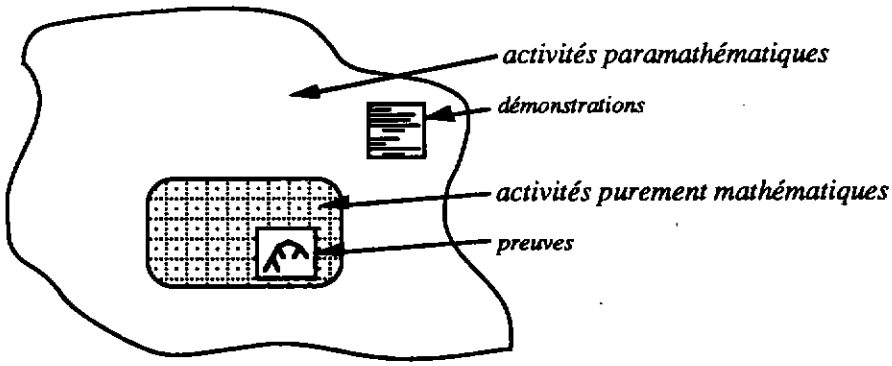


figure 1.2

Le mécanisme d'interprétation des expressions, le choix de critères lexico-syntaxiques et la présentation d'une preuve formelle font partie de l'activité paramathématique. Ces deux activités sont néanmoins étroitement corrélées. Par exemple, la production d'inférences dans un système formel est une activité purement mathématique. Leur mise en forme visuelle et leur écriture est une activité paramathématique.

Nous ferons ainsi le distingo entre preuve et démonstration, une ***preuve*** étant l'objet fictif construit par le mathématicien tandis que la ***démonstration*** en est une présentation textuelle.

Une démonstration a alors essentiellement une fonction de communication. La démonstration se place dès lors fréquemment à une métaniveau explicatif vis à vis de la preuve dont elle cherche à communiquer les points essentiels tout en permettant sa vérification par autrui. Cette notion d'explication et d'argumentation est l'un des acquis majeurs de la pensée grecque. Cela a permis aux mathématiques de transcender la notion utilitaire mais figée des recettes de calcul, en ouvrant la voie de leur compréhension. Pierre Lévy y voit même les fondement de la démocratie...

1.2.2. Description des processus mentaux du mathématicien

Afin d'étudier l'activité mathématique et les caractéristiques de son **assistance au travail**, nous distinguons trois classes de processus mentaux[1], selon le schéma figure 1.3 :

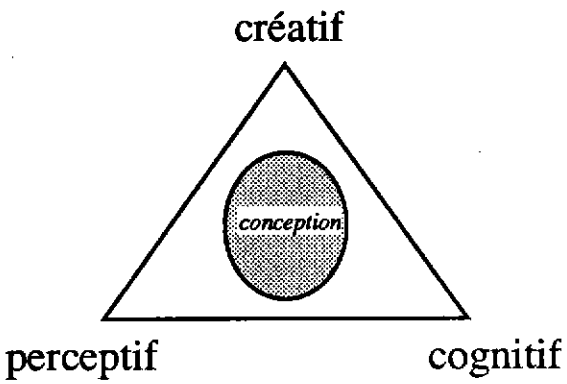


figure 1.3

Les **processus cognitifs** sont ici limités à leur sens étymologique : Connaître. Ils concernent la gestion et la manipulation des structures sémantiques représentant les connaissances. En ce sens, ils se rapprochent de la problématique de compréhension et d'utilisation du Langage Naturel. Les **processus perceptifs** sont mis en œuvre pour relier la conduite d'une activité aux connaissances qu'elle utilise. Ils supportent les manipulations élémentaires dans leur composante visuelle comme dans leur composante opérative (par exemple : réalisation d'une suite de calculs). Les environnements de programmation constituent un domaine technique qui cherche à favoriser l'usage de ces processus perceptifs.

Afin de les différencier de ces processus élémentaires, nous avons regroupé les tâches de haut niveau d'abstraction dans les **processus créatifs.** Dans une activité de conception, la recherche et le raisonnement font appel à ces processus créatifs, en collaboration étroite avec les processus cognitifs et perceptifs. Le Génie Logiciel ou la CAO sont confrontés à l'assistance de ces processus créatifs et proposent des réponses techniques à cette classe de problèmes.

La nature et le fonctionnement de ces processus créatifs sont mal connus. La principale étude date de 1945 [Hadamard 75]. Jacques Hadamard y adapte et développe un modèle de pensée créative toujours d'actualité, emprunté à Graham Wallas [Wallas 26]. Il y propose quatre phases successives, conscientes ou inconscientes : préparation, incubation, illumination, puis vérification et "finition". L'intérêt principal de l'essai est son illustration des représentations inconscientes, subconscientes ou conscientes.

Indépendamment des solutions techniques, les **Sciences Cognitives** étudient directement ces divers processus. L'ergonomie cognitive permet en particulier de proposer et de valider des solutions meilleures. Réciproquement, l'étude des mathématiques permet de proposer un modèle ou des expériences concernant les processus élémentaires d'un domaine conceptuel étendu. Un tel système autorise alors l'exploration des mécanismes de fonctionnement des processus créatifs.

Malgré les progrès de la psychologie cognitive, il n'est pas encore possible de donner un modèle mental détaillé d'un mathématicien à l'œuvre. Un tel modèle permettrait d'organiser les données mentales utilisées lors d'une activité de conception et d'expliquer les opérations effectuées en terme de processeur d'information humain.

Le mathématicien dispose notamment :
- d'un ensemble de connaissances sur le monde et sur lui-même ;
- d'un espace de travail avec des représentations des buts, des intentions, un centre d'intérêt, un historique des actions, etc ;
- de moyens d'action, internes ou externes.

---

[1] D'après une terminologie utilisée par J.M. Ledizés pour "l'apport des systèmes experts pour l'aide à la conception" [Ledizés 88].

Pour les mettre en œuvre, il utilise des capacités de traitement conscientes ou inconscientes (planification et inférences dirigées, procédures familières, classification, généralisation, intuitions, perception, etc) Cela permet au mathématicien de réaliser certains types d'activités telles la résolution de problèmes, la prise de décision, la gestion du langage, l'apprentissage, l'imagination ou l'action.

Dans "Psychologie cognitive : modèles et méthodes" [Caverni 88, p30], Claude Bastien rapporte néanmoins un modèle de résolution de problèmes mathématiques qualifiée de provisoire en 1984 par ses auteurs : Dinnel, Glover et Ronning. Ce modèle considère une mémoire externe représentant l'environnement de la tâche, une mémoire à long terme et une mémoire de travail composée de trois modules : «

- représentation du problème dont les fonctions sont de lire, interpréter, reconnaître, construire un but ;
- planification qui construit et exécute la stratégie de résolution ;
- compte-rendu qui écrit, édite et vérifie. »

Un tel modèle reste très général et n'est guère utile pour représenter par exemple comment le mathématicien effectue une manipulation algébrique ou une résolution d'un problème géométrique. Les mathématiques ont donc besoin de modèles détaillés, qu'il faudra synthétiser ultérieurement par des modèles plus généraux et psychologiquement validés pour des activités de perception et de manipulation de connaissances.

1.2.3. Les phases de l'activité mathématique

L'activité du mathématicien peut être décomposée en fonction de l'évolution de ses connaissances suivant quatre phases :

① une phase d'**apprentissage** du savoir-faire ou des connaissances d'un domaine. L'apprentissage est un processus permanent de toute activité humaine.

② une phase d'**utilisation** des connaissances mathématiques déjà constituées comme support d'expression et de résolution de problème.

③ une phase de **constitution** (acquisition, construction, classification et organisation) des connaissances d'un domaine de l'univers mathématique.

④ une phase de **diffusion** des connaissances et des résultats mathématiques. L'enseignement en est une forme verbale, la publication une forme textuelle.

Les quatre phases de l'activité mathématiques sont alors toutes indispensables, même si nous nous intéressons prioritairement dans cette étude à la phase d'utilisation. De plus, ce découpage en phases est nullement exclusif, et correspond plutôt au processus dominant. Par exemple, lors d'une phase d'utilisation des connaissances pour résoudre un problème, le mathématicien peut être amené à compléter les connaissances constituées ou à y ajouter un lemme résultat. Une démonstration peut être simultanément construite afin d'assurer la diffusion de la solution. Cette solution peut faire intervenir une combinaison d'heuristiques qui viendront enrichir le savoir-faire du mathématicien.

Ces quatre phases sont symbolisées figure 1.4 : la place centrale de l'apprentissage traduit l'influence de l'individu par rapport à la communauté mathématique située à l'extérieur. Nous avons adopté le point de vue d'un mathématicien, par opposition à l'édifice mathématique communautaire. Par exemple, la phase de constitution peut être envisagée par l'étude historique des connaissances de la communauté mathématique (*phylogénétique*), ou par l'étude de l'évolution des connaissances d'un individu (*ontogénétique*). Cela relève directement de l'épistémologie, définie par Jean Piaget comme « l'étude de la constitution de connaissances valables » [Pleiade 67, p 6] ; la connaissance étant définie par la mise en relation (et interaction) par un sujet pensant entre un objet (ici mathématique) et des structures de connaissances. Piaget distingue ultérieurement le processus de constitution des connaissances, de l'état (soumis à analyse) des connaissances une fois constituées.

Le cycle utilisation constitution

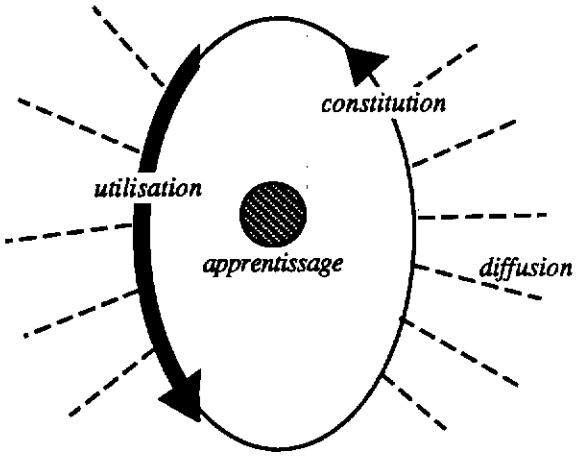


figure 1.4

La diversité de l'activité mathématique a été étudiée sous de nombreux aspects. Citons notamment les ouvrages de (cf. les références bibliographiques) :

- P. J. Davis et R. Hersh pour l'univers et l'empire des mathématiques ;
- J. Hadamard pour l'analyse de la pensée créative chez les mathématiciens ;
- I. Lakatos pour l'évolution des mathématiques par preuves et réfutations ;
- J. Piaget pour l'analyse de l'activité mathématique par l'épistémologie génétique ;
- G. Pólya pour la pratique des heuristiques ;
- A. Schoenfeld sur la résolution de problèmes mathématiques ;
- D. Solow pour le savoir-faire élémentaire de démonstration ;

Un mythe tenace est que les mathématiques peuvent être réduites à l'exploitation d'un noyau de connaissances bien identifiées. De même, les connaissances ne peuvent pas facilement être dépersonnalisées, détemporalisées et décontextualisées. L'étude de ces ouvrages montre que les connaissances explicitement présentées comme telles dans les traités mathématiques sont minoritaires. L'essentiel des connaissances nécessaires à l'activité mathématique est à notre avis implicite et relative à l'activité paramathématique.

1.2.4. Typologie des tâches de l'activité mathématique

Cette section regroupe diverses tâches accomplies par le mathématicien lors de ces quatre phases. Ces tâches correspondent à des activités explicites susceptibles d'occuper longtemps un mathématicien. Elles permettent de catégoriser l'activité quotidienne d'un mathématicien afin d'évaluer l'importance de l'activité paramathématique pour les mathématiques.

Les tâches relatives à la phase d'apprentissage sont pour l'apprenant : observer, comprendre, questionner, se convaincre, s'entraîner, généraliser, s'évaluer, pratiquer, etc. Les tâches sont complémentaires pour l'enseignant : elles mettent en avant les notions de diagnostic et d'explication.

Les tâches relatives à la phase de diffusion sont pour les acteurs la mise en forme, le stockage, la diffusion. Elle se traduisent par exemple par des traités, des articles ou des conférences. Les tâches relatives aux récepteurs sont la documentation, la formation continue, la synthèse de l'état des lieux, la demande d'information, l'enquête, etc.

Les tâches relatives à la phase de constitution sont de répertorier, classifier, faire la synthèse, modéliser, définir, savoir comment utiliser, etc. Elles se cantonnent fréquemment aux aspects statiques d'un domaine mathématique car elles ont des difficultés à définir les caractéristiques dynamiques. Ces caractéristiques dynamiques sont alors souvent implicites et l'expertise en est acquise lors du processus d'apprentissage. Un des apports des mathématiques a consisté à les modéliser sous forme d'algorithmes, de méthodes ou de savoir-faire heuristique.

La simple lecture des tâches précédentes montre l'importance de l'activité paramathématique et le temps qui lui est directement consacré. Les tâches relatives aux quatre phases se recoupent par leur nature bien qu'elles correspondent à des objectifs différents. Il est donc normal d'en retrouver certaines dans la phase d'utilisation que nous allons maintenant détailler.

Nous avons décomposé les tâches de la phase d'utilisation en **inspiration, formulation et action** selon une modalité créative ou pratique. Les objectifs des tâches d'inspiration sont de trouver des aspects encore inconnus, ceux de la formulation sont de verbaliser et préciser l'objet des recherches, tandis que ceux de l'action sont de construire les solutions attendues. La différence d'objectif dénote une plus ou moins forte connaissance a priori du résultat escompté, et un champ de recherche plus ou moins ouvert selon la modalité.

S'il fallait résumer ces points, nous dirions que la tâche est dirigée respectivement par les données, par sa compréhension ou par son but. Nous obtenons pour :

- inspiration créative :
  chercher
  Il s'agit principalement d'une recherche d'un savoir nouveau comme chercher des problèmes à résoudre, de nouvelles conjectures, de nouvelles méthodes de preuves, etc. Elle peut s'inspirer des données existantes en générant des problèmes presque semblables, analogues, etc.

- inspiration pratique :
  analyser
  il s'agit d'apprécier les données existantes afin d'en générer une description, d'effectuer des comparaisons, les identifier, les classifier ou améliorer leur organisation. Elle procède plus par une analyse de l'existant et de son adéquation aux problèmes rencontrés.

- formulation créative :
  expliquer
  Il s'agit de mettre en évidence les raisons profondes de la validité d'une proposition. A ce titre, elle utilise la compréhension intuitive personnelle du mathématicien.

- formulation pratique :
  définir
  Il s'agit de poser un problème, définir un concept, formaliser une méthode. Elle utilise alors les moyens offerts par le langage pour modéliser des caractéristiques existantes par ailleurs.

- action créative :
  résoudre
  C'est la tâche la plus couramment employée dans les exercices de mathématiques. Elle a donc de nombreuses variantes linguistiques selon que le résultat est connu (par ex. prouver que, établir, montrer que) ou inconnu (par ex. trouver un x tel que P(x)). Elle exploite librement l'expérience acquise lors de la résolution d'autres problèmes.

- action pratique :
  utiliser
  La tâche est ici bien intifiée. Il s'agit d'effectuer une opération comme utiliser un théorème, calculer, simplifier, mettre sous telle forme, construire une démonstration à partir d'une preuve. Chaque catégorie d'objet mathématique a ses variantes spécifiques. Les moyens à mettre en œuvre font partie de l'expérience et des connaissances des mathématiciens. Cela correspond à un important potentiel de tâches et de savoir-faire à recenser.

1.3. L'ACTIVITE MATHEMATIQUE DANS SA PHASE D'UTILISATION

1.3.1. La gestion de la construction d'une preuve

La phase d'utilisation qui nous intéresse se traduit par la production de preuves et démonstrations. Nous l'abordons essentiellement à travers l'activité purement mathématique s'attachant à la construction de preuves.

Comme nous recherchons une modélisation discrète, les notions que nous utilisons sont définies en terme d'*états* et de ***transitions***, les transitions représentant un changement d'état. Les *ressources de traitement* constituent les moyens disponibles pour la gestion d'une dynamique de changement d'état, et les ***opérations*** sont des ressources qui implantent des transitions. Les états qui nous concernent ici sont par exemple les formules, les énoncés, les problèmes, etc. Nous les qualifions en général de *propositions mathématiques*.

Un *agent* poursuivant un *but* rencontre un *problème* lorsque la *situation* qu'il perçoit l'amène à s'interroger sur les *moyens* de parvenir à son but. Pour qu'il y ait problème, il faut un agent, une situation et un but. La situation, le problème et le but sont modélisés avec des états.

La **résolution du problème par l'agent** consiste à générer des *tâches* permettant de trouver les moyens de parvenir à ce but. Cette notion de tâche permet de s'adapter à des situations ou des buts imparfaitement définis. Elle permet par exemple de ne préciser la tâche que lorsque cela devient indispensable pour la résolution, ou de se contenter de savoir reconnaître si le but est atteint.

Nous qualifions de *raisonnement* une activité mentale dirigée par la réalisation d'une tâche, et d'*inférences* les opérations intellectuelles que l'agent raisonneur utilise. Un raisonnement mathématique se traduit au niveau observable par un enchaînement argumenté de propositions déductibles les unes des autres, et jugées en fonction de leur lien avec les précédentes. Nous réservons le mot *déduction* à une proposition déduite dans un tel enchaînement. Nous qualifions alors de *raisonnement mathématique* l'activité de production de tels enchaînements. Cette définition à l'avantage de permettre de préciser selon le besoin la notion très générale de raisonnement.

Le propre d'une **tâche** est de disposer de facultés de guidage des opérations dont elle dispose ou des sous-tâches qu'elle génère. Un langage de tâches doit permettre notamment l'expression de décomposition hiérarchique, de la notion de synchronisation et de celle de relation logique entre tâches (parallèle, séquence, alternative, etc.).

Une tâche détermine de plus la nature des états et des transitions qu'elle considère. Une tâche d'introduction de formule a pour état des morceaux de formule incomplets, et pour transition leurs opérations de mutation ou d'assemblage de formules. Une tâche d'introduction de notation ou de concept de la théorie a implicitement pour état l'ensemble des connaissances de cette théorie. D'autre part, une tâche de définition de problème peut être associée à une tâche de construction de preuve dont les transitions procèdent soit par équivalence, soit par (in)égalité. Enfin, certaines tâches comme analyser une formule ou substituer un terme dans une expression apparaissent souvent si élémentaires qu'elles ne sont plus explicites.

Alors qu'une tâche est définie par les besoins, les **opérations** sont définies comme des ressources : elles permettent la production de nouveaux états de manière déterministe en fonction de leurs paramètres et de l'état courant. Certaines tâches simples peuvent ainsi être opérationnalisées. Une tâche peut servir d'explication pour remplacer une séquence d'opérations dans une preuve. Cela est fréquement utilisé dans les démonstrations lors de transitions décrites "par application d'un théorème".

Cette opérationnalisation des tâches signifie que le mathématicien dispose d'*opérations décomposables* de même nature que les tâches, utilisant d'autres ressources. Par exemple "transformer une équation de coordonnées cartésiennes en polaires" applique l'opération de changement de coordonnées puis résout en fonction du rayon. Ce changement de coordonnées effectue de l'analyse de formule (perception), de la génération de symboles et un changement de variable qui se décompose en une substitution et des simplifications... Plus l'apport d'information de la part du mathématicien est faible, et plus les opérations requises sont nombreuses. Néanmoins, le fait de disposer d'opérations décomposables permet de gérer ainsi la plupart des paramètres en évitant au mathématicien des actions systématiques et fastidieuses.

Le **raisonnement mathématique** est classiquement partitionné selon la nature des liens entre les propositions, bien qu'étant relatif au système dans lequel il s'insère. Un raisonnement peut ainsi être déductif, inductif, abductif, par analogie, etc. D'après notre terminologie, utiliser une opération déductive

(ou *règle de déduction*) n'est qu'une des façons possibles pour produire une déduction (ç.-à-d. une proposition).

Les états et opérations disponibles étant nombreux, un guidage ferme est nécessaire. Il peut être défini dans sa globalité par un algorithme, par un guidage global plus souple sous forme de *plans* (organisation des tâches à aborder) ou par des principes de décision locaux valables pour quelques situations (*tactiques*) ou pour leur totalité (*stratégies*). Nous employons le terme de *démarche* lorsqu'un plan se réduit à une succession de tâches.

Un exemple de raisonnement mathématique est l'activité de résolution d'équation. Elle est algorithmique pour les équations du second degré, et s'exprime plus facilement à l'aide de plans pour des équations différentielles. Elle utilise des tactiques comme des factorisations ou des racines évidentes lors d'équations de degré supérieur, et utilise des stratégies valables aussi pour des équations plus générales (comme la réduction des occurences de la variable [Bundy 83]).

Ces modes de guidage du raisonnement ne sont pas exclusifs, ni pour la description, ni pour la mise en œuvre. La qualité du guidage décroît rapidement avec de petites variations des problèmes mathématiques, rendant ardues la définition et l'organisation de tâches pas trop restreintes. Une solution consiste en une **forte adaptabilité aux données traitées.**

Nous qualifions de *conduite du raisonnement* les tâches qui choisissent et orientent ce raisonnement mathématique. Alors que le raisonnement mathématique produit des propositions, la conduite du raisonnement engendre des tâches ou déclenche des opérations. Lorsque l'adaptabilité aux données à traiter est forte, la conduite de raisonnement est *opportuniste*.

Nous considérons que vu les aspects créatifs des mathématiques, même une modélisation "complète" et sophistiquée s'enlise sous les points d'embarras dans la mesure où elle vise un domaine non restreint et non algorithmique. En revanche, nous pensons réduire les points d'embarras à un seuil raisonnable pourvu que cette modélisation se limite prioritairement aux tâches élémentaires. L'enjeu est alors de fournir au mathématicien les moyens de guidage appropriés en s'adaptant étroitement au déroulement de son activité.

La **construction d'une preuve** est parfois comparée à une balade en montagne. Si l'objectif retenu est d'aller au sommet, les indications connues ou recueillies permettent d'élaborer un plan qui dépend de la difficulté des étapes et de la qualité des informations disponibles. Les grandes lignes sont précisées, certaines zones de passage choisies, et des tâches sont prédéfinies en fonction des difficultés anticipées. Cette vision générale est du ressort du mathématicien.

Lors de l'exécution effective, ce plan est dynamiquement remanié. Les données sont plus précises et dépendent de la position (état) courante. Des tâches immédiates sont générées : elles sont plus précises et plus élémentaires. Leur lien avec la planification initiale est souvent ténu. Ces tâches concernent notre système. Elles sont décomposées ou enchainées en fonction des opérations disponibles. Elles se traduisent par des productions langagières.

Par analogie, la phase préliminaire d'un projet architectural se traduit par des esquisses jusqu'à une première représentation dite « base graphique » qui est « assez précise pour pouvoir exprimer [illegible]blème et assez imprécise pour permettre des déformations qui ne remettent pas son existence en [illegible]stion » [Quintrand 85]. Il y a ainsi une finition de construction différente pour l'architecte et pour le plan détaillé fourni à l'artisan, bien qu'ils s'insèrent dans un cycle de développement commun.

La base graphique est comparable aux grandes lignes d'une démonstration à venir. L'idée clé est donc de supporter un jeu d'opérations suffisamment lâches pour gérer l'indétermination de certaines caractéristiques des objets mathématiques construits. Nous introduisons la notion de ***preuve à granularité variable*** pour décrire la multiplicité de finition d'une preuve. Une description sommaire est utile pour la compréhension, tandis qu'un niveau de détail important permet la vérification. De même la modélisation d'énoncés et de raisonnement en Langage Naturel doit pouvoir être détaillée si besoin. Cela a donné lieu à la notion de raisonnement à profondeur variable [Kayser 88].

Une preuve à granularité variable est modélisée par un graphe d'états reliés par des opérations ou des tâches de transition. Elle peut être incomplète, valide... Nous parlons aussi d'*étapes* et de *transformation* pour décrire respectivement les états et transitions choisis dans une preuve. Nous appelons ***parcours*** d'une preuve, la succession d'étapes suivie par le mathématicien. La granularité provient de la faculté de remplacer un ensemble d'opérations par une opération ou une tâche, ou réciproquement. Les opérations ou les tâches peuvent ainsi annoter simultanément une preuve à des niveaux d'abstraction différents. Cela est essentiel pour générer une démonstration à partir d'une preuve en résumant les transitions par des

indications sur les tâches ou opérations employées. Toutefois, les tâches ne sont présentes que lorsqu'elles sont utiles pour (ou déduites de) la construction de la preuve.

Cette section a permis d'introduire les concepts clés de la construction d'une preuve. Une *preuve* est un graphe d'*états* reliés par des *transitions*. Ces transitions sont effectuées grâce à des *opérations* afin de réaliser une *tâche*. Cette tâche a pour objectif de résoudre un *problème* dont la nature est très variée. Enfin, la preuve est *à granularité variable* et ses constituants peuvent être détaillés ou abstraits.

1.3.2. Un modèle interactif de résolution de problèmes

Pour modéliser l'activité de raisonnement nous avons le choix entre :
- l'approche IA
- l'approche par tâches.

L'approche IA présente l'activité de raisonnement comme une activité de résolution de problème qu'il faut résoudre. L'approche IA traditionnelle considère en effet un problème bien formulé dans un espace d'états défini qu'il s'agit d'explorer, et généralise notamment les déductions de la logique. De nombreuses variations sont possibles sur ce modèle, comme un espace d'état non explicite ou une caractérisation indirecte des solutions. Les travaux remontent au début des années 60 avec la Démonstration Automatique de Théorèmes mathématiques (par ex. [Wang 70]) et les réflexions sur une résolution de problèmes universelle avec le "General Problem Solver" [Newel 72]. La complexité des techniques employées indique que les difficultés sont importantes.

L'enseignement et l'apprentissage de la résolution de problèmes est devenue une priorité de la formation en mathématiques aux USA. Elle est donc largement étudiée et a donné lieu à de multiples ouvrages [Schoenfeld 85, Silver 85, Janvier 87]. Alan H. Schoenfeld distingue ainsi (p. 15) quatre catégories de connaissances et de comportements nécessaires pour une caractérisation adéquate de la résolution de problème en mathématiques : les ressources dont dispose le mathématicien, les heuristiques, les procédés de contrôle et les systèmes de représentation.

L'approche IA se focalise sur la recherche de solutions, et ne s'intéresse pas a priori au cycle interactif permettant la formulation du problème, ni au choix d'une telle représentation. De plus, la notion de problème ainsi introduite est statique et n'autorise pas toujours une indétermination suffisante. Nous avons conçu une approche par tâches pour satisfaire nos besoins de résolution interactive. L'**approche par tâches de la résolution de problèmes** se décompose en trois phases, dont les deux extrêmes sont dites **interactives** :

formulation
recherche
présentation.

La **résolution d'un problème** consiste alors pour l'agent à transformer une représentation personnelle du monde et du problème rencontré en une représentation opératoire qui permette de le résoudre. La formulation permet cette transformation, et la présentation cherche à l'expliquer ou à la justifier, et à traduire le résultat en terme de buts ou de moyens. Les études didactiques montrent en particulier la difficulté de ces phases interactives.

Pour un mathématicien, la phase de formulation consiste par exemple pour " l'irrationalité de $\sqrt{2}$ ", à remarquer que la diagonale du carré a une longueur inconnue, à chercher à représenter cette longueur comme le rapport de deux nombres "naturels" (les seuls concepts disponibles), pour aboutir enfin à l'énoncé "$p^2 = 2q^2$ avec...". La phase de recherche exploite les propriétés de divisibilité de la théorie des nombres et aboutit à une preuve de l'absurdité de l'énoncé. La phase de présentation produit une explication de ce résultat sous forme d'une démonstration.

L'activité de raisonnement est principalement réservée à la phase de recherche. Chaque phase se décompose en trois points. Comme le montre l'exemple précédent, la **formulation** comprend :
- identification du problème, qui réifie le problème à partir d'un problème perçu, et détermine la représentation et les théories utilisables.
- analyse du problème, qui transforme les éléments du problème en entités définies dans la représentation visée.

- modélisation du problème, qui transforme le problème initial en problème de la représentation visée, avec éventuellement adaptation et simplification. L'adaptation peut consister par exemple à traiter un problème réduit ou un problème plus général.

La phase de **recherche** se compose de :
- identification des stratégies, c'est-à-dire une recherche des méthodes à employer parmi les méthodes disponibles.
- mise en œuvre des stratégies, c'est-à-dire leur contrôle local.
- la solution partiellement construite, qui correspond à l'état d'avancement du problème sans préjuger forcément du contenu ni de l'obtention d'un état final.

La phase de **présentation** se compose de :
- analyse d'une solution, c'est-à-dire une explication à plusieurs niveaux sur les opérations mises en œuvre, les buts poursuivis, et les justifications de leurs succès.
- mécanismes de présentation, qui organisent l'analyse grâce à des concepts et selon une argumentation compréhensibles par l'agent auquel la présentation est destinée
- mise en forme, qui présente le résultat de l'interaction dans le langage choisi.

Cette approche par tâches s'applique au niveau du problème global ou à la construction de sa solution. Un cycle externe de preuves et réfutations bien décrit par Imre Lakatos [Lakatos 84] peut alors critiquer cette démonstration ou modifier le problème. Réciproquement ce modèle peut s'appliquer aux détails de l'élaboration interactive de la résolution d'un problème, par un cycle interne d'interaction entre agents ou entre des représentations distinctes d'un même agent. On assiste alors au contrôle de la résolution par un dialogue entre agents. Notons que la représentation dans laquelle est présentée la solution est généralement différente de celle dans laquelle opère la formulation. Le dialogue à plusieurs agents (ou à un seul internalisé) doit exploiter au moins deux fois le modèle pour obtenir un cycle avec retour à la représentation initiale.

Ce modèle est explicatif et n'a aucune ambition prédictive. Il n'explique pas par exemple l'affinement du problème par conjectures successives jusqu'à la validation d'une solution. Il cherche uniquement à situer la phase de recherche dans son contexte d'interaction. Cette approche par tâches de la résolution de problème par un agent nous servira de guide pour le modèle de l'activité mathématique et le modèle d'un système d'assistance et de son utilisation.

Les principales leçons tirées des travaux sur la résolution de problèmes sont des principes généraux :
- elle est toujours plus complexe que prévue !
- elle fait appel à des mécanismes réflexifs sur son activité (raisonnement sur le raisonnement).
- les connaissances qu'elle fait intervenir sont principalement spécifiques au domaine traité.
- les experts humains utilisent conjointement un raisonnement qualitatif ou approché, par exemple un raisonnement sur l'homogénéité des types ou sur les ordres de grandeur des valeurs.
- l'interaction entre plusieurs agents raisonneurs peut aider grandement la résolution, par exemple via les suggestions mutuelles entre mathématiciens ou lorsqu'il existe une complémentarité naturelle.

Nous en tirons les conséquences suivantes :
- Il faut chercher à exploiter les compétences humaines sans forcément pouvoir les reproduire. Notamment, il ne faut pas chercher outre mesure à les mécaniser. Nous détaillons cela dans notre modélisation ergonomique de la partie II.
- Il faut détailler les aspects observables de l'activité, afin de s'en servir pour communiquer. En particulier, les connaissances utilisées et la description des tâches influant sur les productions du mathématicien sont largement insuffisantes.

Les limites des choix dont nous disposons pour l'activité de raisonnement mathématique sont alors :
- l'approche IA :
  elle est calculable et ne s'intéresse qu'à la phase de recherche. Elle postule que les compétences informatiques sont suffisantes et cherche à les exploiter. Elle est valide lorsque le problème et son contexte sont bien définis, ce qui implique notamment que toutes les informations nécessaires à la résolution soient disponibles. Cela est déjà difficile au niveau des informations de base car la détermination des besoins, leur choix parmi une modélisation très complète, l'identification de lacunes sont respectivement des problèmes ! Cela est quasiment impraticable pour les stratégies et le contrôle, et cette incomplétude entraîne l'intervention du mathématicien dès que l'espace de recherche devient réaliste...

- l'approche par tâches :
  elle envisage toutes les phases d'interaction. Elle n'est pas opératoire car elle ne décrit pas l'organisation des connaissances des théories utilisées, ainsi que l'enchaînement et le détail des tâches à effectuer. Elle peut néanmoins servir de modèle dans la réalisation d'un système d'IA, en introduisant trois espaces de problèmes. Son intérêt principal est la prise en compte de l'interaction. Dans une optique d'assistance, un modèle qualitatif de l'activité de raisonnement suffit pour la partie dévolue au mathématicien. Par contre, il est nécessaire de disposer d'un modèle détaillé des tâches partagées lors de l'activité, et du dialogue par lequel elles s'articulent. Finalement, un modèle calculatoire selon l'approche IA est utilisé pour mener à bout les tâches dévolues au système d'assistance.

1.3.3. Les dimensions de la construction d'une preuve

Nous présentons le point de vue du mathématicien lors de son activité, relativement aux deux autres acteurs, le langage et les données. Nous utiliserons respectivement les vocables "concepts" et "objets" pour distinguer les données que le mathématicien utilise, des données sur lequel il agit. Notons néanmoins que ces vocables ne s'appliquent que sur les données utilisées de façon consciente.

L'activité du mathématicien consiste à produire de nouveaux objets, puis à les étudier afin de les évaluer. Du point de vue du mathématicien, tout ce passe comme s'il utilisait un modèle en arrière plan afin de produire et nommer des objets qu'il manipule effectivement. Le mathématicien dispose alors d'un micromonde d'objets qu'il peut enrichir à volonté. Il peut alors choisir d'évaluer la validité de certains objets ou de les qualifier relativement à une classification. Dans cette optique, le langage sert à la matérialisation de l'activité du mathématicien.

Nous disposons ainsi d'un modèle des actions "instantanées" du mathématicien, qu'il gère globalement dans un processus de résolution de problème. Le mathématicien :
- dispose d'un ensemble d'objets construits ;
- désigne et assemble ces objets pour en construire de nouveaux ;
- applique des transformations sur ces objets (par ex. applique une définition) ;
- juge et classifie les objets construits (par ex. vrai si n est un entier non nul).

Les mécanismes de jugement raisonnent sur l'interprétation des composants, sur les conséquences d'une application de règle et sur la production de nouveaux objets.

Deux principes se dégagent pour l'adéquation entre langage et données :
- un principe de complétude, qui indique que toute donnée remarquable (par construction, par fréquence d'utilisation, etc.) dispose d'un équivalent langagier pour sa désignation.
- un principe de régularité, qui cherche à avoir des notations sur lesquelles s'appliquent des règles de transformation d'objets aussi homogènes que possible.

Nous abordons maintenant le point de vue réciproque des acteurs "langage" et "données". Ce point de vue est découpé en deux univers, selon que l'on considère l'aspect concret (le langage) ou abstrait (les données).

Nous distinguons **trois axes lors de la construction d'une preuve** (figure 1.5) :
① l'utilisation des objets et des concepts de la théorie utilisée ;
② la construction effective de la preuve ;
③ les mécanismes de jugement et de gestion des tâches de résolution de problème.

Ce modèle nous permet de qualifier les productions du mathématicien en six dimensions (trois axes découpés chacuns en deux univers : langage et données). Chacune des six dimensions est gérée en propre par le mathématicien, même si le couplage est de mise. Nous les présentons ici successivement par axe, deux par deux.

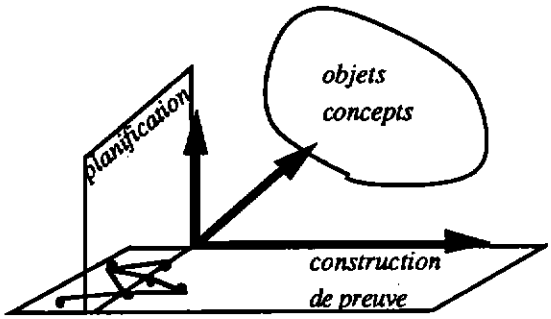


figure 1.5

L'utilisation des objets et des concepts revient à choisir un procédé de désignation pour chacun, tout en bénéficiant des relations qu'ils vérifient. Les procédés de désignation direct utilisent des noms ou des notations s'ils existent, ou des procédés de construction langagiers sinon. Cette dimension langagière introduite, il nous reste à préciser les données.

L'usage et la pérennité des relations entre données sont différentes selon qu'il s'agit d'objets ou de concepts. Par exemple, les objets obéissent à des règles comme "0 + x = x", et sont définis par un ensemble de propriétés mutuelles comme "x et u sont des variables dépendantes", "+ est commutative" ou "la définition de f est...". Ces relations sont liées à l'utilisation.

Les concepts sont utiles pour la constitution du modèle. Nous parlerons ainsi de concept de combinaison pour l'opération mathématique interprétant la règle modus ponens. De même, un domaine ou autre sous-ensemble de l'univers mathématique sera constitué d'un réseau de concepts, regroupant des abstractions telles qu'un quadrilatère, un trapèze, un carré. Enfin, des procédés de désignation et construction permettent d'introduire d'autres concepts comme un carré de longueur 1 ayant telle orientation...

Les dimensions du deuxième axe sont la preuve et la démonstration. La construction de la preuve s'effectue en plusieurs niveaux de structures. Elle utilise les objets et concepts précédents afin de produire des formules et des preuves. Cela permet une validation de la preuve relativement aux concepts de la théorie. On peut ainsi prouver que deux équations sont identiques moyennant un changement de coordonnées, grâce à la construction du lien entre ces deux équations. Cette construction de preuve s'accompagne d'une formulation et d'une présentation à l'aide d'un langage, pour constituer une démonstration.

Les deux axes de construction précédents sont supervisés par des mécanismes de jugement et de gestion des tâches de résolution de problème. La gestion de ce troisième axe constitue l'essentiel de l'exploitation des facultés créatives du mathématicien. La dimension abstraite est prépondérante. En effet, la partie langagière présente dans les démonstrations est couramment réduite à des énoncés de jugement et à des indications.

2. INTRODUCTION A LA MODELISATION DU LANGAGE

2.1. REGARDS SUR LES OBJETS MATHEMATIQUES COURANTS

2.1.1. Deux usages de l'informatique

L'informatique peut être utilisée comme science ou comme outil de traitement de l'information. Tout au long de notre étude du langage, nous utiliserons ce que nous appelons :

Le paradigme informatique d'analyse du langage

Ce paradigme permet d'introduire l'adéquation aux données et au processeur de traitement dans toute étude du langage. Il consiste à comparer les langages naturels et les connaissances qu'ils véhiculent avec le traitement de l'information par un ordinateur. Cette comparaison reste conceptuelle : elle fournit un paradigme d'analyse et non un modèle du traitement du langage par l'homme. L'analogie reste néanmoins fructueuse.

Ce paradigme informatique d'analyse du langage est notamment utilisé par Daniel Lacombe dans son analyse du langage mathématique [Lacombe 84]. Par exemple, ce paradigme explique comment le mathématicien peut travailler en ignorant (en général) la théorie formelle que le logicien a identifiée. Il suffit pour cela de mentionner que le programmeur ignore en général la structure exacte de l'ordinateur dans lequel son programme sera mis ainsi que le langage-machine correspondant.

Nous prendrons soin de distinguer ce paradigme d'analyse, de notre conviction profonde sur l'intérêt épistémologique majeur de l'informatique comme **assistant au travail scientifique**. Ce statut d'assistant surpasse celui d'outil par sa prise en compte conceptuelle tant pour la production de résultats que lors de l'élaboration des moyens et méthodes d'investigation. Nous désignerons cette avancée épistémologique par :

L'assistance informatique au travail scientifique

2.1.2. Rappels classiques sur la définition d'un langage

La figure 1.6 présente selon l'approche logique divers langages développés lorsque les mathématiques forment l'objet de l'étude. Dans chacun d'eux, il y a des objets *atomiques* (relativement au langage, c.-à-d. des "mots") et des objets *composés* constitués d'assemblages d'objets atomiques ("phrases"). Cela constitue la structure du langage définissant tous les constituants membres du langage.

Rappelons que les abus ou confusions se trouvent usuellement au niveau du métalangage. C'est que ce dernier ne peut se contenter de la description informelle du domaine syntaxique "strict". Il a nécessairement une vue plus large incluant les notions lexicales ("lettres") et les interprétations des symboles (par exemple "l'objet sinus"). L'activité du mathématicien se situant à ce niveau, son assistance inclut la description de ces aspects ainsi que celle des théories.

exemples de langages pour les mathématiques

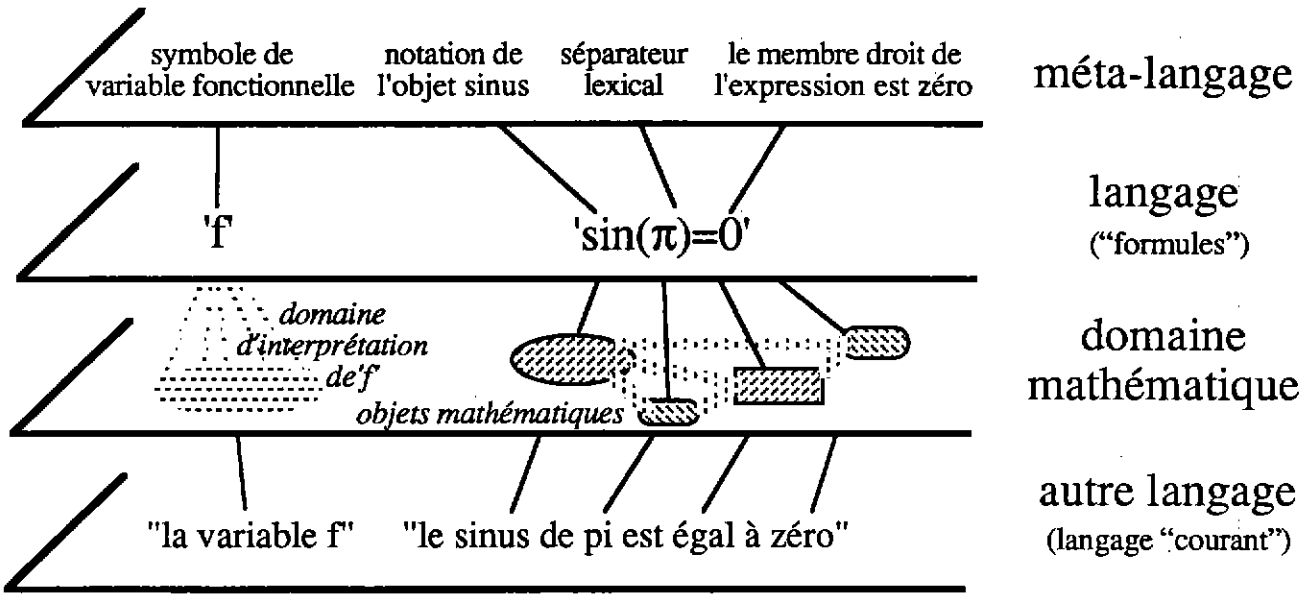


figure 1.6

La terminologie classique définit un symbole par sa portée, ses occurrences, et ses propriétés qui caractérisent l'usage qui en est fait. Dans l'exemple "$\forall x, [\exists y, (x+y=e \wedge y+x=e)]$" le symbole de variable "x" est *lié* et sa *portée* est matérialisée ici par les crochets, ou plus généralement par le *niveau de structure syntaxique* correspondant à son introduction.

Ce symbole "x" est un artifice syntaxique permettant d'indiquer que toutes ses apparitions, dites *occurrences de "x"*, ont un même objet pour ***interprétation*** à l'intérieur de la portée, et n'ont rien à voir avec les "x" utilisés par ailleurs. Si l'on représente la formule par un graphe, les occurrences correspondent à un unique nœud partagé.

Le symbole "e" est selon sa *déclaration*, un symbole de variable *libre* ou un symbole de *constante*. Un symbole de constante est un symbole dont l'interprétation a été fixée a priori lors de la définition de la théorie alors qu'un symbole de variable libre est interprétable (et donc substituable) par n'importe quoi.

Les symboles "+", "$\wedge$" et "=" sont ici des symboles constants : ce sont des symboles de fonctions avec notamment pour propriété un *type* et une *priorité* ici différents, mais une *arité* "deux" et une *syntaxe infixée* commune. Ces propriétés syntaxiques courantes en mathématiques se retrouvent dans les systèmes logiques.

Notons enfin que selon l'emplacement, certains symboles d'une sous-formule (comme celle entre crochets, ou celle réduite à "y") sont libres ou non. Ces contraintes provenant de la construction englobante constituent le *contexte de la sous-formule*.

La ***syntaxe concrète*** d'une formule est l'assemblage de signes qui la constitue. La partie du formalisme support qui ne porte pas d'information est appelée "*sucre syntaxique*". La ***syntaxe abstraite*** est une représentation canonique des relations entre symboles, codée indépendamment du support. Une *grammaire* est un système formel permettant de générer les formules appartenant au langage.

2.1.3. Plusieurs sortes de représentation pour les formules

Cette section :
- développe à partir d'exemples les **représentations informatiques** nécessaires aux manipulations de formules mathématiques ;
- introduit ce faisant les notions de **codage et de traitement de l'information** ;

La chaîne de caractères :

"2(ax+b)".....(1)

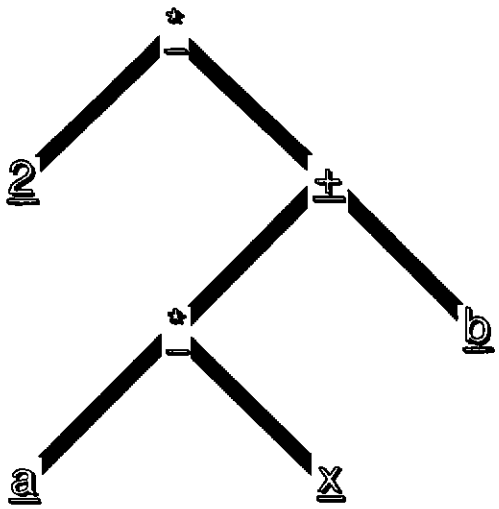
"titille" immédiatement l'esprit du mathématicien, pourtant elle reste désespérément muette pour l'ordinateur. Ce dernier la considère de la même façon que sa consœur "0∫,)(sΣin". C'est que nous avons automatiquement perçu (1), non pas comme une chaîne mais comme une **expression mathématique**.

Nous avons donc lu en substance :

"le produit du nombre 2 par la somme du produit de a par x et de b".....(2)

"le produit externe du nombre 2 par le polynome ax+b".....(2bi)

Dans une représentation calculatoire, la structure de l'expression est décomposée comme l'est (2). L'opérateur est un nœud qui a comme fils ses opérandes. Cette structure arborescente constitue un codage informatique usité. Ainsi la chaîne (1) est représentée selon la forme (3) ci-après :



forme arborescente "calculatoire" .....(3)

Ce résultat (3) dissimule habilement la façon dont il a été produit : il provient en général d'une analyse approfondie de la chaîne originale (1). Les langages "artificiels" sont définis de façon à ce que cette analyse soit syntaxiquement aisée, afin de simplifier la production de la forme "calculatoire" (3) à partir de (1).

La forme (1) représente sa notation mathématique usuelle, sa notation polonaise inverse est :

"2 a x * b + *".....(4)

et la notation parenthésée préfixée est :

"(* 2(+(* a x)b))".....(5)

Les systèmes de manipulation symbolique, tels MACSYMA, utilisent une représentation arborescente proche de (3). Ainsi, pour dériver une expression, ils appliquent une fonction de dérivation selon le type de l'opérateur.

Notons :

(dérivée,x,2(ax+b))..... (1')

la dérivée par rapport à x de (3). La forme externe usuelle de (1') est souvent la chaîne de caractères : " 2(ax+b)' ", ce qui permet un allégement de notation appréciable au prix d'une ambiguïté... Les régles de dérivation d'un produit de fonctions donnent :

(((dérivée,x,2)*(ax+b))+(2*(dérivée,x,(ax+b))))..... (2')

soit après les développements successifs :

((0*((a*x)+b))+(2*(((0*x)+(a*1))+0)))..... (3')

On comprend alors que la partie chargée de la simplification soit importante...

La forme (3) est donc adaptée aux calculs, tandis que la forme (1) : **"2(ax+b)"** facilite la compréhension et constitue le support des activités mathématiques humaines, nous concluons alors :

Le codage des formules dépend de l'usage.

Une *information* est une donnée externe à un *processeur de traitement*. Un processeur suffisamment renseigné sur la structure d'une information est capable de l'exploiter afin de l'intégrer parmi ses *capacités mémorielles* de traitement. Une information ainsi intériorisée est qualifiée de *connaissance*.

Les informations destinées à être transmises entre plusieurs processeurs sont codées en une structure particulière afin d'être facilement exploitées. Chaque processeur doit simplement connaître le point de vue de la structure pertinent pour le traitement qui l'intéresse. Les fonctions de codage et décodage de l'information sont alors les seules connaissances indispensables au processeur.

Vu la faible quantité de connaissances disponibles en regard des informations envisageables, les connaissances de longue durée se doivent d'être en grande partie *génériques*, c'est-à-dire exploitables dans plusieurs situations. Une forme de généricité figée à un certain type d'information consiste à connaître simplement les fonctions de décodage, traitement et codage de l'information.

Le langage mathématique fournit au mathématicien un support dont l'information utile (et utilisée) présente des **aspects très différents** (sémantique, syntaxe, considérations pratiques comme le positionnement, ou considérations pragmatiques liées à l'usage, etc.). Bien que proches à bien d'égards des langages artificiels, les formules sont issues d'une évolution historique multi-domaines, façonnées et adaptées aux besoins des mathématiciens, et non édictées en Norme pour des finalités informatiques. Les formules sont "naturelles" car les informations qu'elles véhiculent sont empiriquement adaptées aux critères cognitifs.

Pour un langage naturel, il faut effectuer l'abstraction non pas d'une unique information, mais d'un **faisceau d'informations** convergentes. Cette focalisation sémantique introduit une redondance d'information dont le rôle de certains aspects reste secondaire mais utile au bon déroulement de l'activité - ici mathématique -. En mathématiques, un traitement principal d'ordre syntaxique ne suffit pas. Comme plusieurs aspects se conjuguent pour donner un sens, il est bon d'en expliciter les diverses structures. Par analogie le traitement de l'information afin de reproduire la vision des couleurs est beaucoup plus qu'une simple analyse spectrale. Cette analyse est simple mais insuffisante aux traitements ultérieurs en vue de la formation des "concepts" de couleur.

Comme c'est le cas dans tout langage naturel destiné à un processeur humain, l'étape intermédiaire entre le support et la prise en compte de l'information est approximativement formalisée. L'exemple (1) : **"2(ax+b)"** et ses développements ont permis de souligner quatre points importants :

① chaque *fonction de traitement* (manipulation, composition...), traite une information qui lui est propre à l'aide d'une représentation concrète appropriée ;
② les règles qui analysent ou composent, font appel au sens mathématique des objets du message, et donc **l'information textuelle ne suffit pas** ;
③ la représentation calculatoire que manipulent les règles de calcul utilisées par le mathématicien, est épurée des parenthèses et autres constituants syntaxiques ;
④ la notation externe est peu adaptée aux manipulations, mais utile comme support visuel du mathématicien ; sa correspondance avec la représentation calculatoire doit être préservée.

C'est pourquoi il est utile de **découpler et représenter** les divers aspects de l'information utilisée par le mathématicien et de répartir pour chaque utilitaire de traitement (toute) l'information qu'il est à même de traiter.

2.1.4. Terminologie et représentation du langage

La redondance des concepts et l'usage de plusieurs modes d'expression font que les possibilités de **paraphrase** sont souvent nombreuses en mathématiques. Toutefois, ces possibilités restent inexploitables sans la définition et la modélisation de la terminologie, c'est-à-dire des mots du langage. La terminologie permet en effet, de former par composition la dénomination textuelle de toute relation ou action mathématique.

Une terminologie spécialisée est employée en mathématiques pour désigner des entités spécifiques. Le rôle de la terminologie est de désigner (faire référence à) des constituants qui interviennent dans la production du sens. Si le sens est défini par l'usage, "le changement de variable précédent" fait référence à une opération tandis que le sens de cette opération inclut notamment son utilisation dans des situations effectives comme "effectuer le changement de variable précédent".

A noter qu'il n'y a pas de terminologie particulière pour désigner des expressions incomplètes ou sans signification comme "sin(", "sin(vrai)" ou "0∫,)(s∑in". Une ***formule*** est alors une expression syntaxiquement bien formée pour laquelle il existe une interprétation lui attribuant un sens.

La terminologie utilisée pour les **constituants syntaxiques** concerne :
- la *forme* ou la position, comme terme indicé, terme parenthésé, terme de gauche, exposant, monôme, etc ;
- le *rôle* relatif à l'opérateur utilisé, comme borne supérieure, élément différentiel, terme principal, numérateur, variable d'index, etc ;
- la *fonction* mathématique, comme variable, fonction, théorème, etc ;
- la *position* dans la structure, comme symbole terminal, variable libre ou liée, etc.

La terminologie courante relative aux **objets et concepts** les distingue selon :
- leur *nature*, comme variable, paramètre, constante, théorème, propriété, proposition, etc ;
- leurs *relations respectives*, comme variables dépendantes ou indépendantes, concept plus spécifique, etc.

Enfin, il existe une terminologie spécialisée à l'**utilisation dynamique** :
- les *manipulations* syntaxiques, comme substitution, remplacement, etc ;
- l'*introduction* d'une entité comme une variable, une notation, une définition de concept, etc ;
- les concepts de *manipulation*, comme changement de variable, application d'un théorème, déduction, etc ;
- les concepts *relatifs à la résolution*, comme absurde, hypothèse, cas, résultat, etc ;
- les conseils, enchainements et autres méta-commentaires, qui relèvent d'expressions "consacrées" plutôt que d'une terminologie particulière.

Une telle **classification de la terminologie** est souvent implicite, voire confuse et n'est pas répertoriée à notre connaissance. Elle fait partie des multiples connaissances empiriques utilisées en mathématiques.

La terminologie sert aussi de support à l'analyse textuelle, les mots reliés sémantiquement le sont naturellement dans la représentation des objets qu'ils désignent. Par exemple, au concept d'intégrale sont reliés ceux de bornes, fonction, théorème de la primitive, intégration par parties etc. Ces mots ou assemblages de mots seront recherchés prioritairement lors d'une analyse dès l'identification de la racine "intégr" (et sa différenciation avec d'autres mots comme "intégrité").

Le mathématicien évolue dans le monde mathématique bien plus souvent que dans celui des symboles qui s'y réfèrent. Une représentation à caractère textuel sur l'assemblage des symboles, n'est qu'une étape : l'ordinateur doit avoir une **représentation des objets mathématiques** eux-mêmes. Dans la formule "sin(x)", le symbole de fonction "sin" n'est exploitable que s'il est relié à la fonction sinus, à ses propriétés, ses relations...

Même d'un point de vue linguistique, il est difficile de comprendre et représenter une phrase comme :

> L'ensemble des complexes peut être défini comme l'anneau quotient de l'anneau des polynomes en x à coefficients réels relativement à l'idéal principal engendré par x^2+1.

Pour ce qui concerne sa représentation, nous considérons par exemple que "peut être défini" signifie qu'il existe une telle interprétation des nombres complexes. Le mathématicien peut alors à tout moment choisir explicitement cette interprétation pour continuer à travailler.

Un choix de représentation du langage, et des activités qu'il autorise, requiert un panorama très diversifié de connaissances : notations, théorèmes, démonstrations, règles de présentation, stratégies de résolution... Ces connaissances doivent comporter plusieurs facettes : structurelle, opératoire, et syntaxique. Pour comprendre un texte, c'est-à-dire pouvoir décrire sa structure, son sens et répondre à des questions le concernant, un dictionnaire, fut-il encyclopédique ne suffit pas. Il faut avoir connaissance du monde "réel", tout comme du langage lui-même. Ainsi, un élève ne sait pas forcément quand se servir d'une règle, et comment bien le faire !

L'ordinateur va donc aussi utiliser dans ses tâches des connaissances opératoires sur le domaine et le langage. Cette tâche de **modélisation du domaine** constitue à notre avis la difficulté majeure des mathématiques. Rappelons en effet que la traduction automatique des langues naturelles a considérablement restreint ses ambitions faute de structures de compréhension adéquates. Il faut donc modéliser le domaine de discours dans ses aspects les plus élémentaires avant d'espérer un résultat conséquent pour l'activité mathématique. Dans ce travail, nous nous contenterons de décrire certains aspects du domaine et de prendre en compte la terminologie lors d'une approche plus modeste focalisée sur les manipulations de formules.

2.2. LA FORMALISATION LOGIQUE DU LANGAGE

2.2.1. L'approche logique

L'étude de la logique, malgré son abord statique et microscopique des théories, apporte une contribution indispensable à l'étude des mathématiques usuelles. Elle est l'aboutissement de la quête des fondements en mathématiques et est à l'origine de la formalisation de la calculabilité.

Outre la possibilité de formaliser une théorie mathématique et son usage, la logique dite "formelle" fournit une **méthode d'analyse du Langage Mathématique**. Elle s'inscrit alors comme la première étape d'une étude du Langage Mathématique qui s'intéresse aux relations entre la forme et le sens.

Cette section ne cherche donc pas à définir un formalisme logique supplémentaire pour représenter la structure des preuves et des déductions mathématiques usuelles. Elle montre l'importance que joue le langage comme instrument de déduction et elle se conclue sur les limites de la logique pour modéliser l'activité mathématique.

L'approche logique représentée figure 1.7 étudie les liens d'interprétation entre un domaine syntaxique construit avec des symboles, et un *domaine d'étude*, ici un domaine mathématique. Un *langage formel* est défini comme un ensemble de formules bâties à l'aide de symboles du domaine syntaxique. On nomme *"interprétation"* l'opération qui consiste à associer aux symboles du langage formel des objets (et des relations entre objets) issus du domaine d'étude. Le résultat de cette opération est un *Modèle* du domaine d'étude (nous distinguons ce sens logique du sens courant par une majuscule).

représentation de "l'approche logique"

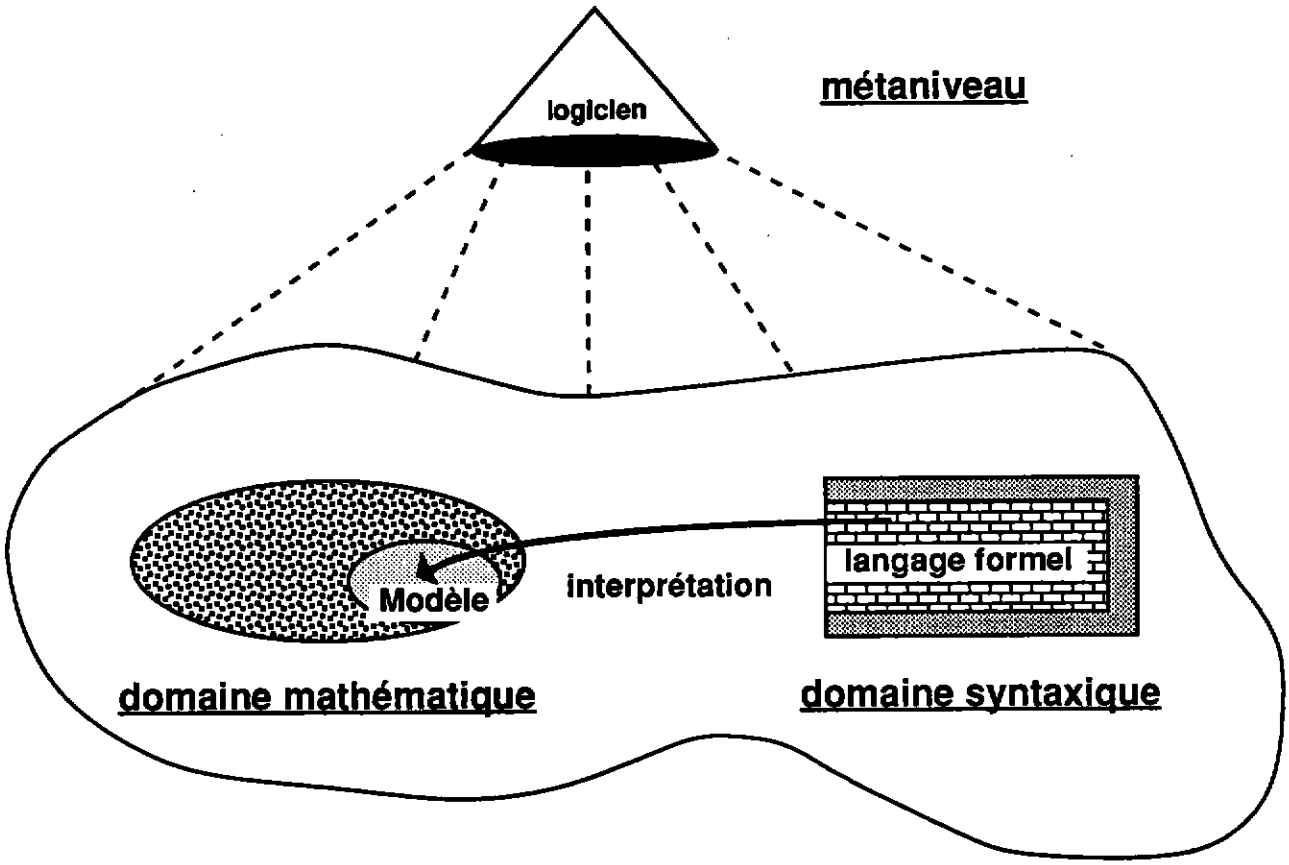


figure 1.7

L'approche logique décrit les structures respectives de ces deux zones d'intérêt, et manipule les liens d'interprétation qui les relie. Le "logicien" est ainsi un observateur de ce système et les descriptions, déductions et commentaires sur ce système lui sont adressées au *métaniveau* dans un *métalangage*.

Une fois ajoutées les caractéristiques opératoires, cette approche logique introduit donc trois "théories" distinctes :

① la **théorie informelle** (intuitive) du domaine mathématique à modéliser ;
② la **théorie formelle** du langage formel, qui est une théorie syntaxique de manipulations de symboles ;
③ la **métathéorie informelle** du métaniveau dans lequel ce système est décrit et étudié.

Afin de préciser la distinction entre un *domaine syntaxique*, un *langage formel* de ce domaine et la *théorie formelle* batie sur ce langage, nous rappelons ici quelques notions de base de la logique. Des bases plus solides sont amplement disponibles dans la littérature, citons par exemple [Barwise 78, Carnap 58, Klenee 71, Pabion 76, Schoenfield 67].

Le domaine syntaxique peut être un ensemble de symboles graphiques et leurs positions bidimensionnelles pour créer des formules, un ensemble de lettres (alphabet) à ordonner pour former des mots, un ensemble de mots à ordonner pour former des phrases, etc.

Un langage formel est alors un sous-ensemble donné de cet univers dont la définition peut faire intervenir des procédés complexes. Il est d'usage à des fins de manipulation théorique ou calculatoire de doter ce sous-ensemble d'une structure récursive simple à utiliser. La théorie formelle permet alors des manipulations de symboles que le mathématicien interprête comme des déductions.

Les théories formelles se distinguent des théories "naturelles" par l'apparence régulière de leurs structures. Ces structures servent de support à leur définition, aux preuves de la théorie, et aux méta-preuves de propriétés de ces théories.

Cette approche logique permet de décrire les liens entre un langage et son domaine d'application, elle ne nous dit rien sur les procédés de modélisation à employer ni sur la pertinence du point de vue adopté.

Ainsi, en utilisant le concept de courbes de niveau, nous effectuons une correspondance entre une carte blanche rayée de courbes et un Modèle d'une zone d'intérêt de cet univers : la France, sous un point de vue topographique. Des concepts plus complexes interviennent pour enrichir le Modèle - et la carte - d'un point de vue géopolitique.

Dans tous les cas, le Modèle de la France dont nous disposerons ne saura englober la totalité des points de vue. Nous considérons que l'univers mathématique est soumis aux mêmes limitations et que seuls ses aspects "apparents" sont aisément modélisables.

2.2.2. L'interprétation des phrases

Nous disposons maintenant avec l'approche logique d'une **méthode d'analyse du langage**, qui s'applique à l'étude de notre langue. Nous sommes alors théoriquement capables de décrire les niveaux d'interprétation d'un énoncé, à condition de parvenir à modéliser les objets de discours et leur imbrication.

La coexistence de plusieurs niveaux d'interprétation dans le langage est en effet l'un des points forts des langages naturels. Il suffit de considérer par exemple l'expressivité de la phrase : "il suffirait de remplacer le paramètre θ par la valeur choisie".

Ces différents niveaux d'interprétation imbriqués consituent par exemple la force de raisonnement d'un système. Ils permettent la production et l'étude des métathéorèmes : alors qu'un théorème est une expression du système formel, un métathéorème est un expression sur le système formel. Le livre de Jacques Pitrat [Pitrat 70] en recense plusieurs types selon qu'ils indiquent qui sont les théorèmes, comment les dériver, introduisent de nouvelles règles équivalentes...

Lorsque les constructions d'une langue (par exemple le français) servent de support à des langages différents, des confusions sont susceptibles de surgir. C'est le cas lors de l'utilisation du français pour parler de la grammaire française, de l'écriture d'un programme qui traite d'autres programmes, de l'utilisation d'une théorie mathématique pour raisonner sur elle-même. Surviennent alors des "boucles étranges" chères à Escher, que l'on retrouve au cœur des théorèmes de Gödel qui limitent l'universalité des théories mathématiques.

Une application de cette méthode d'analyse permet de détecter que la phrase suivante est ambiguë :

"Les paramètres dimension et taille sont à fournir au programme."

Pour illustrer ainsi l'analyse précédente, le domaine d'étude ① est celui des objets abstraits ou exécutables appelés "programmes". Le langage formel ② est constitué de textes écrits en un langage de programmation. La phrase est un commentaire exprimé au métaniveau dans le métalangage ③. Cette phrase peut alors avoir deux significations :

- Si l'interprétation est celle du métalangage pour tous les mots de la phrase, elle signifie qu'il y a deux paramètres à fournir au programme. Les mots "dimension" et "taille" sont alors deux mots du métalangage faisant référence au contenu de ces paramètres.
- Si les mots "dimension" et "taille" sont des identificateurs du texte du langage de programmation, elle signifie que le programme attend une valeur pour ces deux identificateurs.

Un procédé de mise en valeur de cette ambiguïté consiste à utiliser pour le méta-langage une langue différente de celle du langage formel. Ainsi, un exercice de traduction révèle l'ambiguïté. Dans le premier cas les identificateurs recevant les paramètres n'étaient pas connus, et la phrase exprimée en anglais serait entièrement traduite. Dans le second une traduction (accompagnée d'une levée de l'ambiguïté par "The values of...") donnerait par exemple :

"the parameters «dimension» and «taille» are to be provided to the program."

Une solution permettant de bien distinguer les niveaux de langage est de les expliciter ou de ne pas utiliser la même terminologie. Si tous les identificateurs commencent par "$" ou sont syntaxiquement distingués dans le méta-langage, il n'y a plus d'ambiguïté. Toutefois bien souvent, la connaissance de la tâche du mathématicien permet d'éviter les confusions. Il suffit par exemple que les mots "taille" et "dimension" ne figurent pas conjointement dans le programme considéré, même s'ils figurent isolément et qu'ils interviennent dans d'autres programmes.

Pour jongler avec ces interprétations, lorsqu'une formule ou une phrase sont intégrées à notre langage de discours, leur interprétation diffère et est distinguée par les guillemets : "...". Par défaut l'interprétation vise les concepts mathématiques sous-jacents ; par contre, si l'on traite de la syntaxe, elle vise les symboles ou les phrases : l'interprétation est ainsi contextuelle. Dans notre langage de discours, nous utilisons les guillements : "..." pour une altération du sens courant, «...» pour les citations, tandis que les définitions stipulatives sont notées en italique.

2.2.3. Formaliser les déductions

Une fois défini un langage formel, l'intérêt des théories est de modéliser des déductions. Deux voies sont alors possibles selon l'approche logique (revoir la figure 1). La première, qualifiée de *théorie de la Preuve*, s'intéresse uniquement aux manipulations à l'intérieur du domaine syntaxique. La seconde, qui étudie les domaines d'interprétation possibles des formules du langage formel, est appelée *théorie des Modèles*.

Pour commencer par la **théorie des Modèles**, la notion de R-structure formalise le lien associant la structure d'un langage formel, et sa réalisation dans un domaine d'étude via une interprétation. Modulo une certaine cohérence[1], cette réalisation forme un Modèle du langage. L'interprétation des symboles de variable pose problème car ils peuvent être associés à tout objet "constant" du domaine mathématique. La formule "$\forall x$ f(x)" est vraie dans le Modèle, si toutes les assignations de constantes possibles pour "x" dans "f(x)" le sont. Ce procédé pouvant amener à considérer une infinité non dénombrable de cas, il nécessite des raisonnements sur les "choix" à envisager.

Une formule "ψ" est une *conséquence logique* d'une formule "φ" si tout Modèle de "φ" est un Modèle de "ψ". Deux formules sont *équivalentes* si leurs Modèles sont identiques. Cette notion permet de raisonner sur des formules plus simples ("φ" au lieu de "$\neg\neg\varphi$"...). Ces définitions se généralisent à tout ensemble de formules. Un ensemble de formules est *satisfaisable* si et seulement si il a un Modèle, il est *catégorique* si tous ses Modèles sont isomorphiques.

L'approche logique introduit aussi les concepts sémantiques d'*intension* et d'*extension*. Ils correspondent à une distinction de nature ontologique à l'intérieur même du domaine d'étude, entre les objets et les propriétés énoncables sur ces objets. Lorsque le mathématicien comprend une phrase sans savoir si elle est vraie, il **raisonne au niveau intensionnel** sur le sens de la phrase. Cette propriété est une caractéristique importante du Langage Mathématique et du Langage Naturel.

L'intension de "rond ?" est la propriété d'être sphérique, et son extension est la classe de tous les objets sphériques. L'intension de "sin(0)" est le sinus de zéro et son extension est zéro. Ainsi que montre "sin(0)", "1-1" ou "l'élément neutre de l'addition", il y a une infinité d'intensions associées à même extension. Les liens entre elles font partie de la connaissance que l'on a sur le domaine d'étude : ils sont sujets à déduction lorsqu'ils peuvent être établis.

Si, un langage possède des *modalités* (possible, nécessaire...) qui tiennent compte de l'intension de la formule, il est dit *intensionnel*. Sinon, il est *extensionnel* et deux formules sont équivalentes si et seulement si leurs extensions sont identiques. Des développements spécifiques au Langage Naturel sont disponibles dans "Approche logique de l'IA" [Thayse 89, p62-68 et 101-104].

Lorsque les langages formels deviennent trop expressifs, la structure même des Modèles devient insuffisante et on est obligé d'introduire des ensembles ou des univers de Modèles rendant la logique peu satisfaisante pour représenter les connaissances et certains types de déduction. Les mathématiques ont

---

[1] Chaque symbole de constante, fonction et prédicat est interprété par un objet du domaine d'étude. La fonction d'interprétation "I" appliquée à la formule "f(a,b)" donne "I(f) [I(a),I(b)]".

développé pour leurs propres besoins un moyen incrémental de raisonnement sur des Modèles disjoints, à l'aide d'hypothèses et de retour arrière en cas de contradiction.

Ainsi, bien qu'elle fournisse une méthode d'analyse, l'approche logique ne cherche pas à représenter au mieux les connaissances. Elle offre par contre un modèle calculatoire efficace avec la **théorie de la Preuve.**

Une *théorie* (formelle) n'est rien d'autre que la donnée d'un sous-ensemble (souvent infini) de formules du langage appelées *théorèmes*. Ce sous-ensemble est usuellement défini par son procédé dynamique de construction qui décrit comment à partir de formules distinguées appelées *axiomes*, on utilise des *règles d'inférences* pour obtenir d'autres formules : les théorèmes.

Un *système formel* est la donnée d'un langage formel, d'axiomes et de règles d'inférence. Les *théorèmes* sont définis comme l'ensemble des formules obtenues par application récursive des règles d'inférence sur l'ensemble des axiomes. Cet abord est dynamique car il ne décrit pas des structures en terme de leur composition syntaxique mais par l'existence d'étapes de transition valides à partir d'axiomes. Nous différencions ainsi la **structure du langage** (qui détermine les expressions biens formées du domaine ou formules) de la **structure de la théorie** qui détermine les formules valides[2].

L'intérêt de ces approches distinctes (théorie des Modèles et théorie de la Preuve) est de pouvoir identifier les énoncés vrais d'un Modèle aux théorèmes issus d'un calcul formel. Le **théorème de complétude** de Gödel dit que pour les langages formels du premier ordre, raisonner avec le système formel est équivalent à raisonner sur ses Modèles. Le théorème de compacité dit que si tous ses sous-ensembles finis de formules ont un Modèle, le langage du premier ordre en a un. Malheureusement, ces deux théorèmes sont faux en général pour les théories de second ordre qui représentent l'essentiel de la pratique mathématique.

La **correspondance** entre le domaine d'étude mathématique et le domaine syntaxique a toutefois permis de distinguer et d'associer les concepts suivants :

domaine mathématique	domaine syntaxique
intuition	calcul
objet	terme
énoncé	formule
énoncé vrai	théorème
vérité	validité
une "démonstration"	une dérivation
"preuve mathématique"	preuve formelle

2.2.4. Apport de la logique aux mathématiques

La présentation la plus réussie du rôle historique de la logique dans la recherche des fondements est à notre avis l'article de Daniel Lacombe [Lacombe 88]. Cette logique dite mathématique est mathématique par sa formalisation mais non par ses capacités de modéliser les mathématiques et leur activité. Outre sa contribution à l'étude du langage, la logique nous a permis d'éclairer des aspects fondamentaux de l'activité

---

[2] Rappelons la distinction souvent oubliée entre la notion de formule valide - qui est interprétée comme un énoncé mathématique vrai - et celle de tautologie. Une tautologie est vraie de par sa forme syntaxique : c'est donc une formule valide indépendante de l'interprétation. "$(\forall x\ f(x)) \vee \neg(\forall x\ f(x))$" est une tautologie : cela rejoint la propriété des mathématiques de ne considérer que l'information nécessaire à la conclusion, sans avoir à identifier "$(\forall x\ f(x))$" (qui peut être néanmoins une formule valide). De façon analogue, si l'on introduit deux formules différentes "φ" et "ψ", la façon la plus simple d'identifier leurs interprétations respectives consiste à les transformer en une forme syntaxique identique : "ξ".

mathématique. Elle a montré notamment que les mathématiques sont intrinsèquement incomplètes quant aux informations accessibles et que la pratique intuitive ne peut se réduire à un calcul systématique.

La logique fournit un **instrument de réflexion** épistémologique sur des aspects de l'activité mathématique. Plutôt que d'envisager la logique comme le fondement des Mathématiques, il est plus réaliste de la concevoir comme un outil d'analyse, permettant d'étudier des mécanismes atomiques relativement aux théories mathématiques. Elle permet d'étudier des structures communes à plusieurs domaines mathématiques, d'évaluer la puissance potentielle de ces structures, de discerner les expressions indépendantes ou indécidables, etc. Par contre, **la logique n'est pas** une étude de l'activité mathématique, elle n'est pas une alternative au Langage Mathématique, elle ne s'intéresse ni à l'expressivité, ni au rôle de l'usager et de ses connaissances...

Dans sa pratique, le mathématicien est alors confronté à un dilemne créateur : systématiser les déductions dans un espace où tout est calcul, et se réserver des possibilités d'échappement pour ne pas se borner a priori. La pratique des mathématiques est en général trop complexe pour que l'on puisse exhiber un algorithme "systématique" et de complexité "raisonnable". Le problème ne se pose plus alors en terme de décidabilité. Les mathématiques surmontent alors cette limite en renonçant à la possibilité d'un calcul systématique et (ou) par un abord dynamique de la notion de système qui appréhende l'infinité des possibles par une extensibilité "à la carte". Si l'on suit l'approche logique, le Langage Mathématique est "éclaté" en une infinité de langages de niveaux "méta" différents. Les caractéristiques de ce système devraient-elles se réduire à la combinaison de celles de ses parties ?

La pratique des mathématiques est mieux appréhendée comme l'étude des cohésions entre objets d'une théorie, plutôt que par un modèle cosmique d'une théorie unifiée. Ainsi, que la théorie des ensembles, la théorie des nombres ou la géométrie aient une structure commune ne remet pas en cause les résultats établis sans cela. Elle permet au contraire d'enrichir ces théories d'interprétations plus générales grâce à un point de vue unificateur. Les connaissances et heuristiques développées peuvent ainsi être réutilisées et appliquées à d'autres domaines. Cette capacité intuitive ou raisonnée de **guider les déductions** est la différence fondamentale entre les mathématiques et la logique.

Les inconvénients de cette approche mathématique peuvent en retour expliquer les **motivations d'une approche logique**. La logique a dû formaliser l'outil de traitement pour formaliser les déductions, et a donc cherché le modèle universel le plus simple à formaliser. D'autre part, les mathématiques cherchent plutôt à trouver et expliquer leurs résultats, tandis que la logique a d'abord cherché à justifier les déductions. La notion de contexte est fort répréhensible de ce point de vue, puisqu'elle ne cherche pas à discerner les influences respectives des propriétés utilisées pour la preuve en cours. Cette recherche se retrouve actuellement en mathématiques où l'on cherche à analyser les justifications, notamment à trouver les conditions minimales de validité de la justification. Reformulé en terme de tendances historiques, après Trouver puis Justifier, la communauté mathématique cherche à Minimiser afin d'Atomiser le fondement des mathématique en structures Universelles.

Les mathématiques à venir vont-elles trouver un renouveau sur cette base (si ce n'est déjà fait) ? Ces comparaisons nous montrent en tout cas qu'analyser des arguments mathématiques est difficile avec un formalisme logique car l'expression de ces arguments est souvent aussi impossible à écrire que leur démonstration formelle. Nous suggérons au contraire de dériver cette analyse depuis les justifications produites à partir du langage usuel. D'un point de vue métamathématique, construire un système mathématique SM donné en s'inspirant et en réutilisant les constructions du Langage Mathématique est d'une complexité plus réaliste que de chercher à partir d'une quelconque théorie des fondements. Cette approche à rebours permet malgré tout d'étudier des questions métamathématiques comme prouver qu'un théorème est non démontrable dans SM ou chercher les théories T minimales qui sont mathématiquement équivalentes à SM.

La métamathématique n'est autre qu'une mathématisation de la part de l'activité mathématique qui porte sur le langage et les données initiales dont le mathématicien cherche à explorer les conséquences. Mais cette mathématisation n'éclaire guère les productions réelles du mathématicien, ni d'ailleurs la façon dont il dirige ses activités. Il serait bon maintenant de savoir comment le mathématicien produit ses résultats si convoités, que la force même astucieuse n'arrive pas à résoudre. Il serait surtout profitable de pouvoir mieux assister les capacités de production d'un mathématicien en vue de les appliquer à une demande croissante de modélisation scientifique et technique.

2.3. LIMITES DE LA LOGIQUE POUR MODELISER L'ACTIVITE MATHEMATIQUE

2.3.1. Modélisation du langage et monde de référence

L'analyse traditionnelle du langage le décompose en syntaxe, sémantique et pragmatique :

- la ***syntaxe*** concerne les relations entre les mots d'une phrase, et plus généralement définit les structures bâties sur ces mots jusqu'au niveau du discours ;
- la ***sémantique*** étudie le sens de ces expressions syntaxiques relativement à un Modèle ;
- la ***pragmatique*** fait référence - même indirectement - aux interlocuteurs et à ce qu'ils sont en mesure de connaître sur le sujet du discours. Elle est intrinsèquement liée à l'action.

La notion de pragmatique récupère ainsi les thèmes "difficiles" rejetés hors de l'approche logique. Par exemple, quel texte peut être qualifié de démonstration ? Quelles lois régissent le choix des symboles et leur homogénéité ? La pragmatique inclut notamment ce qui touche à l'organisation plus générale du discours, à commencer par son déroulement.

La problématique de traitement du Langage Naturel se rapproche de celle du Langage Mathématique car elle cherche à modéliser une activité humaine et ses phénomènes complexes. Les deux tomes de Gérard Sabah [Sabah 89] constituent la référence actuelle : ils présentent un panorama complet et détaillé des problèmes, techniques et systèmes de traitement. Ils constituent un approfondissement souhaitable de nombreux thèmes abordés ou éludés dans nos propos. **Malheureusement, ce livre de référence n'aborde à aucun moment le Langage Mathématique.** La problématique des mathématiques reste une fois encore ignorée de la problématique linguistique.

L'analyse traditionnelle du langage, que Gérard Sabah montre comment dépasser, est insuffisante car elle considère un univers trop schématique. L'interprétation sémantique ne fait alors qu'une épuration syntaxique de l'information. Une interprétation sémantique plus profonde consisterait à enkyster cette information au sein d'un système de représentation plus complet, mais serait alors dépendant d'un mode de traitement et d'un état des connaissances non universel.

On peut donc se cantonner comme dans l'approche logique à l'entrée qui va déclencher le traitement du sens, ou essayer d'inclure cette entrée dans une représentation plus générale du sujet du discours. Pour les formules, cette entrée est caractérisée par leur représentation en syntaxe abstraite, puisque c'est elle qui est manipulée par les outils informatiques de raisonnement.

La difficulté d'exhiber une sémantique pour les mathématiques vient que cette notion de sémantique est peu adaptée en dehors des mondes formels. Ainsi, le Langage Naturel est doté de multiples sémantiques selon l'usage qui en est fait. André Martinet généralise par exemple la syntaxe à « ce qui permet de résoudre l'antinomie permanente entre la globalité de l'expérience et la linéarité du discours » [Martinet 85]. Cette affirmation pose le problème en reliant le langage à son système de traitement : il ne reste qu'à en proposer des modèles et expliciter les hypothèses concernant le mode de traitement !

Dans son acceptation courante, on appelle *modèle* toute structure formalisée utilisée pour rendre compte d'un ensemble de phénomènes qui possèdent entre eux certaines relations. La ***modélisation*** désigne ainsi le processus de constitution d'un modèle. Un modèle est choisi en fonction de notre analyse de ces phénomènes. Il peut être :

- *descriptif* s'il se contente de les décrire ;
- *explicatif* s'il les analyse en fonction de causes non observables ;
- *normatif* si la structure formalisée est imposée a priori ;
- *évaluatif* s'il juge les phénomènes selon une finalité (comparative, diagnostique, etc) ;
- *prédictif* s'il permet d'en anticiper l'évolution par une *simulation*.

L'*expérimentation* est la confrontation des résultats du modèle avec ceux issus de la réalité de son utilisateur : c'est un cas particulier d'évaluation. L'*évaluation* du modèle permet la validation ou la vérification du modèle relativement à un ensemble de phénomènes. La validation porte sur le monde intensionnel des idées alors que la vérification concerne le monde extensionnel des faits.

La *validation* du modèle est liée à la notion d'**utilité**. Elle définit l'intérêt du modèle pour son utilisateur, et s'effectue souvent à l'aide d'une grille d'évaluation (pertinence, c'est-à-dire ici évaluation de l'approximation). La *vérification* du modèle est liée à la notion de **vérité**. Elle compare les productions considérées vraies dans le modèle avec les productions que l'on considère justifiées par les phénomènes à modéliser. Elle s'intéresse aux insuffisances du modèle, soit dans sa production de résultats incorrects

(pertinence, c'est-à-dire ici contradiction), soit dans son insuffisance à trouver des résultats vrais (incomplétude).

Le paradigme informatique permet de cerner la notion de sémantique des mathématiques. La donnée d'une sémantique aux programmes logiques comme ceux de PROLOG revient à manipuler un Modèle des relations vraies parmi toutes les relations exprimables. Une sémantique des mathématiques est alors l'existence d'un Modèle cohérent qui satisfasse toutes les relations introduites. L'article de S. Shapiro [Shapiro 85] soulève les difficultés théoriques de l'existence d'un tel Modèle des mathématiques.

Il n'existe pas à notre connaissance de sémantique bien formalisée correspondant à l'activité mathématique.

Pour cela, les mathématiques rejoignent les langages de programmation dont la sémantique est définie en fonction des opérations de traitement. Si nous arrivons à caractériser les opérations effectuées par le mathématicien, et si nous obtenons des résultats non contradictoires, nous dirons que nous avons capturé une sémantique opérationnelle naturelle de cette branche des mathématiques. Cette sémantique sera relative aux opérations introduites et à leurs lois de manipulation. La notion de vérité mathématique est ainsi subtilement remplacée par celle de validité...

Les notions de syntaxe et de sémantique permettent alors de caractériser les formules et énoncés mathématiques jugés vrais, ainsi que les liens qui ont permis d'établir ce jugement. Reste alors à définir le cadre dans lequel ce processus de jugement intervient : c'est le rôle de la notion de pragmatique. Par ailleurs, si un modèle définit syntaxiquement et sémantiquement ce qu'est une preuve, ce qui concerne le processus de construction de la preuve sera fréquemment rejeté comme "pragmatique" relativement à la situation ainsi formalisée. La distinction essentielle entre trouver et vérifier une preuve est ainsi mise en évidence.

Notre approche d'assistance au mathématicien tend à favoriser la pragmatique au détriment de l'automatisation des raisonnements purement mathématiques.

Les activités paramathématiques qui nous intéressent constituent pour l'essentiel une prise en compte des aspects pragmatiques. Nous sommes ainsi amenés à distinguer trois sortes de pragmatique du langage, chacune étant relative à l'interprétation des acteurs de l'activité mathématique (mathématicien, langage, données) :

- une **pragmatique du processeur**, de prise en compte du mathématicien-processeur ;
- une **pragmatique langagière**, d'usage du langage et des connaissances langagières d'un domaine des mathématiques ;
- une **pragmatique mathématique**, concernant les faits et les vérités mathématiques ;

Ces trois pragmatiques se combinent lors de l'interprétation d'une argumentation ou plus généralement d'un discours mathématique.

La pragmatique du processeur concerne les paramètres mentaux du mathématicien lors de son activité de raisonnement. Ces paramètres concernent notamment les buts et les intentions du mathématicien. Cette pragmatique de l'activité mathématique cherche à reconstituer ces paramètres à partir des énoncés observés (ici les écrits). Elle permet de relier les causalités entre la succession d'énoncés mathématiques. Elle concerne la partie purement mathématique de la situation d'énonciation.

La pragmatique langagière concerne la gestion et l'utilisation du langage en corrélation avec son processeur de traitement, mais indépendamment des connaissances relatives à la vérité mathématique. Elle inclut l'utilisation de "grammaires" structurant les divers niveaux du langage. Elle détermine par exemple l'usage des notations, l'analyse des formules, la gestion des références implicites, les formes possibles d'une définition, les critères de choix d'une présentation, etc.

La pragmatique mathématique concerne l'évolution des connaissances mathématiques représentant les faits mathématiques et leurs relations. Le traitement d'une démonstration est en effet linéaire : il se traduit par une accumulation de faits avec relégation dans le "contexte" des points secondaires à la focalisation de l'attention. Elle gère les objets mathématiques introduits, infère leur type et leurs caractéristiques, bref elle permet à un autre mathématicien de reconstituer le monde dans lequel se place le mathématicien locuteur. Sa partie immergée comporte notamment les multiples connaissances mathématiques communes aux interlocuteurs.

Le plus souvent, le sens ne se réduit pas à ce qui est effectivement dit, mais se traduit par une structure qui inclut une mise en relation avec des informations implicites. Les structures qui composent le sens d'une phrase sont parfois qualifiées de dénotation.

2.3.2. La logique et la résolution de problèmes

Examinons les caractéristiques de la logique et son rôle dans la modélisation de l'activité mathématique. La logique utilise des théories déjà constituées réunissant des objets dépouillés de leur sens et munis de structures récursives "homogènes" idéales pour vérifier par induction et calculer mécaniquement. Ses applications sont alors étrangères aux caractéristiques essentielles de l'activité humaine.

Notre approche interactive de la résolution de problèmes permet de distinguer schématiquement certains aspects de l'activité mathématique où la logique n'est pas adéquate. Voici notamment avant de les développer, trois points sur lesquels la logique pourrait être envisagée au premier abord :

❶ l'usage d'un **langage logique** comme outil d'expression du mathématicien. Il n'aide pas à formuler le problème et doit subir des modifications pour être adapté aux caractéristiques humaines de compréhension et d'utilisation. Les choix langagiers sont donc inappropriés ;

❷ la mise en œuvre de **l'activité mathématique** humaine. Les procédés logiques de déduction sont inopportuns : notamment, la logique ne s'intéresse pas aux procédés cognitifs et aux méthodes de recherche dans la résolution de problèmes ;

❸ l'**argumention** et la **présentation** : la logique ne fournit par exemple aucun critère pour le niveau de détail et le choix des notations pour présenter des structures comme les formules, preuves et démonstations. Leur signification pour l'homme est dissimulée voire perdue dans les aspects syntaxiques.

Développons ces trois points afin de les nuancer. L'usage ❶ de formules logiques est peu fréquent dans la pratique mathématique. Et pourtant, qui n'a pas utilisé dans une forme dérivée de la logique l'expression de la continuité d'une fonction "f" en un point "x_0" :

$$\forall \varepsilon > 0, \exists \alpha > 0 / |x - x_0| < \alpha \rightarrow |f(x) - f(x_0)| < \varepsilon$$

Les mathématiciens utilisent les notations mathématiques usuelles à l'intérieur de ces formules, sans se préoccuper plus que coutume d'un éventuel lien avec des symboles primitifs élémentaires. Les objets manipulés sont typés : les logiques typées rendent compte de ce phénomène, mais en y adjoignant leur propre structure de types. Enfin, il y a des raccourcis dans l'usage des quantificateurs. Ils permettent de focaliser l'attention sur le point à montrer, et d'insérer dans son contexte les éléments nécessaires à son intérêt. Si par exemple "$\exists \alpha > 0$" est remplacé par "$\exists \alpha / \alpha > 0 \wedge$", cela revient à introduire insidieusement dans les manipulations le cas "$\alpha \leq 0$" qui n'est d'aucun intérêt.

Ces procédés permettent d'**impliciter** les informations, au risque de ne plus pouvoir discerner celles qui sont utilisées. La logique et son langage ont pris le parti d'**expliciter** toutes les informations pour les ramener au même niveau. Voici une définition possible du graphe "e_0" d'une fonction de "e_1" dans "e_2" en théorie logique des ensembles :

$$(\forall e_3)(\forall e_4)[<e_3,e_4> \in e_0 \leftrightarrow (e_3 \in e_1 \wedge e_4 \in e_2 \wedge (\forall e_5)(<e_3,e_5> \in e_0 \rightarrow e_4 = e_5))]$$

Le mathématicien retrouve la définition usuelle du graphe d'une fonction, c'est-à-dire un ensemble de couples de la forme "<x, f(x)>". Seulement cette formule en dit déjà trop car elle explicite l'appartenance des éléments au domaine de la fonction ainsi que leurs propriétés. Sans parler de sa difficulté de formulation, pour comprendre la formule il faut tracer pas à pas les constituants et les reformuler en des termes significatifs. Pourtant, la définition précédente est présentée dans une syntaxe adaptée à l'homme. De plus, elle est abrégée puiqu'il reste à remplacer l'abréviation du couple "<x,y>" par l'ensemble "{{x},{x,y}}" dont il faudra ensuite exprimer chacune des abréviations respectives en terme "purement" logique.

La définition complètement développée devient alors incompréhensible et sa manipulation inutilement combinatoire. Pour illustrer l'inutilité pratique d'une telle approche, R. Godement [Godement 66] compare le mathématicien à l'alpiniste :

> Le mathématicien qui essaierait de manipuler de pareils assemblages ressemblerait à l'alpiniste qui, pour choisir ses points d'appui sur une paroi rocheuse, examinerait celle-ci au microscope électronique.

Le langage logique est ainsi trop éloigné du mode de représentation du mathématicien. Ce dernier utilise notamment un mécanisme de structuration par niveaux d'abstraction qui lui permet d'**organiser ses connaissances**[3]. Les concepts introduits sont alors explicités par besoin, pour détailler un niveau d'abstraction inférieur. Cette structuration permet ainsi de morceler l'information en items d'utilisation pour l'homme. Il est, par exemple, souvent utile de considérer "1" comme un signe fondamental.

Sous ces conditions, un **formalisme logique "adapté"** est parfois le meilleur moyen d'exprimer une information, comme dans le cas de la continuité. La formule de la continuité présentée auparavant reste parfois superflue dans le cas où il suffit simplement pour raisonner, de savoir que la limite de "f" en "x_0" est "$f(x_0)$" !

2.3.3. Des procédés déductifs différents

La pratique de l'activité humaine ❷ ne correspond pas non plus aux règles logiques usuelles de déduction de théorèmes. Elles n'utilisent pas de raisonnement hypothétique (si de l'hypothèse "a" on dérive "b", alors "a—>b" est un théorème). Au contraire, les logiques usuelles cherchent à dériver directement "a—>b" à l'aide de règles et axiomes.

A la suite de Gerhard Gentzen, d'autres systèmes dits de **déduction naturelle** cherchent à se rapprocher de l'activité mathématique. Ils utilisent des règles d'introduction et d'élimination de quantificateur ou de connecteur (pour le "ou" on dérive par exemple "a ou b" à partir de "a" ; essayez donc de trouver la règle qui autorise à dériver "c" à partir de "a ou b"). Selon les règles ou les systèmes, certains types de raisonnements mathématiques sont reproduits. De même, différentes approches des mathématiques peuvent ainsi être concrétisées : par exemple, les intuitionnistes refusent la loi du tiers exclu pour des raisons de constructibilité des objets, et éliminent la règle permettant de conclure "a ou non a".

La connaissance de la logique est toutefois indispensable pour mieux comprendre certains aspects de l'activité mathématique. Tout élève craint les pièges du maniement de la négation, de l'implication et des quantificateurs. Même s'ils expriment correctement qu'une fonction n'est pas continue, rares sont ceux qui savent qu'il suffit de montrer l'existence d'un "α" sans devoir chercher à le construire ! De même, la logique éclaire la distinction entre le symbole et l'objet, entre égalité et équivalence, entre implication et déduction, ou entre réduction à l'absurde et contraposition.

L'intérêt de la logique dans la mise en œuvre de l'activité mathématique se situe à deux niveaux :

- mettre en œuvre une activité de raisonnement basée sur une mécanisation informatique des inférences. La logique se comporte alors comme un autre langage informatique dit "assembleur", dédié aux activités de raisonnement (cf l'usage habituel de PROLOG).
- fournir le cadre théorique d'une procédure de décision spécialisée combinatoirement efficace, comme celle de Wu [Chou 88] en géométrie. Une telle activité calculatoire de vérification ou de recherche de théorèmes ne s'applique qu'à des domaines mathématiques spécifiques. La logique n'est alors que l'une des méthodes théoriques possibles, de même que l'algèbre aboutit par exemple à l'algorithme de Risch pour le calcul intégral.

Une variante assouplie de cette approche consiste à proposer des outils de preuve généraux issus de formalismes logiques "associés" au lambda-calcul typé. Son usage est discuté par exemple dans [Parigot 88]. Le Calcul des Constructions [Formel 89] propose une façon théorique et pratique de l'utiliser pour des applications mathématiques, moyennant assistance...

---

[3] Nous présentons partie III (5.2.1) quelques conséquences facheuses d'une structuration des connaissances selon l'approche logique.

Ces résultats concrétisent la qualité d'achèvement de travaux issus d'une majorité de **domaines mathématiques où la pratique humaine est reine.** Pour ces domaines de structure plus hétérogène, l'avenir est dans la modélisation du langage et des connaissances humaines afin d'assister le mathématicien. Cela réintroduit l'automatisation de façon masquée pour effectuer des décisions plus élémentaires selon un objectif d'assistance, mais le savoir-faire et l'intuition du mathématicien peuvent alors s'exprimer de façon déterminante.

L'un des inconvénients principaux des outils logiques est de ne pas offrir de cadre théorique de structuration de définitions, de concepts et de théories. Les logiciens savent mal étendre le langage formel qu'ils étudient. Bien sûr, ils étudient formellement ce que donne l'ajout d'un symbole, d'un axiome ou d'une règle à un système formel. D'ailleurs en logique, une définition est considérée comme une facilité d'écriture du métalangage : elle casse toute preuve basée sur la nature syntaxique des formules, comme par exemple pour compter le nombre d'occurences (cf [Schoenfield 67, p6-8]).

Ainsi en logique, une définition permet uniquement de dire les mêmes choses sous une autre forme. Le point de vue des mathématiques est essentiellement différent : il est conceptuel plutôt que syntaxique. Pabion note par exemple [Pabion 76, p166], que pour le mathématicien le concept de nombre premier est beaucoup plus qu'un simple auxiliaire de raisonnement : il est étudié pour lui-même ! Même avec un point de vue syntaxique, le remplacement systématique provoquerait une explosion combinatoire des déductions. L'édification et l'utilisation de théories ne peut donc se faire sans cette gestion des abréviations introduites, ce qui est un art propre aux mathématiques.

Pour une mise en œuvre de l'activité mathématique, une grande quantité de connaissances mal structurées rend l'approche logique combinatoirement inefficace. L'article de Bledsoe [Bledsoe 86] est ainsi révélateur des difficultés d'utilisation de clauses logiques et met l'accent sur l'importance d'une base de connaissances structurée.

Les possibilités de structuration d'un système formel se font alors sur des considérations pragmatiques calquées sur la pratique du mathématicien. La logique devient alors un simple mode d'expression dont le contrôle constitue le problème principal. Pourquoi dès lors ne pas donner a priori au mathématicien ses connaissances, ses formules et son langage usuel, et étudier expérimentalement ses procédés de recherche et leur contrôle ? Le rajout de définitions et d'axiomes ad hoc ne convient que dans une perspective limitée indépendante d'une **stratégie d'acquisition et d'utilisation des connaissances.**

2.3.4. Une argumentation inexistante

Le dernier point ❹ concerne l'argumention et la présentation. Il est le plus simple à traiter puisque la logique est étrangère à ces considérations. Néanmoins, la logique étant un outil universel, des logiques de discours ont été proposées [Veronis 89]. Indépendamment de la prise en compte du **dialogue**, l'argumentation et la présentation requièrent la **compréhension** d'une démonstration relativement aux besoins de l'interlocuteur humain. Il est évident que l'approche logique ne dit rien sur ce que Popper considère comme le niveau minimal de compréhension d'une preuve : pouvoir saisir l'argument[4].

Quand on fait des mathématiques, on ne produit pas de preuve logique : ce qui nous intéresse ici est la façon dont le mathématicien exprime et résoud ses problèmes, afin de l'assister dans son activité. Si le mathématicien cherche de plus à vérifier sa "preuve mathématique" produite, ce sera éventuellement par traduction en une théorie logique visée, par les mêmes procédés qui transforment des spécifications informatiques en langage machine. La forme de présentation habituelle au mathématicien est ainsi préservée. Les concepts usuels sont en effet essentiels pour l'utilisation des connaissances de démonstration du mathématicien, tandis qu'une présentation adaptée est déterminante pour la compréhension du discours et l'identification des règles utilisables.

La pratique mathématique est d'ailleurs semblable à un discours au métaniveau qui donne des indications de méthode à employer et de points de passage, plus que des règles explicites comme "généralisation" ou "modus ponens". Une preuve n'est pas alors explicitement construite, et les règles employées sont moins élémentaires que celles des théories logiques. D'ailleurs, les méthodes générales de démonstration

[4] p 576 du livre [Kline 89], qui est d'un intérêt historique et épistémologique certain pour les liens entre logique et mathématiques.

mathématique (récurrence...) sont exprimées sous forme de métathéorèmes en logique du premier ordre (principe d'induction...).

La présentation des systèmes formels a permis de donner un **modèle descriptif idéal** de la partie déductive des mathématiques. Il ne faut pas confondre pour autant un procédé de cartographie avec le monde qu'il permet de représenter. Une preuve mathématique réelle est-elle alors une preuve logique formelle abrégée ou incomplète ? Notre réponse est mitigée : une preuve réelle et sa démonstration sont destinées à convaincre les usagers auxquels elles sont destinées. En ce sens, elles n'ont aucunement l'intention ni souvent la possibilité de remonter à une axiomatisation logique primitive, même dans des théories possédant une formalisation logique comme la théorie intuitive des ensembles ou l'Analyse Non Standard.

Il résulte de cette analyse que :

L'approche logique n'est pas adaptée à l'activité mathématique naturelle

Il nous reste alors à étudier et proposer d'autres solutions pour pallier les limites d'une formalisation d'aspiration logiciste.

3. REGARDS SUR L'USAGE DU LANGAGE MATHEMATIQUE

3.1. L'ETUDE DU LANGAGE MATHEMATIQUE

3.1.1. L'outil de pensée du mathématicien

> Réfléchir sur le langage mathématique tel qu'il est employé par les grands mathématiciens eux-mêmes dans leurs recherches, leurs publications ou leur enseignement, me semble indispensable. Réflexion d'autant plus urgente que le formalisme des mathématiques actuelles donne trop souvent l'illusion de se suffire a lui-même.
>
> M. Loi, organisateur du séminaire de philosophie et mathématiques
> à l'Ecole Normale Supérieure de la rue d'Ulm [Loi 82].

Le Langage Mathématique est l'outil de pensée du mathématicien. Son étude est importante pour comprendre la résolution de problèmes autrement que comme l'application d'heuristiques données a priori pour résoudre une classe restreinte de problèmes. L'étude du langage montre aussi que les connaissances utilisées par le mathématicien ont une organisation complexe qui doit être spécifiquement étudiée.

Le langage est l'intermédaire indispensable à toute assistance. Notre recherche d'assistance s'intéresse au déroulement de l'activité mathématique et pas seulement à des tâches très restreintes. Il est donc indispensable d'étudier le langage et son utilisation comme outil de pensée du mathématicien. Les quelques études du langage s'intéressent en effet rarement à l'utilisation du langage dans la résolution de problèmes.

Par son étude du rôle du langage dans l'activité mathématique, ce chapitre nous permet de comprendre le rôle de la flexibilité d'utilisation du langage. Il balise ainsi implicitement des contraintes à prendre en compte pour réaliser un système susceptible de satisfaire et d'attirer les mathématiciens.

Le postulat initial de notre analyse est que :

Il est possible de modéliser l'activité mathématique !

Notamment, cette analyse cherche à déterminer les mécanismes d'utilisation du langage afin de pouvoir ultérieurement les modéliser.

Les tentatives effectuées jusqu'à présent sont limitées par le fait qu'aucune n'a véritablement cherché à **modéliser un domaine mathématique non restreint**. Les mathématiques sont constituées de domaines très vastes dont la description et la modélisation constituent un projet ambitieux. La donnée de ces contraintes statiques permet, selon notre approche expérimentale, d'élaborer et de modéliser les aspects dynamiques du guidage et du raisonnement.

La limite classique due à l'introduction de la créativité dans le raisonnement n'existe pas dans notre approche d'assistance : elle est simplement à la charge du mathématicien. Les **difficultés** à surmonter sont alors de :

- représenter les productions mathématiques sans les schématiser : formules, théorèmes, définitions, problèmes, preuves, démonstrations ;
- modéliser un vaste domaine conceptuel dans ses composantes mathématiques et langagières ;
- utiliser cette modélisation dans une dynamique d'activité mathématique ;

Nous explorons ici ces difficultés. Nous proposons ultérieurement de surmonter la quantité et la diversité des ingrédients nécessaires par une boucle de réponse rapide aux besoins engendrés par la pratique d'un système informatique attrayant.

Notre objectif présent est donc principalement une recherche et une tentative de description des domaines mathématiques usuels, par opposition aux tentatives passées qui focalisaient leur efforts sur l'automatisation du raisonnement.

3.1.2. Partition du Langage Mathématique

Pour qualifier le langage des Mathématiques dans sa généralité, nous avons choisi le vocable "Langage Mathématique" par analogie avec "Langage Naturel". Afin de l'étudier, nous avons identifié deux fonctions du Langage Mathématique :

- La fonction mathématique correspond à l'activité purement mathématique individuelle du mathématicien ;
- La fonction de communication permet d'échanger des informations entre le mathématicien et l'extérieur, ce qui relève d'une activité paramathématique.

Lorsqu'elle s'applique à un texte, cette partition permet de distinguer la forme syntaxique du sens mathématique. Nous en matérialisons la frontière par une *ligne de démarcation*. Cette ligne de démarcation où la précision du langage est optimale n'est atteinte que dans les systèmes formels où le sens d'une formule est réduit à sa forme syntaxique.

Comme pour le Langage Naturel, nous distinguons schématiquement deux niveaux différents dans un texte mathématique : le *niveau élémentaire* des constituants du langage, et le *niveau macroscopique* du discours. Le niveau élémentaire concerne les constituants jusqu'au niveau des énoncés, tandis que le niveau macroscopique les assemble pour former des preuves et démonstrations.

Pour appréhender cette distinction, examinons la compréhension d'une démonstration. Pierre Cartier déclare [Cartier 89] :

> il arrive qu'on soit obligé d'avancer comme dans un tunnel, pas à pas.

Cette progression se traduit par le suivi du niveau élémentaire sans compréhension du niveau macroscopique. Il est alors possible de vérifier pas à pas la correction des transitions sans comprendre la globalité de la preuve.

Vu la complexité, cet obstacle serait rédhibitoire pour trouver une démonstration. D'un point de vue heuristique, Georges Glaeser [Glaeser 83] note que la construction pas à pas apparaît artificielle. Cette nuance entre construction pas à pas et compréhension constitue à notre avis la limite essentielle de la démonstration automatique. Par contre rien n'empêche a priori un assistant de suivre la progression d'un mathématicien. C'est pourquoi l'assistance est, d'après nous, la seule voie permettant un support informatique de l'activité mathématique.

La structure du niveau élémentaire se prête aux vérifications, tandis que celle du niveau macroscopique traduit les principes heuristiques de la résolution de problèmes. Nous pouvons alors dire sommairement que le sens du niveau élémentaire est aisément transcriptible par des procédés formels, mais que celui du niveau macroscopique se ramène beaucoup moins facilement à la composition de ses parties.

Le texte : "montrer que la fonction d(x,y) est une distance euclidienne"
a une signification apparente associée à une notion de problème et à une liste de propriétés apparentes concernant les fonctions, la distance euclidienne et "d". Toutefois, la compréhension ne s'arrête pas au simple maniement formel de ces caractéristiques apparentes : elle inclut un savoir-faire associé à une expérience personnelle voire intime des manipulations.

Selon leur rôle, les connaissances associées à ce savoir-faire sont corrélées de façon plus ou moins directe (nous emploierons le qualificatif de "*profond*") aux significations apparentes. Ainsi, un énoncé mathématique ne peut être relié à un sens utilisé dans l'activité mathématique que de façon complexe et indirecte, tandis que les objets comme la fonction "d" peuvent être caractérisés par une structure associée regroupant leurs propriétés. Cela correspond informellement à une écart plus ou moins important à partir de la ligne de démarcation de la figure 1.8.

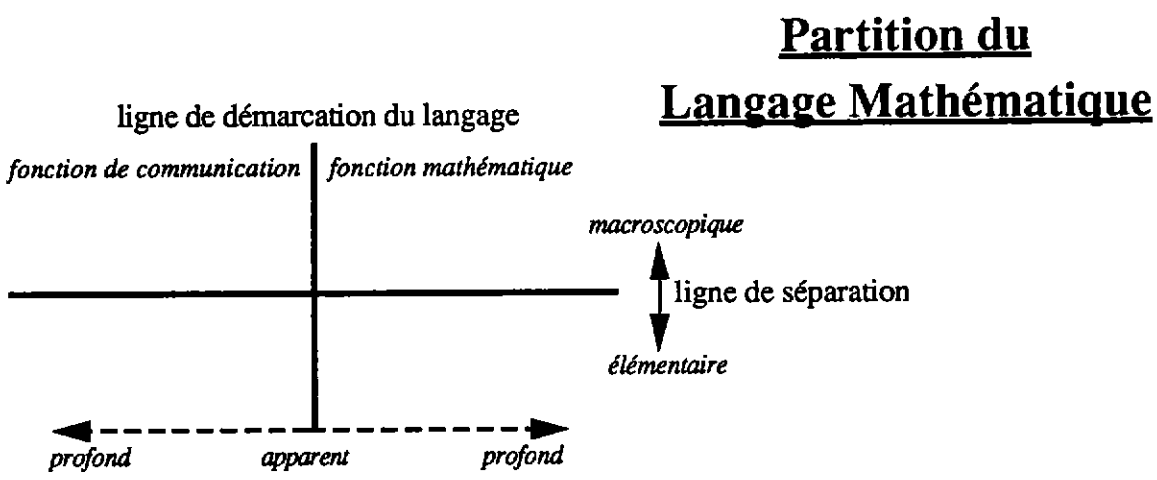


figure 1.8

Bien entendu, les quatre zones de cette partition du Langage Mathématique interagissent. Par exemple au niveau élémentaire, l'introduction de la notation "d(x,y)" est la trace textuelle (fonction de communication élémentaire) d'une composition de concepts définissant la distance euclidienne (fonction mathématique élémentaire). Cette dernière est le résultat d'une réflexion macroscopique qui se traduit par un texte macroscopique qui utilise (ou définit) cette notation. Par ailleurs, un choix élémentaire comme celui d'un symbole, dépend de sa fonction mathématique et de conventions macroscopiques qui elles-même sont relatives au domaine mathématique traité, ici la géométrie euclidienne.

3.1.3. La distance élémentaire-macroscopique

Nous voulons déterminer la distance qui sépare un énoncé mathématique élémentaire des structures macroscopiques dans lesquelles il intervient.

Pour la fonction mathématique, cette distance se conçoit comme la structure d'organisation des connaissances mathématiques pour la phase de constitution, ou la structure d'une preuve et démonstration pour la phase d'utilisation. Elle se conçoit éventuellement comme la structure d'organisation mentale pour la phase d'apprentissage.

Pour la fonction de communication, active dans la phase de diffusion, nous avons identifié une dimension langagière et une dimension linguistique qui viennent englober la dimension élémentaire initiale. La distance élémentaire-macroscopique est alors schématisée figure 1.9.

Les pages 158 et 159 de [Godement 66] (reproduites ci-après) sont illustratives des dimensions identifiées. Nous nous contentons ici d'un commentaire très succint de ce texte.

Pour la dimension langagière, la présentation visuelle inclut les aspects graphiques et typographiques comme l'utilisation d'encadrement, d'italiques, de centrage, de paragraphes, etc. L'espace de désignation regroupe les moyens terminologiques, textuels et langagiers faisant en permanence le lien entre les objets de discours.

puisque pour additionner des vecteurs d'origine O il suffit d'additionner leurs composantes par rapport aux axes de coordonnées Ox, Oy.

Soit P le point d'affixe $z = x + iy$; le nombre

$$N(z) = \bar{z}z = x^2 + y^2$$

est visiblement donné par

$$N(z) = OP^2,$$

en vertu du théorème de Pythagore. La distance de P au point O, i.e. le nombre

$$r = OP = \sqrt{x^2 + y^2} = \sqrt{\bar{z}z} \tag{15}$$

s'appelle le **module** ou la **valeur absolue** de z, et se désigne par la notation

$$|z|;$$

on a toujours $|z| \geqslant 0$, et on a $|z| = 0$ si et seulement si $z = 0$; de plus, les inégalités classiques entre les côtés d'un triangle montrent, si l'on examine la figure, que l'on a $|u + v| \leqslant |u| + |v|$ quels que soient $u, v \in \mathbf{C}$, et plus généralement

$$\boxed{|z_1 + \cdots + z_n| \leqslant |z_1| + \cdots + |z_n|} \tag{16}$$

quels que soient les nombres complexes $z_1, \ldots, z_n$; ce résultat est souvent cité sous le nom d'**inégalité du triangle**

Supposons $z \neq 0$; l'angle

$$\theta = \widehat{(\overrightarrow{Ox}, \overrightarrow{OP})},$$

qui est un nombre réel modulo 2π (§ 4, *Exemple* 10), s'appelle l'**argument** du nombre complexe $z \neq 0$, et se désigne souvent par la notation

$$\mathrm{Arg}(z).$$

La figure montre que les parties réelle et imaginaire de z sont données, en fonction de la valeur absolue r et de l'argument θ de z, par les formules

$$x = r.\cos\theta, \qquad y = r.\sin\theta, \tag{17}$$

de sorte qu'on peut aussi écrire

$$\boxed{z = r(\cos\theta + i.\sin\theta)}; \tag{18}$$

cette formule s'appelle la **représentation trigonométrique** de z. On notera qu'inversement la relation

$$z = r(\cos\theta + i.\sin\theta) \quad \text{avec} \quad r > 0 \quad \text{implique} \quad r = |z|, \ \theta = \mathrm{Arg}\,(z);$$

car la relation considérée montre que les parties réelle et imaginaire de z sont

$$x = r.\cos\theta, \qquad y = r.\sin\theta,$$

d'où résulte d'abord que $r^2 = x^2 + y^2$, et, comme r est positif, $r = |z|$; notant θ' l'argument de z il vient alors $\cos\theta = \cos\theta'$, $\sin\theta = \sin\theta'$, d'où $\theta = \theta'$ à un multiple près de 2π, ce qui établit notre assertion.

On va déduire de là les formules

$$\boxed{|z_1z_2| = |z_1|.|z_2|, \qquad \mathrm{Arg}\,(z_1z_2) = \mathrm{Arg}\,(z_1) + \mathrm{Arg}\,(z_2)}, \tag{19}$$

qui permettent de calculer le module et l'argument d'un produit de nombres complexes. Pour cela, écrivons

$$z_1 = r_1\,(\cos\theta_1 + i.\sin\theta_1), \qquad z_2 = r_2\,(\cos\theta_2 + i.\sin\theta_2)$$

en mettant en évidence les modules et les arguments de z_1 et z_2; il vient

$$z_1z_2 = r_1r_2\,(\cos\theta_1 + i.\sin\theta_1)\,(\cos\theta_2 + i.\sin\theta_2)$$

et comme r_1r_2 est positif, il suffit, pour établir le résultat cherché, de prouver la relation

$$(\cos\theta_1 + i.\sin\theta_1).(\cos\theta_2 + i.\sin\theta_2) = \cos(\theta_1 + \theta_2) + i.\sin(\theta_1 + \theta_2);$$

or, d'après les règles de multiplication des nombres complexes, la partie réelle du premier membre est égale à

$$\cos\theta_1.\cos\theta_2 - \sin\theta_1.\sin\theta_2 = \cos(\theta_1 + \theta_2),$$

et la partie imaginaire à

$$\cos\theta_1.\sin\theta_2 + \sin\theta_1.\cos\theta_2 = \sin(\theta_1 + \theta_2),$$

ce qui établit le résultat annoncé.

Ce résultat s'étend naturellement à un produit de plusieurs facteurs, sous la forme

$$\boxed{\prod_{p=1}^{p=n}(\cos\theta_p + i.\sin\theta_p) = \cos\theta + i.\sin\theta \quad \text{où} \quad \theta = \sum_{p=1}^{p=n}\theta_p}. \tag{20}$$

En particulier, si l'on suppose les θ_p égaux à un même angle θ, il vient la célèbre **formule de De Moivre**

$$\boxed{(\cos\theta + i.\sin\theta)^n = \cos(n\theta) + i.\sin(n\theta)}. \tag{21}$$

Remarque 2. Les formules

$$|z_1z_2| = |z_1|.|z_2|, \qquad \mathrm{Arg}(z_1z_2) = \mathrm{Arg}(z_1) + \mathrm{Arg}(z_2)$$

permettent de donner une interprétation « géométrique » de la multiplication

4, 2 [Godement 66]

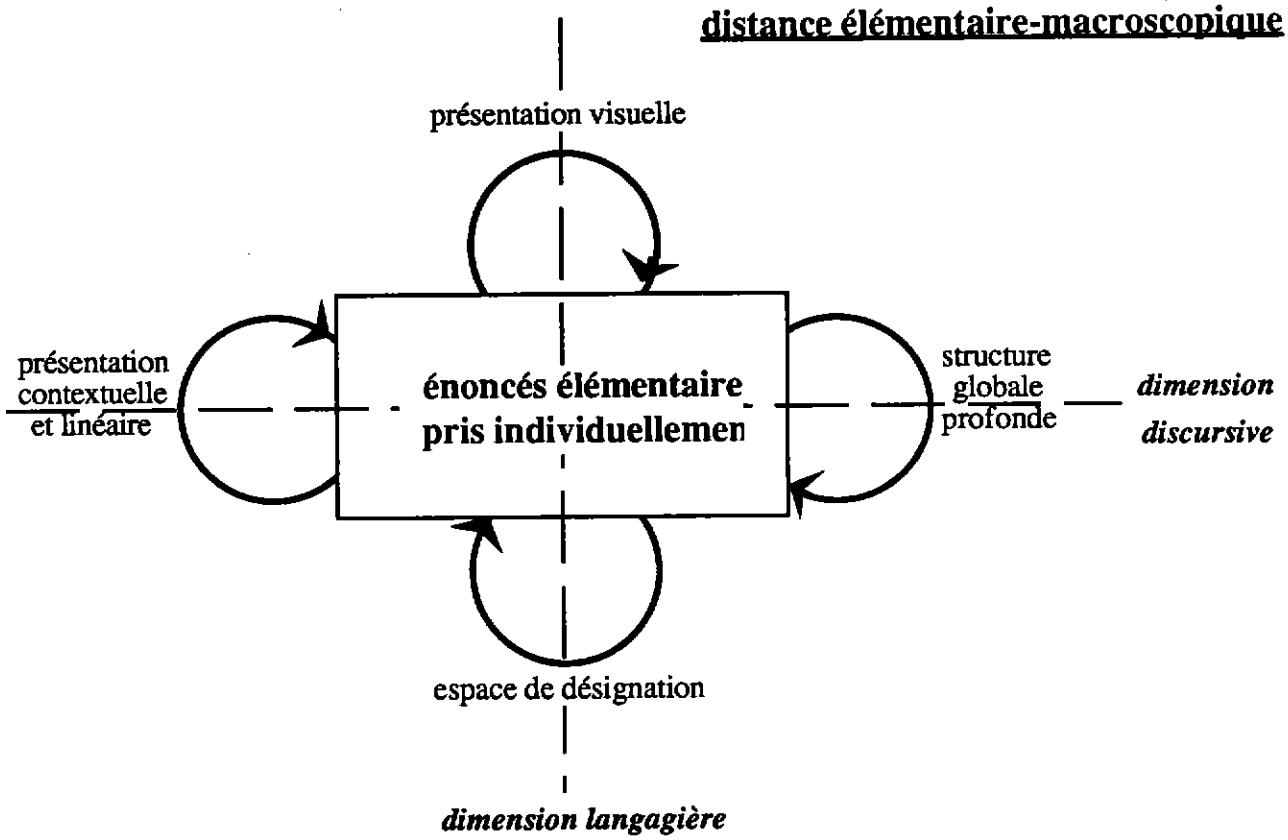


figure 1.9

La dimension discursive est partagée entre une structure globale profonde et une présentation contextuelle linéaire qui décide comment linéariser le discours. Plusieurs composantes de la structure globale du discours peuvent ici être recensées. Nous leur associons ci-dessous des exemples dans la présentation contextuelle linéaire :

- l'introduction des concepts et leur définition mathématique et terminologique :
  cela est matérialisé par des phrases relatant par exemple la structure mathématique associée aux nombres complexes.
- l'introduction de relations calculatoirement intéressantes :
  elle s'effectue par le biais de structures relativement explicites de justification, allant de la simple description ("La figure montre") à une véritable argumentation ("Pour cela, écrivons...").
- l'exploitation des résultats connus par un raisonnement :
  elle fait intervenir la dynamique de raisonnement et comprend les passages justifiables de l'appellation : démonstration.

L'examen de cette distance élémentaire-macroscopique révèle que la dimension langagière est modélisable à l'aide de paramètres mathématiques relativement simples. Par contre, la dimension discursive se révèle très ardue. Un premier pas consiste à modéliser des structures mathématiques capables de regrouper les énoncés élémentaires intervenant dans la présentation textuelle linéaire. Pour la constitution d'un modèle seule la structure profonde est utile. Pour la phase d'utilisation, les moyens mis en œuvre doivent être plus importants afin de modéliser et présenter linéairement des textes dits "démonstrations".

Nous qualifierons dorénavant d'*élémentaires* les manipulations et les connaissances qui concernent ces énoncés élémentaires (symboles, formules, règles d'inférences, déductions...). Nous allons progressivement étendre ce qualificatif "élémentaire" à ce qui est aisément exprimable et réalisable par un mathématicien, et dont le traitement ne lui demande pas un effort conscient significatif.

3.1.4. Les fonctions du Langage Mathématique

William Hamilton (cité dans [Hadamard 75, p 80-81]) utilise une comparaison intéressante avec la construction d'un tunnel dans une couche sablonneuse. Le langage est pour l'esprit ce qu'une arche de maçonnerie est au tunnel : le pouvoir de penser ne dépend pas des mots, mais ils aident à pousser l'excavation plus avant.

Les deux fonctions du Langage Mathématique - communication et mathématique - peuvent être analysées selon deux objectifs :

- savoir quelles fonctions utilitaires assume le langage, afin qu'elles soient prises en compte dans un système d'assistance à l'activité mathématique.
- savoir comment le Langage Mathématique favorise la production de nouveaux résultats mathématiques. Ces aspects créatifs suggèrent un usage non stéréotypé du langage et viennent conforter notre paradigme d'assistance.

Pour bien exploiter le langage et définir les tâches envisageables, nous précisons les principales fonctions utilitaires qui sont récapitulées figure 1.10 :

- la **fonction mathématique descriptive** permet d'exprimer des formules, des connaissances et des preuves ;
- la **fonction mathématique évaluative** permet de conjecturer et de vérifier la validité des énoncés par des déductions ;
- la **fonction de communication épistémique** permet la diffusion du savoir et ainsi sa validation par absence de réfutation ;
- la **fonction de communication explicative** permet de traduire et présenter une pensée par le biais d'énoncés réfutables.

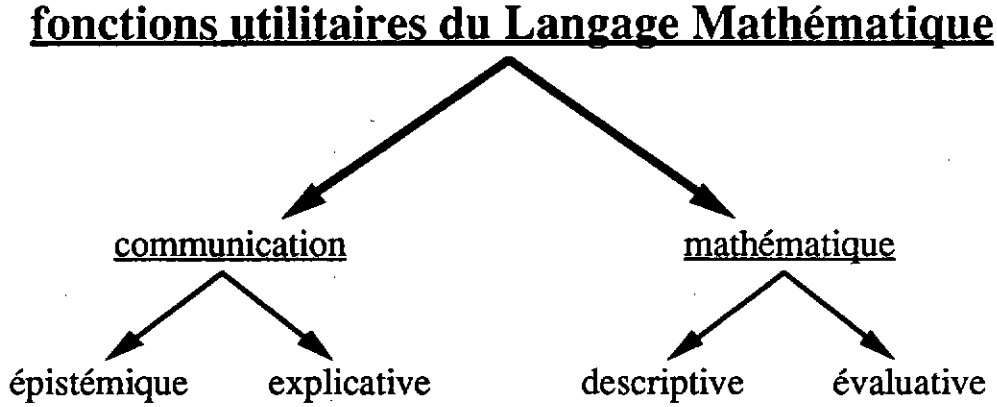


figure 1.10

Le rôle historique des **aspects créatifs** du langage a été étudié notamment par Jean Dhombres. Il se traduisent par un savant dosage de l'invention et de la découverte pour la fonction mathématique du langage. Nous en avons présenté un aperçu section précédente.

La fonction de communication a aussi une importance capitale pour la créativité en mathématiques grâce à la validation et la diffusion des résultats et des démonstrations. Le temps d'une boucle n'est pas forcément réparti sur des décennies comme dans la formule d'Euler citée par Imre Lakatos. Il peut être très court dans le cas d'une communication à usage personnel. Le fait de disposer de brouillons hâtivement griffonnés pour capturer le fil de la pensée est par exemple un facteur certain de créativité.

La question relative à tout langage est alors : "**comment la communication est-elle possible ?**", c.-à-d. comment partager des concepts aussi précis que ceux utilisés en mathématiques ? Pour ne pas développer davantage ce point, nous postulons que les concepts sont définis relativement les uns aux autres et que seule une partie opératoire et sociale est partagée par la communauté mathématique (et évolue dans le temps).

3.2. LES LANGAGES DE FIGURES

3.2.1. Les voies de communication de l'activité mathématique

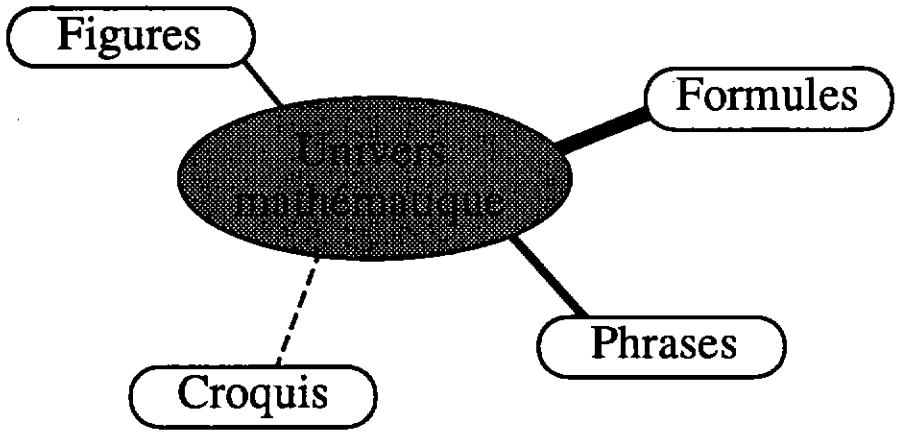


figure 1.11

La figure 1.11 caractérise tout d'abord l'activité mathématique par ses manifestations tangibles. Le vocable *"Univers mathématique"* désigne le monde virtuel dans lequel opèrent les mathématiques. Nous scindons les **voies de communication** disponibles en quatre catégories de langages. Nous parlerons ainsi des langages de Phrases, Formules, Figures et Croquis.

Le vocable ***"Formule"*** (resp. ***"Phrase"***) qualifie les entités sémiologiques (resp. textuelles) des formalismes utilisés pour désigner l'Univers mathématique. Nous utiliserons le vocable *"formule"* pour désigner des liens correspondant aux intensions de l'Univers mathématique comme "le sinus de zéro est zéro" ou "un moins un". Ces formules peuvent selon le choix être présentées dans un langage de Phrases comme dans un langage de Formules.

De même, le vocable ***"Figure"*** qualifie les entités graphiques organisées comme moyen de communication ou comme moyen de résolution ayant une validité mathématique reconnue (opération "posée" de multiplication ou d'extraction d'une racine carrée, tables de logarithmes ou d'intégrales, schéma d'un dispositif physique, figure géométrique, tableau de variation d'une fonction, graphe d'une preuve, manipulation interactive d'hypercube, etc.).

Nous les avons différenciées des ***"Croquis"*** (brouillons, graffitis, esquisses, essais d'un exemple, etc.) que leur concepteur destine à son usage personnel et dont l'aspect informel des signes et de leurs groupements ne permet pas de les constituer en système de communication. Nous ne développerons pas ici l'intérêt d'un support mémoriel externe pour saisir les aspects importants de la pensée, sans trop la ralentir par des contraintes de mise en œuvre.

Les langages de Croquis recherchent ainsi les fonctions créatives sans aucune prétention utilitaire : ils recherchent une expressivité maximale au détriment d'une transcription utilitaire ultérieure. Leur étude est importante pour la compréhension du cheminement de la pensée car elle révèle des aspects non verbalisés ou dont le compte-rendu est omis.

Notre dichotomie des voies de communication a un objectif descriptif. Par exemple l'imposant travail sur les marqueurs logico-discursifs [Gerbier 87] ne fait pas de distinction entre Phrases et Formules. En revanche, ses auteurs proposent une comparaison très intéressante de textes de démonstrations présentés graphiquement dans un langage de Figures et de Croquis.

La thèse de Colette Laborde [Laborde 82] étudie les modes d'intégration des langages de Phrases, Formules et Figures. Elle étudie par exemple comment des éléments des Formules sont intégrés dans les Phrases, ou par quels moyens une Figure est décrite à l'aide de Phrases. Cette étude se place dans une perspective pédagogique plus générale. La nature des erreurs et imperfections montre en particulier la difficulté d'utilisation intrinsèque et conjointe de ces langages.

3.2.2. L'importance des Figures

Les langages de Figures sont fréquemment utilisés pour la commmunication et la compréhension d'informations à caractère mathématique. Il est donc indispensable de les analyser afin de connaître :

- les raisons qui justifient leur usage ;
- les diverses sortes de Figures possibles ;
- leur rôle dans le discours mathématique ;
- les moyens mis en œuvre pour leur traitement (de l'écriture à l'interprétation) ;

Nous abordons successivement ces points en commencant bien entendu par les raisons qui justifient l'usage de langages à caractéristiques essentiellement visuelles, par opposition aux langages discursifs linéaires.

Il est toujours possible de transcrire un message en langage de Figures avec un langage de Phrases ou de Formules. Par exemple une image peut être digitalisée. Ce postulat est valable dès que la quantité d'information pertinente est finie. Cette transcription théorique révèle la complémentarité des langages de Figures avec les langages de Phrases et Formules : leur utilisation est utile car nullement indispensable !

L'article de J.H. Larkin et H.A. Simon explicite et illustre l'avantage d'une figure sur la linéarité des mots [Larkin 87]. L'avantage est issu du plan de la perception : une représentation picturale - quoique d'information équivalente - peut être **calculatoirement supérieure** à une représentation linéaire. Elle met en effet en évidence des invariants et regroupe des informations topologiques et géométriques. Cela favorise les intuitions et procure des contre-exemples. Par ailleurs, elles traduisent mieux les images mentales (cf. par exemple [Denis 89]). Bref, une figure permet de manipuler un modèle (synthétique) au lieu d'énoncés analytiques.

Cet usage du spatial reste valable lorsqu'il s'agit simplement de présenter graphiquement des informations, par exemple dans un tableau. Il est ainsi recommandé de transcrire un problème en langage de Figures dès que les structures manipulées par le problème disposent d'une interprétation graphique "pertinente". Cela suppose réciproquement que toute Figure est associée à une interprétation (sa légende) permettant de décoder les informations à exploiter.

Une figure géométrique n'est pas un assemblage de position spatiales. La précision de son contour est limitée et son tracé a nécessité des choix d'emplacements caractéristiques. C'est pourquoi la puissance d'expressivité des figures est plutôt utilisée pour des fonctions créatives, leur contenu mathématique étant recodé en langages de Phrases et Formules pour acquérir une fonction utilitaire reconnue.

Une Figure est susceptible de présenter plusieurs types d'information à partir des mêmes données. Cette **pluralité** est aussi l'un des avantages notables des langages de Figures. Une illustration de cette pluralité est donnée par le graphe de la fonction entière bornée utilisée pour le jeu du professeur Rufus Isaacs. La figure 14.8 (cf. Annexes) qui sert de support au commentaire général comporte :

- la représentation du plan discret (représentation de base du problème) ;
- les possibilités de déplacement depuis un point particulier (exemple décrivant l'espace des opérations possibles) ;
- les valeurs en chaque point de la fonction d'évaluation (fonction pertinente pour la résolution du problème) ;
- une mise en valeur des points gagnants (permettant de visualiser l'espace des chemins solution) ;
- une signalisation des coups diagonaux gagnants (indication d'un choix gagnant pour un type d'opération plus difficile à diagnostiquer).

Le tout est bien entendu coordonné de façon à respecter la lisibilité de la figure. La légende, ou plutôt ici les commentaires que la figure accompagne, décrit explicitement les quatre premiers points. Les possibilités de déplacement depuis un point particulier bénéficient des modes visuels et textuels : elles sont indiquées à la fois sur la figure et dans les commentaires. Cette **complémentarité** se traduit ici par duplication, mais elle peut le faire plus simplement en exploitant le support le mieux adapté. Ainsi, la répartition des chemins solution, visuellement mise en évidence dans la figure, est évoquée explicitement dans les commentaires.

Pour conclure, l'absence présente de la figure commentée rend l'interprétation de ces paragraphes difficiles, alors que la duplication de la figure située en Annexes suffirait à les rendre évidents...

3.2.3. Typologie des Figures

Après analyse d'un échantillon dont les ingrédients caractéristiques sont présentés en Annexes, nous avons différencié trois types de Figures :

- les **produits cartésiens discrets** :
  tableaux, matrices, déterminants, etc ;
- les **représentations spatiales** :
  objets "physiques", géométrie, graphes de fonctions, etc ;
- les **structures discrètes** :
  graphes, diagrammes fonctionnels, diagrammes de Venn, diagrammes d'états, fractions continues, etc ;

La frontière entre les divers types de langages est floue en mathématiques puisque les langages de Formules exploitent déjà l'apparence visuelle. Certains problèmes d'intégration privilégient ainsi la reconnaissance de la forme de la formule. Nous avons ainsi placé dans les langages de Figures tout langage privilégiant la disposition graphique dans la nature de son traitement.

Nous classons les tableaux et matrices dans la typologie des Figures, car leur traitement s'apparente à celui des autres objets visuels. De même, les graphes sont des structures à exploitation essentiellement visuelle quoiqu'ils puissent être codés par le produit cartésien de leurs sommets. A la limite, nous considérons les fractions continues parmi les structures discrètes, puisqu'elles exploitent un procédé de construction que le mathématicien se représente d'abord visuellement. De même, nous comptons la pose d'une addition et de sa preuve par neuf comme un exemple de Figure (structure discrète).

Vu la pluralité des informations possibles, cette typologie caractérise les Figures à partir de leur organisation descriptive. Prenons l'exemple des produits cartésiens discrets ; nous disposons dans notre échantillon de :

- un échiquier
  cases colorées référencées par numérotation des lignes et colonnes
- une table de multiplication
  cases référencées par nommage des lignes et colonnes
- un tableau de correspondance entre types de sommets
  cases référencées par nommage des lignes et colonnes
- un triangle de Pascal
  emplacements référencés par nommage des lignes et colonnes
- un tableau de coordonnées point-valeur
  cases référencées par le contenu, avec identification des deux lignes
- un graphe de fonction entière bornée
  points référencés par leurs coordonnées
- une correspondance entre écritures des nombres
  tableau informel en trois colonnes référencées par le contenu et identifiées par le type de l'écriture (décimal, binaire, développement binaire)
- des matrices et déterminants
  référence par numérotation standard implicite, avec possibilité de nommage de blocs, lignes et colonnes (exemple de Δ_1)

Cet échantillon montre la diversité des organisations descriptives possibles, mais aussi la faisabilité d'un paramétrage simple (taille, case ou point, quadrillage ou non, type de référence, etc). Une typologie plus précise revient à classifier ces paramètres, et à identifier certains choix standard. En effet, des organisations descriptives classiques sont couramment utilisées et déductibles de l'énoncé du problème (matrice, déterminant, graphe de fonction entière, tableau de correspondance, etc).

De même, une typologie des représentations spatiales et des structures discrètes peut être établie. Il est alors possible de proposer des outils informatiques pour gérer ces représentations et connaître les organisations descriptives courantes. Un tel transfert sur informatique de schémas de figures paramétrables reste à constituer. De tels outils seront particulièrement intéressants lorsqu'ils pourront être exploités en relation directe avec le problème à traiter. Cela nécessite très certainement une compréhension relative du rôle et des moyens de traitement des figures.

3.2.4. Le rôle des Figures

Les Figures sont souvent utiles comme intermédiaires de formalisation d'un problème "réel". C'est le cas notamment du premier exemple de chaque type de Figure présenté dans les Annexes :

- l'échiquier, pour les produits cartésiens discrets ;
- les coupes d'une pizza, pour les représentations spatiales ;
- le graphe de l'Europe, pour les structures discrètes.

Par exemple, la représentation échiquéenne supporte l'ensemble des contraintes envisageables dans les problèmes dérivés à partir de cette représentation. Selon la nature des contraintes introduites le problème peut, comme ici, être ultérieurement formalisé par un tableau, puis traduit en un graphe. De même, le problème des coupes d'une pizza est formalisé en une représentation géométrique et il utilise finalement la théorie des nombres pour sa résolution.

Une représentation est alors un espace de description initial qui contient les ingrédients susceptibles d'intervenir dans les problèmes qu'elle permet d'exprimer. La représentation échiquéenne apporte ainsi des contraintes spatiales, des opérations et des objectifs échiquéens standard à partir desquels est formé l'espace de recherche tel qu'il est classiquement décrit pour la résolution d'un problème.

La représentation initiale se traduit généralement bien à l'aide de langages de Figures, mais elle est oubliée dans un problème mathématique formalisé. Nous avons alors identifé quatre rôles principaux des Figures :

- **description** :
  Il s'agit ici simplement de présenter des informations. Cela convient pour chacun des trois types lorsque la description est régulière (table de valeurs) ou graphique (graphe ou graphe d'une fonction).
- **construction** :
  Le but est ici de privilégier l'aspect dynamique en décrivant des relations entre informations corrélées. Cela s'applique aux trois types, par exemple à l'aide d'une succession de déplacements dans un espace cartésien discret, d'une séquence de construction en géométrie, ou d'une extraction d'un sous-graphe. Ces procédés, utilisés dans nos Annexes pour des textes écrits, acquièrent une dimension nouvelle grâce à la visualisation dynamique (construction d'une image fractale, etc.)
- **aide à l'intuition** :
  L'exploitation de la forme des matrices est un exemple illustrant l'aide d'une disposition spatiale pour l'intuition. Cette aide exploite un autre point de vue plus global susceptible de mieux faire apparaître certaines régularités, éventuellement non discernées auparavant. En ce sens, une figure est aussi une aide à la découverte. Par ailleurs, les Figures peuvent être volontairement utilisées pour servir d'interprétation abstraite d'une autre représentation, comme dans le cas des diagrammes de Venn[1]. La nature des informations ainsi exploitables sont ainsi la localité, la forme, les dépendances, la catégorisation par traits, etc.
- **illustration** :
  Une Figure complète avantageusement une description linéraire. Donner un exemple sur une Figure permet de visualiser et facilite ainsi la compréhension, la mémorisation, la mise en œuvre active ultérieure, etc. Nous avons ainsi identifié trois sortes d'exemples, qui sont d'ailleurs valables quel que soit le langage des mathématiques utilisé :
  - les **exemples génériques** :
    Tous les autres exemples peuvent s'y ramener moyennant des transformations préservant les invariants pertinents du problème. C'est le cas par exemple du choix des coupes d'une pizza : les figures à n coupes "générales" sont toutes superposables, par un procédé similaire à celui employé en topologie algébrique pour caractériser les singularités des surfaces. Lorsque la description présente plusieurs cas, l'exemple n'en recouvre qu'un seul.
  - les **exemples typiques** :
    Il s'agit ici d'un exemple simple présentant des difficultés semblables à celles susceptibles d'être rencontrées. Ils exploitent ainsi les capacités de généralisation latentes pour montrer comment traiter le cas général. La simplicité choisie doit néanmoins rendre impossible l'utilisation de cet exemple pour illustrer un autre concept pertinent.
    Alors que les exemples génériques mettent l'emphase sur les aspects descriptifs, les exemples typiques exploitent plutôt le procédé de construction pour atteindre la généralité qu'ils ne peuvent exhiber. Il font alors appel à des analogies de construction. L'exemple du graphe biparti-complet $K_{3,2}$ permet ainsi au mathématicien de trouver le procédé de description et de construction de tout graphe biparti-complet.

[1] L'informatique a redécouvert et formalisé récemment le procédé de raisonnement dit d'*interprétation abstraite*. D'autres exemples caractéristiques sont la règle des signes ou le raisonnement sur les dimensions en physique ou en mathématiques (cf [Cipra 83]). Les mathématiques ont l'avantage de disposer de procédés constructifs grâce à l'expérience accumulée dans l'utilisation de tels raisonnements.

- les **exemples particuliers (spécifiques et singuliers)** :
  Il s'agit ici d'exhiber un élément d'un ensemble. Nous l'appelons *exemple spécifique* lorsque cet élément n'a pas de caractéristique notable, et *exemple singulier* lorsqu'il s'agit d'un cas limite, présentant des singularités. Un exemple du problème des 8 reines est un cas spécifique tandis qu'un exemple de matrice à une colonne est un cas singulier. Toutes ces distinctions sur les exemples dépendent du problème, des capacités de raisonnement ainsi que de la facon dont l'exemple est exploité...

Selon le problème, la figure peut ainsi décrire un exemple ou constituer une donnée du problème. De plus, les Figures sont exploitées à des étapes différentes de la résolution d'un problème selon le rôle qui leur échoit. Par exemple, un rôle descriptif est compatible avec l'introduction des données du problèmes, la description d'une étape intermédiaire ou de la construction à effectuer, et une synthèse finale des résultats.

3.2.5. Moyens de traitement des Figures

Le traitement des informations visuelles relève d'une problématique différente de celui des opérations langagières (cf par ex. [Lindsay 80]). Par exemple, le travail de Jacques Bertin (1967) constitue une recherche imposante dans le domaine spécifique du graphique statique servant de support d'information. Nous séparons ici les ressources disponibles, la composition de la Figure, et son interprétation.

Une caractéristique des ressources utilisées est qu'elles sont fréquemment :
- implicites et mal codifiées,
  sinon elles seraient déjà mathématisées !
- limitées dans leur possibilité de formalisation :
  elles ne sont exploitables que si les procédés de bon sens qui guident leur utilisation sont modélisés.
- limitées dans leur champ d'application :
  l'exemple du diagramme de Venn décrit justement un problème concernant son utilisation ; au delà de trois ensembles, il faut imaginer d'autres procédés que des cercles à moins de renoncer à l'exhaustivité des cas représentés.
- employées de façon non homogène :
  l'exemple des coupes d'une pizza montre que le procédé de numérotation change lorsque la complexité augmente.

Un exemple de ressource, illustré en Annexes, concerne des capacités de perception permettant une reconnaissance des "formes symboliques" (formes souvent exprimées à l'aide de points de suspension). D'autres exemples de ressources sont le tracé d'une courbe fermée quelconque, l'écriture d'un élément d'une matrice, mais les ressources les plus complexes sont les ressources de synthèse comme un algorithme de reconnaissance et d'étiquetage des zone connexes du plan ou un algorithme de calcul de la disposition des points d'un graphe. Ces ressources de synthèse peuvent toutefois se limiter aux cas simples. L'ergonomie cognitive est néanmoins une contrainte qui s'ajoute à la diversité des ressources nécessaires et à celle de leur champ d'application.

Les choix de composition globale exploitent des ressources propres au type de Figure pour **traduire des intentions**. Par exemple, la mise en évidence d'une opération peut se faire :
- par exhibition de la séquence d'états, dans le cas des coupes d'une pizza ;
- par superposition, dans le cas des coups du jeu du professeur Rufus Isaacs et dans l'exemple de la cycloïde ;
  par mise en valeur, puis isolement d'une forme, dans le cas de l'extraction d'un sous-graphe (base de cycle) ou celui de la contraction de deux graphes ;

Bien que cela nécessite un travail conceptuel important, il nous semble possible de modéliser point à point des ressources suffisantes pour fournir une boîte à outils. Ce long travail n'est possible qu'en abordant les Figures par catégorie, et à des fins d'utilisation pratique. Cet objectif étant relativement lointain, nous nous contentons ici de soulever d'autres difficultés à surmonter.

Une fois planifiée la décision d'effectuer une Figure et les intentions qui l'accompagnent, le premier choix intervient souvent pour composer une Figure ayant un rôle d'illustration. Prenons les définitions des graphes et de leurs propriétés déjà présentées en Annexes. Un graphe injectif générique n'existant pas, il s'agit d'exhiber un exemple spécifique présentant un compromis entre sa typicité et sa simplicité.

Cette composition de typicité se traduit souvent par l'analyse des formes usuelles compatibles avec la définition. Ces formes sont alors assemblées et présentées dans des situations aussi atypiques que possible. Une vérification globale est alors effectuée pour veiller en particulier à ce que l'ensemble n'ait pas de propriété trop restrictive susceptible de donner lieu à interprétation erronée. La typicité est donc aussi une caractérisation par négation.

Par exemple ici, le graphe injectif doit veiller à avoir des sommets de degrés différents car c'est une dimension d'analyse. En particulier, il doit impérativement éviter d'avoir le degré extérieur de tous les sommets égal à 1 car c'est une caractéristique importante pour l'analyse. De plus, l'absence de boucle est un signe caractéristique implicite de son impossibilité (les concepts qui les acceptent le montrent dans leurs exemples typiques). **La composition de la typicité traduit alors des intentions soumises à des lois d'interprétation implicites résultant de l'usage mathématique.** Ce développement de la typicité est intéressant car il est aussi valable pour les autres langages des mathématiques.

La composition d'une figure géométrique dépend de la faisabilité et de la validité de son interprétation. La difficulté de la composition est illustrée par l'énoncé suivant : "Prenons un triangle quelconque,...". Cette notion de quelconque, couramment utilisée en mathématiques, traduit en pratique l'impossibilité ultérieure de faire des choix spécifiques sur la nature du triangle. La difficulté de la construction de la figure géométrique est qu'un choix de triangle doit être effectué a priori, et qu'il ne doit pas interférer ultérieurement avec une propriété imposée par la construction. Il ne faut pas par exemple qu'un angle du triangle quelconque puisse correspondre à celui d'un angle indépendant, relativement aux approximations d'évaluation et de construction.

Cette **prise en compte de l'interprétation lors de la composition** est l'une des caractéristiques des langages de Figures lorsque leur présentation est suffisamment lâche. Elle culmine pour les représentations spatiales. C'est pourquoi nous classons par ordre de difficulté croissante les produits cartésiens discrets, les graphes de fonctions, les structures discrètes et les autres représentations spatiales.

L'interprétation des figures est à modéliser au cas par cas, par catégorie. Elle ne peut être complète sans une représentation utilisable par les langages de Phrases et Formules. C'est pourquoi nous proposons de différer l'interprétation des langages de Figures après celle des langages de Phrases et Formules. Nous ne choisissons donc pas la géométrie comme domaine d'étude et nous nous autorisons des exceptions pour les produits cartésiens discrets simples à modéliser.

3.3. LES ROLES DU LANGAGE POUR LE MATHEMATICIEN

3.3.1. Les langages de Phrases et de Formules

3.3.1.1. Rôle de l'écriture en mathématiques

> L'écrit n'est pas neutre en mathématique et bien au contraire les mathématiciens jouent très souvent de ses **ressources** aussi bien linéraires que spatiales, de ses **qualités** suggestives ou descriptives au risque d'**ambiguïtés**, tant pour exposer que pour investir de nouveaux territoires, intuitionner l'impensé et organiser l'impensable.

Ce paragraphe est la conclusion d'un texte de Jean Dhombres lors d'un séminaire de philosophie et mathématiques [Dhombres 81]. De nombreuses interrogations surgissent sur l'écrit et son traitement, interrogations auquelles il faut partiellement répondre pour spécifier un premier modèle de l'activité mathématique.

Les interrogations soulevées sont alors :

- Quelles sont les **ressources** du Langage Mathématique, comment les répertorier, les classifier, les implanter et surtout les exploiter ?
- Quelles sont les **qualités** suggestives ou descriptives, comment les préciser et s'affranchir de la compréhension littérale du langage, ou au minimum quels sont les choix possibles et ceux qui optimisent ces qualités dans une situation donnée ?
- Comment autoriser et gérer l'**ambiguïté**, à savoir quelles sont les composantes ambiguës du langage, qu'apporte l'ambiguïté au mathématicien et enfin quand et comment sera-t-elle éventuellement levée ?

- Comment l'écrit peut-il investir de nouveaux territoires, intuitionner l'impensé et organiser l'impensable ?

Ce dernier aspect rejoint les préoccupations épistémologiques et ne peut-être finement modélisé qu'après avoir répondu aux questions précédentes. Pour y répondre sommairement, il semble approprié de laisser jouer l'intuition du mathématicien pour décider au mieux de ses choix pratiques. Une validation expérimentale étudiant les motivations de ces choix et comparant leur pertinence semble pouvoir guider en retour les mathématiciens.

Le langage est souvent abordé dans une perspective phylogénétique pour évaluer sa contribution à l'édifice mathématique et à l'activité mathématique. Cette approche s'intéresse par exemple à l'évolution des notations et aux progrès mathématiques qui les ont accompagnés.

Jean Dhombres distingue et illustre ainsi plusieurs **rôles de l'écriture mathématique** :

- un rôle par défaut qui est l'écriture de l'impensable et provient de la généralité latente de l'écriture utilisée ;
- un rôle par absence ou ambiguïté (tantôt trompeur, tantôt créatif) ;
- un rôle par présence ;
- un rôle par analogie.

L'écriture axiomatique et l'écriture algorithmique (ou calcul symbolique) jouent leur rôle par présence. Jean Dhombres note alors que l'écriture ne sert plus qu'à distinguer :

> En perdant toute référence ontologique, l'écriture mathématique semble avoir également perdu les critères du choix des objets à étudier, des relations à établir.

Cette réflexion rejoint celle de François Lo Jacomo qui définit le langage par rapport à l'édifice mathématique [Lo Jacomo 75]. Le langage n'est alors qu'un tas de pierre par opposition à un édifice construit : il manque le principe organisateur ! A la recherche des principes premiers du langage, l'approche linguistique de François Lo Jacomo retrouve notamment le rôle par analogie identifié par Jean Dhombres : les signifiants (mots) sont choisis afin de favoriser l'analogie entre leur signifiés (sens). Ainsi, les mots "point", "droite", "espace" et "distance" sont utilisés dans de nombreuses théories avec des sens très différents.

Il semble alors évident que le langage mathématique a évolué de façon à servir l'activité mathématique, non seulement comme support d'expression d'une représentation conceptuelle, mais aussi comme acteur dans les processus cognitifs constituant l'activité mathématique.

3.3.1.2. L'ergonomie du langage

> It is a profoundly erroneous truism repeated by all copybooks and by eminent people when they are making speeches that we should cultivate the habit of thinking what we are doing. The precise opposite is the case. Civilization advances by extending the number of important operations which we can perform without thinking about them. Operations of thought are like cavalry charges in a battle — they are strictly limited in number, they require fresh horses, and must only be made at decisive moments.
>
> A.N. Whitehead[2]

Cette déclaration explique l'un des apports de la formalisation du Langage Mathématique à l'aide de langages de Formules. Les manipulations symboliques des Formules nécessitent une attention réduite pour conduire au but recherché. Mais cet outil de pensée ne s'est pas imposé a priori. Les notations actuellement retenues représentent un bon compromis pour l'usage du mathématicien.

Nous avons regroupé sous ce thème les facteurs visant à améliorer le langage lui-même. Le livre [Graham 89] est d'ailleurs particulièrement intéressant pour l'attention qu'il consacre aux définitions et aux idées qui les accompagnent. Nous n'abordons pas ici les techniques visant à faciliter l'utilisation du langage dans l'activité, et celles visant à obtenir des résultats aussi généraux que nécessaires.

---

[2] Cité dans [Graham 89, p 489].

Nous voulons explorer les facteurs qui interviennent dans l'ergonomie du langage. Notre objectif est triple :

- savoir adapter le langage vers une activité assistée par informatique ;
- justifier indirectement l'importance de notations et définitions variées et adaptables dans l'activité mathématique ;
- comprendre pourquoi les choix simplificateurs des systèmes informatiques se révèlent déficients ;

Nous avons recensé des **facteurs intervenant dans l'ergonomie du langage** :

- les matériaux élémentaires, comme le choix des notations et définitions ;
- les facilités d'expression de relations composées ;
- la présentation visuelle (typographie, graphisme, etc) ;
- la pertinence du langage par rapport à la phase de recherche de la résolution de problèmes ;
- la facilité d'utilisation et de mémorisation par son processeur de traitement.

Pour résumer ceci, la définition d'un langage est un compromis entre **expressivité, homogénéité de manipulation et aspects mnémoniques**. L'expressivité peut être envisagée par analogie avec les trois phases de la résolution de problème interactive : la facilité de formulation, l'utilité des relations mathématiques où elle intervient, et surtout la concision de sa présentation.

La facilité de formulation est l'un des critères qui justifie l'introduction de n! au lieu de se contenter du produit de ses facteurs. Elle impose aussi une clarté conceptuelle refusant par exemple les "excès d'algébrisme" que l'on retrouve dans le langage APL. APL a en particulier le mérite de découler de réflexions de Keneth Iverson [Iverson 79] sur l'usage d'un langage.

L'utilité de $x^0=1$ dans la formule du binôme est utilisée dans [Graham 89, p 162] pour justifier ce choix au détriment de 0 (limite de 0^x) ou indéfini. Une définition peu utile est reléguée. L'analyse du recueil de Cajory sur l'histoire des notations permettrait par exemple de pouvoir évaluer notre postulat darwinien : les notations (et définitions) peu utiles ou rendues caduques disparaissent.

La **concision de la présentation** est un aspect essentiel de l'ergonomie d'un langage. La principale concision des mathématiques est de disposer de plusieurs niveaux de description permettant le non-dit implicite et le flou temporaire. Les mécanismes de cette concision sont à notre connaissance inexplorés et inexploités dans les systèmes informatiques. L'effet essentiel de la recherche de concision en mathématiques est le passage progressif des langages de Phrases aux langages de Formules. La concision se traduit aussi en langage de Phrases par l'introduction de concepts abréviateurs (tel un Espace Vectoriel Topologique).

Nous qualifions de *concision sémantique* la concision apportée par les facteurs précédents car ils agissent principalement sur la nature des mathématiques. Mais la concision dépend aussi de la minutie des efforts consentis sur des aspects plus élémentaires. Par contraste, nous qualifions ce type de concision de *concision syntaxique*. Ici encore, les systèmes informatiques usuels sont souvent déficients.

L'introduction d'une nouvelle définition s'accompagne d'une recherche de concision et de généralité au détriment d'une surcharge supplémentaire du langage et de son traitement. L'introduction de la notation O(x) pour faciliter les manipulations d'analyse asymptotique est exemplaire. Son intérêt et l'emploi du signe "=" pour une telle égalité approximative sont notamment discutés dans [Graham 89, p 429+].

Plus simplement, le fait de pouvoir grouper plusieurs propriétés dans une même phrase est une facilité que beaucoup de systèmes n'offrent pas (soit f une fonction continue sur D et dérivable presque partout). Enfin, les procédés de désignation multiples et les marqueurs d'inférences succints et codifiés, évitent un verbalisme excessif dans la conduite du raisonnement.

La concision de la présentation ne se borne pas à l'introduction de nouvelles définitions, mais à une exploitation judicieuse des moyens disponibles. Par exemple R. Godement p 83 note 2.3=1 l'identité dans Z/5Z alors qu'il vient juste d'introduire la notation $2.3\equiv 1 \pmod 5$! Ainsi, cette notation ne se justifie que dans un contexte où une notation plus concise serait insuffisante car ambiguë. Un autre exemple de présentation synthétique est l'utilisation d'accolades dans des égalités pour simplifier leur expression (cf. Annexes : c'est une forme masquée de tableau de valeurs).

L'homogénéité de manipulation consiste à **limiter les besoins de réflexion du mathématicien à l'essentiel.** Un langage bien conçu est comme une monture autonome qui guide le mathématicien vers sa destination. Les membres de Bourbaki utilisaient la métaphore "l'âne qui trotte" pour qualifier ce type de démonstration où il suffit d'indiquer la voie. Le langage offre alors un cadre de pensée suffisamment bien organisé pour qu'il ne reste plus qu'à exploiter ses propriétés.

Le propre de telles progressions est de créer des automatismes. Il est bon pour cela d'utiliser des règles aussi uniformes et générales que possibles. Il est nécessaire aussi de limiter dans cette progression le nombre de conditions à vérifier ainsi que leur difficulté de vérification.

Un facteur qui nous semble essentiel est le **degré d'intégration des Phrases et Formules.** L'étendue des idées exprimables dans les langages de Phrases les rend mieux aptes aux fonctions descriptives, tandis que la conc on des langages de formules favorise leur usage évaluatif. Rien de tel alors, pour briser l'esprit d'une preuve, que d'interrompre en permanence les calculs par des argumentations auxiliaires. De plus, la présentation condensée autour d'une formule est plus expressive (cf Annexes).

Un autre facteur essentiel est paradoxalement de limiter les calculs aux opérations indispensables. Par exemple exprimer "$0<2k+1\leq p$" en fonction de k dans une sommation alors que cette relation va être modifiée à l'étape suivante. De même, ajouter des termes nuls dans les sommations est utile lorsque cela permet de considérer des intervalles plus homogènes. Les manipulations deviennent plus **homogènes.**

Ces deux facteurs sont illustrés par la convention de notation utilisée dans [Graham 89] et empruntée à K. Iverson. Il s'agit d'intégrer des prédicats dans les formules en ne considérant la valeur de la formule que lorsque les prédicats sont tous vérifiés (et égaux à 1, cf. Annexes). Cette convention permet notamment des manipulations d'indices multiples plus aisées. Ainsi, la somme des inverses des nombres premiers inférieurs à n se note :

$$\sum_{p} (p \text{ premier}) (p \leq n) / p$$

Les **aspects mnémoniques** sont le dernier critère recensé pour l'ergonomie d'un langage. Nous y retrouvons l'influence de l'homogénéité dans les considérations esthétiques. Il est ainsi facile de mémoriser et d'utiliser des relations symétriques ou des dualités comme celles entre les fonctions sinus et cosinus, min et max, pgcd et ppcm, etc.

Ces aspects mnémoniques ne sont pas seulement subis, ils sont recherchés. Les auteurs de [Graham 89] ont ainsi choisi leurs conventions de façon à ce que les formules de dérivation et d'intégration discrètes soient aussi proches que possible de leurs équivalents continus.

De plus, la terminologie est souvent employée pour faciliter la catégorisation (par ex. le qualificatif "algébrique") ou les analogies (par ex. l'usage surchargé des mots "points" "droite" et "espace"). Elle peut jouer aussi un rôle mnémonique supplémentaire en traduisant la cause ou l'effet[3]. Le choix de la terminologie est parfois **crucial pour le sens et le guidage des calculs** dans un problème issu d'un problème externe aux mathématiques. La terminologie est alors l'un des points qui favorisent l'intuition du mathématicien.

---

[3] par ex. $\binom{n}{r} = \binom{n}{n-r}$ est qualifiée dans [Graham 89] de "symétrie", une autre identité de "absorbtion".

3.3.1.3. Notre analyse du langage

Un bon artisan a toujours de bons outils.

Cette maxime s'applique étonnemment bien au langage du mathématicien. Il s'adapte au problème à traiter et s'impose alors comme outil universel. Le Langage Mathématique a peu de choses en commun avec un langage logique, et ses produits ne ressemblent pas à des preuves formelles. Pourtant, les tentatives de rationaliser le Langage Mathématique ne manquent pas : s'il avait dû être remplacé par des formalismes plus "logiques", ce serait déjà fait ! Le Langage Mathématique s'est simplement contenté d'intégrer les apports utiles de la formalisation. L'explication de cette résistance qu'il oppose aux mutations formalistes est simple :

le Langage Mathématique favorise la résolution de problèmes

par un mathématicien.

Notre analyse traite l'adéquation du langage à son "processeur" de traitement : le mathématicien. Nous reprenons les trois facettes du modèle mental du mathématicien : perceptif, créatif et cognitif, en y ajoutant le point de vue holistique de la production et la résolution de problèmes mathématiques. Nous isolons deux interrogations pour chaque thème, la seconde étant dans un rapport méta avec la première.

Vu l'ampleur des points à aborder, notre contribution introductive se focalise sur des points encore peu explorés dans la littérature. Les questions abordées sont :

Question 1 :	comment le mathématicien utilise-t-il le niveau élémentaire du langage ?
Question 1bis :	quelle influence ont les caractéristiques élémentaires du langage ?
Question 2 :	comment le mathématicien écrit-il une démonstration ?
Question 2bis :	comment le mathématicien construit-il une démonstration ?
Question 3 :	comment sont organisées les connaissances utilisées par le mathématicien ?
Question 3bis :	dans quel but les connaissances sont-elles ainsi organisées ?
Question 4 :	comment les problèmes mathématiques sont-ils produits ?
Question 4bis :	comment le mathématicien trouve-il une démonstration ?

Des réponses partielles ont été apportées à chacune de ces questions, mais aucune action systématique n'a été entreprise (ni à notre connaissance dans la littérature). Une linguistique des mathématiques serait d'ailleurs nécessaire pour en explorer plus avant les mécanismes...

Cette analyse porte à la fois sur les symboles et la syntaxe utilisée pour les formules, sur le procédé de construction des preuves, et sur la façon de rédiger une démonstration. Ce faisant, nous nous sommes posés des questions dont les réponses apportent un élément original dans la connaissance de la pratique du Langage Mathématique.

Le principal résultat s'il en est, est une prise de conscience des lacunes et des besoins de connaissance sur l'activité mathématique et ses productions. Ce résultat pourrait sembler paradoxal s'il n'existait pas, à notre connaissance, de travail de synthèse sur une approche cognitive de l'activité mathématique. Les travaux dont nous disposons se répartissent en effet en deux catégories : des travaux informatiques qui abordent une classe de problèmes précis, en mettant néanmoins en valeur la complexité de la tâche, et des travaux issus d'autres approches cognitives qui font un bilan assez complet d'un thème ponctuel, souvent d'ailleurs sans aucune intention de modélisation informatique.

3.3.2. Les processus perceptifs

3.3.2.1. L'utilisation des symboles favorise la perception

> Les expressions mathématiques sont des objets abstraits résultant de notations complexes et concises, auxquelles se rattachent des savoir-faire liés en particulier à la perception visuelle.
>
> Monique Baron [Baron 85]

Notre première question est :

Question 1 : comment le mathématicien utilise-t-il le niveau élémentaire du langage ?

Afin de visualiser l'importance de l'utilisation des symboles en mathématiques, nous avons choisi de la comparer avec des textes faisant des mathématiques dans une approche de vérification de preuve logique. Ces textes présentent l'avantage de fournir des exemples concrets de formules mathématiques où l'utilisation des symboles est d'une importance secondaire a priori.

Le problème traité maintenant est celui de "l'existence d'un symétrique" à partir d'une définition minimale des lois de groupe. Il s'agit de montrer qu'un inverse à droite est aussi inverse à gauche pour une loi associative.

L'énoncé du problème est celui du système MAD, un système d'aide interactive à la construction de preuves en logique du premier ordre [Fallot 89]. Le symbole de fonction prédéfini 0 étant défini au préalable comme le symbole de l'élément neutre de la fonction +, la présentation du système est :

$\forall l_1, \forall l_2, \forall l_3, (l_1+l_2)+l_3=(l_1+l_2)+l_3$ .....(1)
$\forall l_1, l_1=l_1+0 \wedge 0+l_1=l_1$ .....(2)
$\forall l_1, \exists l_2, l_1+l_2=0$ .....(3)
on introduit alors la formule de logique du premier ordre à démontrer :
$\forall l_1, \exists l_2, l_1+l_2=0 \wedge l_2+l_1=0$.....(4)

Ce qui frappe le mathématicien au premier abord, c'est le choix uniforme des symboles de variable. Si l'édition moderne permet de visualiser "l_1" à la place de "l1", l'approche logique favorise une stratégie de notation uniforme puisqu'il ne s'agit que du "sucre syntaxique". Le choix effectué pour le système MAD est de nommer ses variables liées "l_i" et de les numéroter dans l'ordre de quantification. D'autres symboles x,y,z,t... ont été déclarés a priori et sont utilisés comme symboles de variables libres apparaissant dans les formules. Leur portée est alors définie pour un ensemble de formules regroupées au sein d'une dérivation syntaxiquement explicite.

Le choix des symboles mathématiques constitue la première distinction marquante entre un mathématicien et un système logique. Les choix de symbole d'un système logique se veulent universels y compris dans les notations, tandis que les choix des mathématiques se veulent pratiques et orientés vers l'homme. Le libre choix des symboles mathématiques - et plus généralement des notations et autres procédés syntaxiques - sert ainsi les mathématiciens et se voit donc soumis aux règles, usages et préférences individuelles ou communautaires.

L'écriture mathématique visualise mieux

les facteurs pertinents et les invariants du problème.

Les mathématiciens ont développé des **stratégies de notation** permettant de favoriser la reconnaissance et l'utilisation des symboles. Cette stratégie utilise la liberté de choix en fonction de l'usage du symbole.

Ainsi, lors de la lecture d'un symbole le mathématicien cherche à en identifier la forme syntaxique et à lui associer un rôle dans la formule. Ce rôle n'est pas forcément lié à son interprétation mathématique (par ex. des manipulations de variables d'indice d'une double sommation portent plus sur un jeu d'écritures que sur le contenu mathématique). Les critères à rechercher sont donc des critères relatifs aux objets et à la tâche que l'agent doit effectuer, et secondairement relatifs à l'ordre d'apparition.

3.3.2.2. L'influence des choix de notation

Les réflexions précédentes nous amènent à l'interrogation suivante :

Question 1bis : quelle influence ont les caractéristiques élémentaires du langage ?

Pour illustrer cette influence, nous allons comparer l'effet de deux sortes de choix à partir d'un exemple. Le premier consiste à choisir ou non des symboles adaptés : x,y,z au lieu de l_1,l_2,l_3. Le second consiste soit à respecter l'ordre d'apparition des variables dans la formule (nommage logique) soit au contraire à privilégier l'ordre d'association des variables à l'intérieur du prédicat (nommage naturel). Le nommage naturel permet un nommage des variables correspondant à la lecture du prédicat. Il procure un aspect général relativement indépendant d'une formule. Nous obtenons ainsi quatre cas.

Pour illustrer ces choix, nous avons présenté en quatre exemplaires une démonstration issue de MAD. Cette démonstration forme une succession de trois formules dont chacune se déduit de la précédente. Nous obtenons alors :

stratégie / choix des symboles	naturelle	logique
adapté	$\exists x, \forall y, \forall z, q(x,y,z)$ $\forall y, \exists x, \forall z, q(x,y,z)$ $\forall y, \forall z, \exists x, q(x,y,z)$	$\exists x, \forall y, \forall z, q(x,y,z)$ $\forall x, \exists y, \forall z, q(y,x,z)$ $\forall x, \forall y, \exists z, q(z,x,y)$
non adapté	$\exists l_1, \forall l_2, \forall l_3, q(l_1,l_2,l_3)$ $\forall l_2, \exists l_1, \forall l_3, q(l_1,l_2,l_3)$ $\forall l_2, \forall l_3, \exists l_1, q(l_1,l_2,l_3)$	$\exists l_1, \forall l_2, \forall l_3, q(l_1,l_2,l_3)$ $\forall l_1, \exists l_2, \forall l_3, q(l_2,l_1,l_3)$ $\forall l_1, \forall l_2, \exists l_3, q(l_3,l_1,l_2)$

Le théorème résultat est obtenu à partir de la première et de la dernière formule. La démonstration logique aux symboles non adaptés est la démonstration avec MAD du théorème logique suivant :

$$\exists l_1, \forall l_2, \forall l_3, q(l_1,l_2,l_3) \Rightarrow \forall l_1, \forall l_2, \exists l_3, q(l_3,l_1,l_2)$$

La présentation mathématique correspond aux symboles adaptés avec une stratégie de notation naturelle. Elle correspond à une autre présentation du théorème :

$$\exists x, \forall y, \forall z, q(x,y,z) \Rightarrow \forall y, \forall z, \exists x, q(x,y,z)$$

Bien que la densité d'information rende la lecture de chacune des cases peu engageante, la présentation mathématique permet d'arriver plus rapidement à la conclusion que l'usager a "permuté les quantificateurs en laissant le reste inchangé". Le simple fait de multiplier les symboles par un choix non adapté ralentit cette identification. De même, les symboles n'acquièrent leur signification qu'en fonction de leur rôle dans l'expression. La stratégie de présentation logique rend le rôle d'un symbole incertain à la lecture du quantificateur. La stratégie mathématique permet de connaître le rôle d'un symbole dès la lecture du quantificateur. La qualité du nommage fait la différence.

Une étude expérimentale des déplacements visuels et des temps de réponse pour la reconnaissance de formules donnerait des résultats intéressants pour la connaissance des mécanismes de compréhension. Elle permettrait de valider un modèle de lecture qui intègre les informations implicites du mathématicien.

3.3.2.3. Les stratégies macroscopiques d'utilisation des notations

Afin d'aller plus avant dans notre réponse aux questions "élémentaires", il nous faut maintenant introduire la vue macroscopique du Langage Mathématique : nous avons écrit indépendamment la preuve du problème de "l'existence d'un symétrique" et voici la façon dont le brouillon peut être transcrit :

soit $x \in G$, il existe x^d tel que $x\ x^d = 1$
et il existe aussi x^{dd} tel que $x^d\ x^{dd} = 1$
$(x\ x^d)\ x^{dd} = 1\ x^{dd} = x^{dd}$ .....(1)
or (1) est aussi égal à
$x\ (x^d\ x^{dd}) = x\ 1 = x$ .....(2)
on a donc $x^{dd} = x$ .....(3)
soit $\forall x$, $x^{dd} = x$.....(4)
par définition de x^d et x^{dd} cela se reformule en
$\forall x, \exists x^d, (x\ x^d = 1) \wedge (x^d\ x = 1)$.....(5)

Le mathématicien utilise les conventions pour faciliter ses manipulations.

Pour cette démonstration, nous avons utilisé une notation multiplicative pour la possibilité d'omission de l'opération interne. Cela est de plus conforme à un usage mathématique qui veut que « la notation additive s'emploie uniquement pour les lois de composition commutatives » [Godement 66, p 107].

De même, nous avons choisi "x^d" tel que "$x\ x^d = 1$" afin de matérialiser la dépendance fonctionnelle vis-à-vis de "x" : en effet, "x^d" joue le rôle d'inverse à droite de x et son existence est conditionnée par celle de x. Cette notation n'a d'intérêt que parce qu'elle sied au problème présent et qu'elle s'étend de façon satisfaisante en "x^{dd}".

Lorsque l'on aboutit à (3) : "$x^{dd} = x$", le sens pratique de ce résultat se retrouve plus facilement que si l'on avait "$z = x$". Sans aller chercher le sens de z, le mathématicien voit "$(x^d)^d = x$", il interprète que l'inverse à droite de x^d vaut x, et le reformule donc en "$x^d\ x = 1$". Le propre d'une bonne notation est de faciliter les transferts entre représentations de façon à bénéficier des avantages de chacunes. Une notation permet donc au mathématicien de se constituer une représentation plus aisée à manipuler que les phrases qu'elle symbolise.

Une notation est dépendante de ce qu'elle suggère et de l'usage qui en est fait.

Une notation est utile pour un certain usage. L'intérêt de son usage constitue sa limite naturelle. Ainsi, l'introduction d'une notation comme "x^d" ne se justifie pas pour simplement exprimer le théorème résultat, car il met en valeur la dépendance au détriment d'une appartenance générique au groupe. De même, la notation vectorielle $\overrightarrow{AB}$ est utile lorsque le mathématicien manipule un représentant particulier du vecteur. Elle s'abstrait en $\vec{u}$ pour désigner le vecteur lui-même, et se simplifie en u lorsque la distinction entre scalaires et vecteur n'a plus de sens.

Le choix d'une notation peut introduire des présupposés absents du problème. Par exemple, nous avons adopté lors d'une étude ultérieure la notation (g,c,d) au lieu de (x,x^d,x^{dd}). Nous voulions ainsi marquer la symétrie de rôle autour d'un élément central. Cette symétrie implicite suppose que l'existence d'un inverse à droite et à gauche dépend de celle de c. Cela est faux et s'est avéré trompeur dans notre problème.

Les critères de choix des notations ne sont ni recensés, ni modélisés à notre connaissance. Pour ce qui est des symboles, les objets couramment manipulés dans la théorie ont souvent leurs lettres réservées dans un police de caractères. Le problème à modéliser est alors plutôt celui de la résolution des conflits et des changements de contexte. Lorsque l'alphabet le permet, les symboles ont aussi des connotations mnémoniques comme G et H pour les groupes. De même, la célèbre formule d'Euler sur les polyèdres : "S - A + F = 2" se reconstitue aisément en cherchant les objets concernés : les sommets, les arêtes et les faces. Cette facilité de perception est essentielle dans les manipulations de formules.

Les principes ergonomques et les critères de choix des notations composées se résument par la recherche du principe :

L'identité syntaxique implique l'identité mathématique.

Cette recherche d'identité syntaxique est ainsi à la base de l'usage et du développement des notations mathématiques. Ce principe explique notamment pourquoi la notation f(x) n'est pas utilisée pour les relations car elle entraînerait une ambiguïté intrinsèque. Ce principe se raffine en : "deux sous-expressions syntaxiquement identiques ont la même interprétation mathématique si elle existe à condition que leurs contextes respectifs ne diffèrent en rien pour ce qui les concerne".

Le corollaire de ce principe est qu'un système est syntaxiquement valide si deux objets mathématiques différents - ou plutôt deux objets mathématiques qu'il est opportun de distinguer - ne peuvent avoir une forme syntaxique commune. Par exemple "A×B×C" sera utilisé en lieu et place de "(A×B)×C" et "A×(B×C)" sauf si un besoin précis nécessite de distinguer ce niveau de détail.

On obtient ainsi en mathématiques une abstraction ou ambiguïté volontaire, similaire aux ambiguïtés de "+" dans "2+3", qui est par défaut compris comme l'addition entière sachant effectuer une conversion ultérieure en addition réelle si cela est pertinent. Il y a donc un phénomène de polymorphisme inhérent aux mathématiques.

3.3.3. Les processus créatifs dans une démonstration

3.3.3.1. L'écriture des démonstrations

> Lorsqu'il s'agit de mathématique, les concepts de "raisonnement" et de "démonstration" entretiennent manifestement des rapports étroits.

Cette citation de Daniel Lacombe introduit clairement la pertinence d'une étude des démonstrations quant à l'étude du raisonnement. Le premier problème est alors celui de la nature des démonstrations produites par le mathématicien. C'est l'objet de notre question :

Question 2 : comment le mathématicien écrit-il une démonstration ?

La logique propose une réponse indirecte à cette question. Elle a formalisé les preuves de façon à traduire les opérations du mathématicien grâce à des systèmes dits de "déduction naturelle". Cette formalisation est censée modéliser les procédés de manipulation des hypothèses et des connecteurs logiques. Une démonstration est alors la présentation exhaustive ou holophrastée d'une telle preuve (la distinction n'a guère d'intérêt en logique...).

Le système MAD qui nous sert de référence est ainsi novateur par son utilisation de la déduction naturelle. De plus, son approche est comparable à celle que nous cherchons puisqu'il s'agit d'un système d'aide interactive à la construction de preuves (en logique du premier ordre). En effet, même les preuves simples ont besoin d'un guidage humain car la complexité d'une démonstration automatique serait souvent trop importante.

L'écriture de la preuve de "l'existence d'un symétrique" occupe une page entière et son abord difficile dissuade une lecture attentive. De plus la longueur de sa construction est aussi dépendante de celle du résultat final malgré les aides que l'on peut fournir.

Deux solutions sont possibles : trouver un langage de commande qui exprime plus abstraitement les actions à effectuer, ou adopter directement un formalisme de travail plus approprié. Le Langage Mathématique joue sur ces deux plans.

Tout usager est alors confronté au problème de faisabilité d'une démonstration en logique du premier ordre. L'approche de MAD est prometteuse, mais nous pensons que le choix d'un formalisme logique est la source des difficultés qu'éprouve l'homme chargé du guidage heuristique.

Nous pouvons ainsi analyser une démonstration écrite par un mathématicien comparativement à une démonstration proposée selon un formalisme logique se voulant "naturel". La démonstration mathématique précédemment présentée se targue d'être plus lisible et plus expressive que son homologue logique. S'il fallait la résumer, nous dirions qu'elle va à l'essentiel.

Les deux premières lignes introduisent les objets en présence (x, x^d et x^{dd}) et leurs définitions ; les points (1) à (2) exploitent leurs propriétés et le point (3) récapitule le résultat obtenu. Ce dernier est généralisé en (4), qui est à son tour rendu indépendant des objets précédemment introduits en y intégrant leurs définitions ; le résultat (5) correspond au théorème à montrer.

Ainsi, s'il y a une structure macroscopique dans une démonstration, c'est plutôt un ensemble de principes de présentation. Ces principes respectent les critères d'entendement humains et la présentation évolue avec les centres d'intérêt d'un mathématicien qui aurait choisi la même démarche heuristique. Quels que soient ces principes, un invariant peut être dégagé :

L'écriture d'une démonstration simule des opérations mathématiques.

3.3.3.2. La construction des démonstrations

Pour savoir comment écrire une démonstration, et quelle peut-être sa structure macroscopique, il faut donc savoir comment elle est construite. D'où la question plus générale :

Question 2bis : comment le mathématicien construit-il une démonstration ?

Les études de brouillon ont été notamment utilisées par Dominique Pastre [Pastre 78] et Alan Schoenfeld [Schoenfeld 85] pour l'observation des activité cognitives du mathématicien. Pour cela, voyons ce que nous apprend l'étude du brouillon, plus sensible à l'activité de raisonnement et de surcroît révélatrice de la construction de notre démonstration.

L'introduction des symboles et des notations ainsi que les justifications restent souvent inexprimées, tandis que la suite des formules matérialisent les points d'arrêt mentaux. Nos observations confirment que le mathématicien explicite l'introduction des symboles et notations lorsque les informations associées sont utiles à la compréhension de la suite de la démonstration.

De même, la rédaction d'une démonstration à partir du brouillon s'accompagne de l'ajout de justifications. Ces justifications sont l'objet de transitions mentales souvent conscientes mais omises pour la lenteur de leur rédaction. On retrouve néanmoins dans les brouillons des graffitis indiquant les liens d'une justification importante (e.g. la transition de (1) à (2) est présente sur le brouillon par "=").

Les points importants de l'énoncé sont notés dans la représentation opérative appropriée :

- les données du problème s'écrivent avec les concepts qui les résument (ici un rappel en trois points superposés "assoc", "élt neutre gche + drte 0"[4], "existence invdrte") ;
- la propriété comme "x^d est l'inverse à droite de x" se traduit par "x x^d = 1" car cela fixe les symboles et l'emploi qui en sera fait.

Dès sa conception, la démonstration fait ainsi intervenir un jeu de possibilités et de nuances qui ne s'expliquent que par la façon dont le mathématicien a pensé initialement la solution de son problème. Cela n'a qu'un rapport lointain avec la stratégie de recherche (dite *heuristique*) qui aboutit à cette solution : ce qui guide l'écriture d'une démonstration est principalement la façon dont le mathématicien effectue les manipulations des données pour construire sa solution.

Les démonstrations sont adaptées aux opérations des mathématiciens.

---

[4] Il s'agit ici de l'analyse des lois présentée dans le texte de MAD. La décision d'adopter une notation multiplicative a été prise ultérieurement.

3.3.3.3. Des manipulations d'objets adaptées au mathématicien

Ici encore, l'observation confirme que les procédés mathématiques employés sont plus nombreux et plus riches que ceux disponibles en logique, même lorsque cette logique utilise la variante assouplie baptisée déduction naturelle. La manipulation des quantificateurs en est l'exemple type.

Dans sa quête de formalisation, la logique a rendues fastidieuses des tâches empiriquement simples. La manipulation des expressions quantifiées s'accompagne ainsi de tout un protocole syntaxique destiné à expliciter et garantir une manipulation correcte des symboles de variables. Ainsi pour éliminer un quantificateur le système MAD introduit une dérivation avec un début et une fin syntaxiquement explicites. Par exemple : "$\forall l_1, \exists l_2, l_1+l_2=0$" donne "$\exists l_1, x+l_1=0$" puis "x+y=0" par introduction successive de deux blocs de dérivation introduisant "x" et "y".

En fait, l'usage mathématique explicite rarement l'élimination des quantificateurs et se contente d'introduire directement la formule (ici "x+y=0") afin de détailler les manipulations ultérieures jugées intéressantes. Cette élimination des quantificateurs se fait ainsi en bloc pour une formule, et non quantificateur après quantificateur. Elle se traduit par des dépendances (ou indépendances) fonctionnelles entre les variables introduites :

soit $x \in G$, il existe x^d tel que $x\ x^d = 1$
et il existe aussi x^{dd} tel que $x^d\ x^{dd} = 1$
$(x\ x^d)\ x^{dd} = 1\ x^{dd} = x^{dd}$ .....(1)

x^{dd} est dépendant de x^d qui est dépendant de x ; 1 est indépendant du reste, et le tout est dépendant du groupe (G,X) et donc de ses constituants.

Ces dépendances s'introduisent toutefois en deux étapes si l'on doit instancier deux fois la même formule comme pour (x,x^d) et (x^d,x^{dd}) dans les deux premières lignes de la démonstration. Pour déterminer la portée de la dérivation dans une démonstration, seule l'introduction des symboles de variables est fréquemment marquée (soit x..., prenons x...). La nature du quantificateur éliminé, ou même le passage d'un symbole libre au même symbole lié (comme ici celui de (3) à (4) qui est ponctué par "on a donc") se font alors implicitement.

Une démonstration traduit la façon dont le mathématicien effectue

les manipulations des données pour construire sa solution.

Ces manipulations d'introduction d'objets sont considérées comme mineures et standard par les mathématiciens. Toutefois, elles ne sont pas simples à effectuer et engendrent parfois des erreurs. Un bon modèle du Langage Mathématique doit aussi expliquer ces erreurs. Par exemple, l'expression (4) nous a semblé satisfaisante pour conclure la problème en disant que l'inverse à gauche de x^d était identique à son inverse à droite.

soit $\forall x,\ x^{dd} = x$ .....(4)
par définition de x^d et x^{dd} cela se reformule en
$\forall x, \exists x^d, (x\ x^d = 1) \wedge (x^d\ x = 1)$ .....(5)

Nous omettions ainsi que l'existence de x^d n'impliquait pas celle de x et qu'il fallait interpréter (4) selon x. Cela nous a conduit à introduire l'expression (5) finale.

Les manipulations de variables observées dans notre problème permettent d'expliquer la confusion pédagogiquement fréquente entre propriété existentielle et propriété universelle. Nous nous sommes même fait piéger ici par cette ambiguïté. Le mathématicien manipule identiquement les variables existentielles et universelles dès l'élimination des quantificateurs : il se donne une variable par hypothèse et voit ce qu'il peut faire avec ! Les schémas suivants expliquent ceci :

"Si il existe x qu'on se donne, alors on peut dire cela "

Ce schéma est souvent utilisé dans les démonstrations par l'absurde, il ressemble fort au :

"Si je me donne x que j'ai le droit de choisir car il en existe un, ..."

De plus, la confusion est facilitée car ce dernier a une lecture trompeuse lorsqu'il est quantifié :

"Il existe x (et on aura le droit de se servir d'un tel objet par ailleurs)".

La manipulation d'objets existants vient ainsi se téléscoper parfois avec celle d'objets introduits par hypothèse.

La gestion des démonstrations par hypothèses est la caractéristique majeure des démonstrations naturelles. Pourtant, comme les autres opérations, elle n'est pas assez approfondie. Les déductions en logique dite naturelle ne donne qu'une vue atomique trop lourde de ces manipulations de contexte et ne savent pas bien gérer les dépendances entre objets.

3.3.3.4. Les traces de la démarche heuristique

Si la façon dont le mathématicien construit une démonstration conditionne ce qu'il écrit, il est normal de s'interroger sur les raisons qui lui permettent de trouver une démonstration. Ce point concerne la recherche heuristique déployée par le mathématicien dans son activité de raisonnement et il fait l'objet de la question 5 ultérieure. Pour ce qui concerne l'écriture d'une démonstration, signalons d'emblée que les aspects heuristiques ne forment pas l'objectif d'une preuve. Une démonstration gomme ainsi la plupart des aspects créatifs ou ne garde de façon anecdotique que les plus marquants. La majorité des considérations heuristiques se trouvent ainsi implicites dans la recherche d'une présentation linéaire.

La démonstration proposée ici débute par le point de démarrage écrit "(x x^d) x^{dd}" alors qu'une démonstration "achevée" gommerait cet indice heuristique en amalgammant (1) et (2) et en démarrant par "$x^{dd}=$". On entrevoit ainsi les contorsions que devront permettre un système de construction de preuve s'il veut à la fois aider à construire puis à présenter une démonstration.

Notons néanmoins que **la présentation d'une démonstration reproduit une démarche heuristique possible, une fois gommés les aléas de la recherche.** Dans notre exemple, le mathématicien peut partir du but à atteindre "$x^{dd} = x$" et le transformer pour chercher à rendre comparables et identiques ses deux membres. Plus élégamment ici, il peut partir du membre gauche "$x^{dd} =$" pour aboutir à "x". La démonstration proposée ici part d'une propriété à utiliser : l'associativité. Rien n'est dit explicitement cependant sur les motivations de ce choix, ni sur ce qui motive l'introduction de x,x^d et x^{dd}.

Lors de notre exploration de l'activité mathématique, nous avons cherché ce qui motivait l'introduction de "(x x^d) x^{dd} = 1 x^{dd} = x^{dd}" (1), après une tentative avortée de débuter par "(x^d x) (x^d x)^d = 1". C'est qu'il nous faut ici trois éléments dépendants minimum et qu'il n'y a qu'une seule façon de les générer, d'où (x,x^d,x^{dd}). Il est alors possible d'expliquer les démarches précédentes en explorant les manipulations potentielles de ces données.

Pour récapituler, même si ces considérations heuristiques n'interviennent guère dans l'écriture des démonstrations, cette écriture traduit néanmoins fortement la façon dont le mathématicien effectue ses manipulations.

3.3.4. Les processus cognitifs

Si un consensus se dégage sur la nature des connaissances de base en mathématiques, rien n'est disponible à notre connaissance sur le traitement de l'information par le mathématicien, notamment :

Question 3 : comment sont organisées les connaissances utilisées par le mathématicien ?

La logique propose un modèle axiomatique où les connaissances se ramènent à un ensemble de formules où les symboles de variables sont liés dans chaque formule. Elle manipule alors un ensemble d'objets disjoints. En mettant l'emphase sur les connecteurs logiques, la logique tend à favoriser un niveau d'abstraction trop faible. Les formules peuvent être qualifiées par des propriétés comme "existence et

unicité de l'élément neutre". Le mathématicien peut alors structurer les relations entre formules en nommant et en caractérisant leur différence, au lieu de mémoriser tel quel une formule comme :

$$\exists e, \{(\forall x, x \perp e = e \perp x = x) \wedge [\forall y, (\forall x, x \perp y = y \perp x = x) \Rightarrow y = e]\}$$

Alors que les logiciens conçoivent les symboles comme des symboles, les mathématiciens utilisent les symboles comme les représentants des objets qu'ils désignent. Cela renverse le point de vue des manipulations : au lieu d'être assujetti à une formule, un symbole devient porteur de sens, et les formules deviennent des attributs de ce symbole, ce qui a pour effet secondaire de supprimer la quantification correspondant à ces objets.

Ce modèle de traitement **regroupe alors les connaissances autour des objets introduits et des symboles qui les représentent.** Lorsqu'un objet est introduit, son existence est établie "sans condition" pour l'élément neutre, ou par un lien fonctionnel pour l'inverse. Cette introduction répond à un besoin qui peut survenir en permanence pour l'élément neutre, et dès qu'un élément est présent pour l'inverse. Ainsi en mathématiques, dès qu'un objet ou une relation utile est identifié, un symbole lui est attribué qui va permettre d'établir son existence et sa persistance.

De ce fait lors d'une session, un même objet sera désigné par le même symbole. Du point de vue pratique, cela évite de manipuler aussi la définition et les propriétés de l'objet à chaque fois qu'il intervient, car les propriétés sont rattachées à l'objet au lieu d'être dispersées dans les formules. Ce procédé permet au mathématicien, lorsqu'il détecte un symbole, d'associer ou d'énumérer mentalement l'ensemble des propriétés utilisables qui concernent sa fonction.

Ainsi, l'organisation des connaissances est fonction de leur utilisation, ce qui répond au corollaire de la question 3 :

Question 3bis : dans quel but les connaissances sont-elles ainsi organisées ?

La persistance des symboles (et objets) introduits fait aussi intervenir le modèle de traitement du mathématicien, mais cette fois par la structure de leur utilisation. Les informaticiens sont familiers à la notion d'environnement qui regroupe l'ensemble des choix de variables effectués à un instant donné (le point de vue du système ou du programme). Le mathématicien manipule quant à lui la notion de contexte qui regroupe l'ensemble des choix et des états présents dans l'esprit du mathématicien à l'endroit considéré par sa tâche courante.

L'introduction d'un nouveau symbole ou celle d'une structure de groupe sont des actions mémorisée dans le contexte, mais ces actions sont parfois contraintes par des dépendances. La loi de persistance des objets consiste à minimiser leurs dépendances afin que leur zone d'utilisation soit la plus étendue possible. Ces objets et leurs symboles sont alors **hiérarchisés en fonction des besoins de leur utilisation.** Cela détermine l'organisation des connaissances mathématiques.

Examinons par exemple le cas de la structure de groupe. Cette structure est issue d'un cadre ensembliste basé sur un ensemble et des éléments indéterminés. Elle hérite des notions ensemblistes et des conventions régissant la manipulation des ensembles et de leurs éléments. L'ensemble de base, la loi de composition et les trois propriétés de définition sont rattachées à cette notion de groupe car ils sont imposés par la structure de groupe. De même, les symboles et les noms qui servent à les désigner lui sont rattachées (avec des conventions de symboles par défaut). Tout cela est indispensable dans toute utilisation de la notion de groupe.

Considérons alors les divers éléments de l'ensemble de base et leurs relations. Tant que rien n'impose la nécessité de leur utilisation, ils n'ont pas de raison d'être associés à la notion de groupe. Toutefois, leur fréquence d'utilisation peut justifier un intérêt autre que particulier. Ainsi, l'existence d'un élément neutre permet d'introduire "la classe de tous les éléments neutres" auquelle pourra être associée si besoin une convention de notation pour elle et pour ses éléments.

Bien que cette classe puisse avoir une existence pour elle-même en dehors de la structure de groupe, elle revêt vis-à-vis de cette dernière des relations suffisamment étroites pour pouvoir lui être associée (dans une version spécifique à la structure de groupe). Lorsque le mathématicien montre l'unicité de l'élément neutre dans un groupe, il renforce l'intérêt de cette classe d'éléments (de cardinal indéterminé), et autorise alors sa substitution par un objet unique. La convention de notation éventuelle est ainsi remplacée par une notation unique d'un objet vérifiant une propriété renforcée.

Comme la classe, cet objet qui est ici un élément spécifique ("élément neutre" ou le symbole "e") est rattaché à la structure de groupe. Le choix d'une désignation globale à la structure permet alors de disposer de ses propriétés pour toute manipulation sans avoir à les réintroduire (notamment dans un théorème utilisant la structure de groupe).

De même, l'usage pratique montre l'intérêt de l'existence d'un élément associé (l'inverse), et suggère l'introduction d'une notation explicitant les dépendances (un symbole masque la dépendance). Ces symboles et conventions, cette notation de l'inverse et les propriétés génériques qui leur sont associées sont ainsi rattachés à la notion de groupe : ils sont automatiquement hérités dès l'introduction d'un groupe et font partie de l'organisation des connaissances.

L'analyse du langage telle qu'elle est présentée en logique ne correspond donc pas à celle qui apparaît lors de l'étude pratique du Langage Mathématique. **Il faut repenser l'analyse du langage en étudiant l'action du mathématicien lors de son activité mathématique.** Un modèle mental simple suffit, puisque l'essentiel consiste à adopter le point de vue du mathématicien au cours de sa résolution de problème. Vu les analyses précédentes, nous considérons donc que le mathématicien :

- adapte la zone paramathématique élémentaire du Langage Mathématique de façon à pouvoir exercer ses capacités perceptives et mnémoniques (le choix des symboles, des notations et leur accès direct aux connaissances qui les concerne) ;
- cherche à regrouper et organiser ses connaissances en fonction des besoins de leur utilisation (le concept de groupe se voit ainsi associé à un paquet de symboles et de conventions mathématiques et paramathématiques) ;
- organise la structure macroscopique du Langage Mathématique d'après la façon dont il effectue ses déductions et dont il manipule les données.

Si le mathématicien choisit les symboles et notations en fonction de la façon dont il manipule les expressions, c'est que **l'activité mentale du mathématicien conditionne son usage du Langage Mathématique.**

C'est pourquoi nous nous proposons d'explorer un problème voisin révélateur de l'influence sur le langage, des manipulations telles que le mathématicien les conçoit :

Montrer l'unicité de l'élément neutre d'un groupe défini par :	
$\forall x,y,z,\ (x \perp y) \perp z = x \perp (y \perp z)$ .....(1)	associativité
$\exists e,\ \forall x,\ x \perp e = e \perp x = x$.....(2)	existence d'un élément neutre e
$\forall x,\ \exists y,\ x \perp y = e$.....(3)	tout élément a un "inverse" à droite

Le choix de notations "adaptées" pose problème dès l'énoncé des lois du groupe, reformulées à partir des énoncés logiques. Les réponses que nous avons choisies viennent compléter l'exploration des questions 1 précédentes sur le niveau élémentaire. Nous avons choisi des symboles de variables qui respectent des conventions usuelles. Pour la loi de composition interne, nous avons repris la notation de Roger Godement explicitement choisie parce que ce symbole n'intervient nulle part ailleurs (en algèbre) [Godement 66, p 106]. Enfin, nous avons adopté le choix mathématique "$x \perp e = e \perp x = x$" qui évite le connecteur logique "$\wedge$" peu prisé des mathématiciens : cela regroupe les propriétés au lieu de les décomposer.

Un problème plus délicat est celui du choix de "$\forall x,y,z,$" au lieu d'un "$\forall x, \forall y, \forall z,$". Un simple abus de langage dira le logicien, juste toléré pour faciliter l'emploi d'un niveau inférieur. Pas du tout ! Ce procédé que le logicien considère comme un simple abus, est à la base des procédés de notation qu'emploie le mathématicien pour adapter son langage au niveau de travail le mieux approprié. Une fois prouvé l'équivalence, ce qui compte pour le mathématicien c'est le nouveau langage résultant ! Cela explique la difficulté ultérieure des mathématiques de retrouver la trace des présupposés, ou de pouvoir revenir à des opérations microscopiques. Ces diverses strates du langages sont en effet comprimées voire oubliées au profit des couches les plus externes.

La notation choisie a inévitablement une incidence sur les manipulations qui l'utilisent. Ici, ce procédé permet d'abstraire les lois de permutation des quantificateurs qui sont considérées comme peu intéressantes voire pertubatrices. Cela ne s'étend pas au-delà d'une zone de quantification dans notre exemple, mais cela rejoint le problème plus général de manipulations modulo la commutativité et l'associativité. Plus abstraitement, l'élimination de quantificateurs inutiles rétablit la symétrie intrinsèque des variables qui était indirectement capturée par un procédé de construction artificiel. Nous aurions pu choisir un "$\forall(x,y,z)$", rendant plus explicite la distinction entre ce niveau méta (celui de

la manipulation des formules) pour lequel les règles de manipulation des variables sont associatives, et le niveau "objet" où la loi considérée ne l'est pas forcément.

Ici comme pour les autres choix de notation, l'essentiel est de savoir quand et comment introduire un nom destiné à faciliter les désignations ultérieures, en sachant que cela ne se fait pas sans contrepartie...

3.3.5. Les processus créatifs dans l'activité de raisonnement

3.3.5.1. Les péripéties de la résolution de problèmes

Nous avons toutefois écarté jusqu'à présent les aspects macroscopiques les plus difficiles : ceux relatifs au problème lui-même et à sa résolution. L'énoncé de notre problème montre avec quelle précision est dosé son **contenu informatif.**

En mathématiques, un énoncé est défini par le point précis qu'il s'agit d'étudier, toutes choses étant connues par ailleurs. Dans ce problème de communication, les rares rappels consistent à compléter les informations éventuellement manquantes du destinataire et rappeler des choix spécifiques.

La démonstration de l'unicité de l'élément neutre s'écrit par exemple en se dérivant de la définition :

soient e et e' deux éléments neutres, on a :	
neutre(n,⊥) ≡ (∀x, x⊥n = n⊥x = x)	par rappel de définition
e'⊥e = e⊥e' = e'	pour n=e et x=e'
e⊥e' = e'⊥e = e	pour n=e' et x=e
ce qui donne finallement e=e' cqfd.	

En réalité, cette démonstration exploite le fait que si une loi admet un élément neutre à droite, et un élément neutre à gauche, cet élément est le même. Si e est le neutre à droite, cela se résume en e' = e'⊥e = e par exploitation des propriétés de e, puis de celles de e'. Cela démontre donc l'unicité d'un élément neutre à la fois à droite et à gauche.

Ici, l'énoncé considère que le destinataire connaît le contexte mathématique ensembliste nécessaire aux manipulations des groupes. Il rappelle seulement les propriétés de définition pour lever l'ambiguïté de définitions équivalentes et pour préciser les notations adoptées.

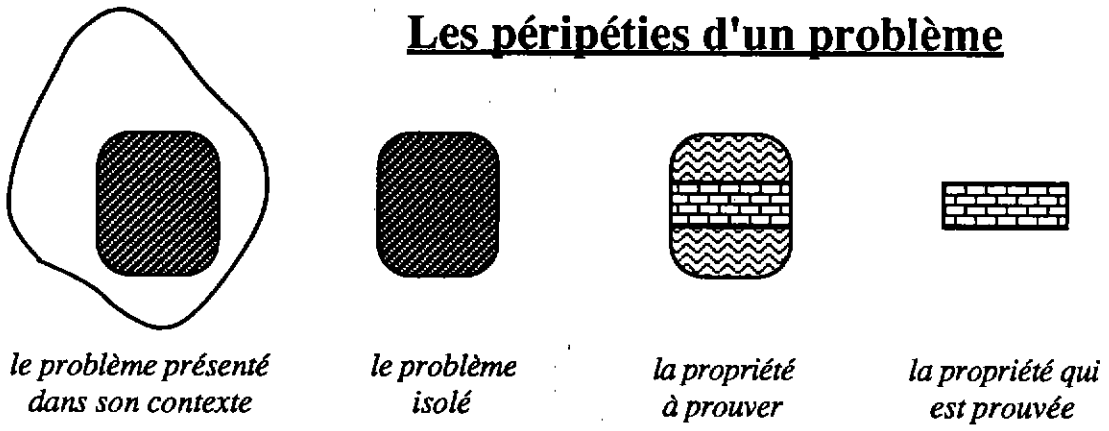


figure 1.12

Les diverses propriétés peuvent alors être analysées en fonction de leur contenu informatif :

- Le problème concerne ici la méta-propriété "unicité" qui indique qu'une propriété est exclusive c'est-à-dire qu'elle n'est vérifiée que par un élément au plus (une méta-propriété voisine est "l'existence").
- Le problème isolé consiste ainsi à prouver l'unicité de l'élément neutre, c'est-à-dire d'un élément "e" vérifiant la propriété "neutre(n,⊥)".
- La propriété à prouver comprend l'ensemble des informations utiles sur "e", c'est-à-dire mathématiquement ce qui sera retenu sur "e" une fois le problème résolu.

- La propriété qui est prouvée dans la démonstration de l'unicité est une forme de la méta-propriété "unicité" particularisée à la propriété "neutre(n,⊥)", particularisation qui fait ici allusion à l'existence de "e".

Cette propriété qui est prouvée bénéficie ainsi du schéma pratique de démonstration propre à la méta-propriété unicité : si deux symboles distincts vérifient la propriété "neutre(n,⊥)", alors ils ont la même interprétation car ils sont égaux. La clarté d'argumentation d'une démonstration tient pour beaucoup à ce genre de schémas simples. Une difficulté provient du **iatus entre ce que le mathématicien déclare chercher** (unicité) **et ce qu'il montre en pratique** (égalité de deux élements convenablement choisis).

C'est pourquoi en mathématiques une démonstration achevée propose un problème bien posé et découpe sa solution point après point pour que le lecteur en saisisse l'essence. Il n'est ainsi pas étonnant que ce processus de démonstration suive des règles standardisées auquelles les mathématiciens sont familiers et qu'il rende compte imparfaitement du processus de découverte de cette démonstration tout en passant sous silence des facteurs comme l'intuition ou des représentations ayant servi à cette intuition.

Cet exemple montre que :

La résolution d'un problème est d'abord sa reformulation.

La formulation initiale du problème et sa reformulation sont d'ailleurs l'un des écueils majeurs de la résolution de problèmes par la machine. Cela justifie leur importance et montre que ces opérations sont trop difficiles ou trop peu connues. Les mathématiques et la modélisation de leur expertise permettraient d'évaluer plus précisément leur difficulté.

3.3.5.2. Les aspects heuristiques dans les démonstrations

Les développements précédents ont illustré le rôle du Langage Mathématique dans la conduite et la formulation de la résolution d'un problème. Mais rien ne dit comment ces problèmes sont apparus ! Ainsi, la structure de groupe est issue de régularités provenant de nombreuses structures différentes, telles par exemple le groupe des invariants de transformations géométriques, bien connu en cristallographie. **Si la structure de groupe peut être induite de l'observation naturelle, comment est-elle exploitée une fois intériorisée en mathématiques ?** Là encore, la façon dont le mathématicien aborde les problèmes est indissociable de son langage et de son activité. Nous posons donc maintenant la question ultime, touchant à l'origine même du langage :

Question 4 : comment les problèmes mathématiques sont-ils produits ?

Pour y répondre, il faut constater qu'en situation de découverte le mathématicien sait où chercher avant de savoir conjecturer ce qu'il cherche. La thématisation de certaines propriétés ou des objets sur lesquels elles portent est la cause principale de ce mécanisme de découverte.

Une fois isolée, la structure de groupe est analysée pour savoir quelles sont ses propriétés, pour quelles raisons certaines propriétés se déduisent des autres, quels ensembles minimaux de propriétés suffisent à la définir, et ce qui change si l'on ajoute ou retranche une nouvelle propriété ? Pour ce qui nous concerne, l'origine de toutes ces sessions provient de la lecture des lois de groupe (1) à (3) initiales contenant notamment :

$$\forall l_1, \exists l_2, l_1+l_2=0 \quad \text{.....(3)}$$

Leur expression nous a paru familière, faisant ainsi ressortir l'étrangeté de la conclusion (4) à montrer :

$$\forall l_1, \exists l_2, l_1+l_2=0 \wedge l_2+l_1=0 \quad \text{.....(4)}$$

D'où peut venir le besoin - voire l'idée - de démontrer cela ? De nombreux chercheurs ont expliqué cela par le rôle d'heuristiques de découverte très générales. Ainsi, une heuristique qui analyse les asymétries détecte (3) (inverse à droite) et propose d'essayer de démontrer (4). Douglas Lenat combine de telles heuristiques dans son système AM [Davis&Lenat 82] en montrant leur puissance de découverte (e.g.

l'argument de la diagonale - ici "(4) $\wedge$ $l_1=l_2$" introduit les éléments nilpotents) et leur limite (car elles s'essoufflent et ne produisent plus rien d'intéressant).

Toutefois l'essentiel du potentiel créatif est à rechercher dans l'interaction des acteurs de l'activité mathématique, à savoir le mathématicien, son langage et les données qu'il décrit. Ici la donnée d'un élément neutre et d'un inverse se prêtent à toutes sortes d'altérations constructives (l'associativité est vite considérée comme une primitive assez fondamentale de toute manipulation) : briser la symétrie droite et gauche, briser l'unicité éventuelle, et s'interroger sur les dépendances qui régissent l'existence des divers sous-produits. Mais alors, l'expression (4) n'est-elle pas une forme "symétrisée" masquant une genèse qui serait :

Dans notre présent contexte : $l_1+l_2=0 \Rightarrow l_2+l_1=0$ .....(4')

Cette nouvelle formulation (4') du théorème à montrer traduit l'intention du mathématicien et apparaît d'autant plus naturelle que l'introduction de (4) paraissait étrange : étant donné (1), (2) et (3), le mathématicien montre (4') et conclut (4). De plus (4') n'est pas dépendante d'une formulation "affaiblie" (c.-à-d. optimisée) de la définition d'un groupe : le mathématicien peut se demander si (3) ne suffirait pas au lieu de (4), et trouve ainsi (4'). L'heuristique de découverte d'un mathématicien propose ainsi (4') plutôt que (4), alors que s'il fallait reconstruire les mathématiques, il montrerait (4) directement.

Cette facheuse tendance de **masquer la génèse des problèmes risque d'aseptiser les mathématiques**. Elle a d'ailleurs une influence pernicieuse pour la réutilisation de preuves, puisque la preuve construite pour (4) n'apporte rien de plus à celle de (4'). La preuve de (4') sert en effet à résoudre d'autres problèmes voisins tel une réciprocité droite-gauche des problèmes ou tel l'unicité de l'inverse à gauche une fois qu'il existe.

De plus, la symétrie du théorème à montrer sur les inverses n'est intéressante qu'une fois le résultat acquis, afin de l'utiliser. Lorsque le mathématicien s'y intéresse à un niveau de détail tel que ce résultat n'est pas atomique, ce théorème masque la causalité qui l'a engendrée, en y intégrant la réciprocité de cette causalité et de la construction qui l'exploite. En notant "d" l'inverse à droite (et pour raison de symétrie visuelle : "b" l'inverse à gauche) cela se traduit schématiquement par :

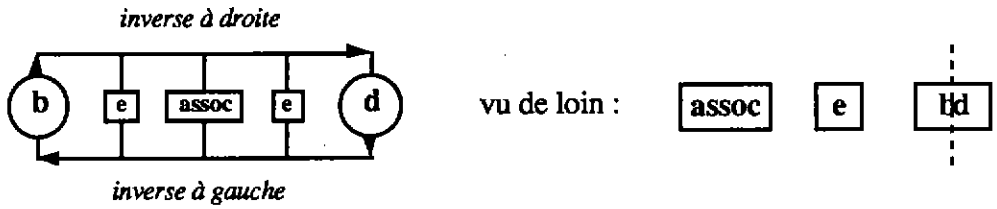


figure 1.13

Le schéma de gauche permet d'exprimer les liens de causalité entre arguments. Ainsi nous avons construit la preuve parfois à partir de l'associativité, parfois à partir d'un bout, et l'on pourrait procéder par équivalence à partir de "b = d".

Le schéma de droite représente une vue lointaine où le schéma de gauche est rendu atomique en ne considérant qu'un inverse à gauche et à droite. L'énoncé des lois de groupe se résume alors à trois points.

Les mathématiques relèvent donc d'une construction qui se décrit à un niveau d'organisation différent selon la nature du problème à résoudre. Le non respect de ce principe masque l'origine de certains problèmes et rend les démonstrations incapables de décrire l'argument essentiel. Par exemple, une preuve de "l'irrationalité de $\sqrt{2}$ " basée sur la parité de p et q tels que "$p^2 = 2q^2$", masque l'argument fondamental qui est basé sur leur décomposition en facteurs premiers.

3.3.5.3. Les processus de découverte heuristique

La recherche effective durant l'activité mathématique porte non seulement sur un résultat à prouver, mais est indissociable de la recherche des notions et des conjectures auxquelles le mathématicien va s'intéresser.

Le rôle des représentations et des « images vagues » est amplement détaillé dans l'ouvrage de J. Hadamard [Hadamard 75]. Ces représentations interviennent dans toute activité mathématique, et pas

seulement en géométrie. Hadamard utilise par exemple (p 75-78) des images mentales même pour des études arithmétiques, algébriques ou analytiques (et non géométriques). Elles lui permettent une vue simultannée du raisonnement, lui rappellent une démonstration (et non pas sa preuve), et sont suffisamment vagues pour le conduire sans fausse route. Ces représentations sont alors des guides de la pensée et l'un des facteurs d'intuition.

Sachant cela, il est possible d'étudier les mathématiciens dans leur résolution de problèmes donnés. Cette observation des mathématiciens a été notamment effectuée par Dominique Pastre afin d'utiliser les heuristiques dégagées pour la Démonstration Automatique de Théorèmes [Pastre 78]. Ce travail, particulièrement novateur, a comme spécificité d'adopter une approche cognitive comme fondement d'une modélisation informatique.

Les brouillons sont suffisamment proches de l'activité mathématique pour capturer cette genèse heuristique. Par contre, les démonstrations produites sont souvent trop éloignées car elles résultent d'un réarrangement notable de l'argumentation masquant de nombreuses causalités.

Les démonstrations produites adoptent souvent le point de vue lointain du schéma de droite de la figure 1.13 précédente, car elles ne bâtissent que pour exploiter le résultat et non pour comprendre ses mécanismes. A l'extrême le style académique ne cherche qu'à être élégant, quitte à s'encombrer de facteurs superflus et à employer des figures de style inversant les causalités entre arguments. Comment, dès lors, disposer de bases saines pour répondre au corollaire de la découverte de problèmes intéressants :

Question 4bis : comment le mathématicien trouve-il une démonstration ?

Il serait trompeur de croire que parce que les mathématiciens savent faire des démonstrations, ils connaissent l'activité qui les produit et les étapes de construction d'une preuve. Ils savent présenter des démonstrations, mais savent à peine ce qu'elles sont et ne disposent que d'une abstraction logique comme modèle d'une preuve mathématique !

Une étude de preuves et démonstrations nous permettrait de mieux les connaître, et une étude de leur construction permettrait de mieux connaître l'activité mathématique et les facteurs de créativités qu'elle exploite. Il pourrait ainsi y avoir des démonstrations destinées simplement à justifier, et d'autres destinées à analyser et comprendre. Enfin, la connaissance du processus de construction d'une démonstration permettrait de s'interroger sur les raisons et la pertinence de certains choix académiques, ainsi que de proposer des solutions différentes ou meilleures.

3.4. SYNTHESE DE CE PANORAMA DU LANGAGE MATHEMATIQUE

Le Langage Mathématique participe à l'activité mathématique

Nous voulons mettre ici l'accent sur les mécanismes de découverte à partir du langage. Par exemple, nous avons détecté un lapsus lors d'une première écriture de l'unicité de l'élément neutre. Des manipulations langagières visant à explorer la signification de cette formule "fausse" nous ont conduit au concept d'élément absorbant. L'examen des propriétés d'un élément absorbant a mis en évidence que le groupe se réduisait à l'élément neutre à cause de l'inversibilité. Nous savons maintenant pourquoi un élément absorbant est "forcément" en dehors de la structure de groupe (d'où $(Z^*, \times)$!). Des structures plus faibles sont alors nécessaires pour examiner l'antinomie entre éléments inversibles et éléments absorbants.

Cette session heuristique à partir des propriétés des groupes n'est pas sans rappeler des exemples de conjectures et réfutations tels celui analysé par Imre Lakatos sur la formule d'Euler [Lakatos 84], ou celui que Imre Lakatos a inspiré à Reuben Davis et Reuben Hersh sur les nombres d'or [Davis 85]. Comme ces derniers nous l'appliquons à la créativité intériorisée aux mathématiques sur une courte durée, avec ici un mathématicien de niveau mathématique restreint.

Cette session générale résulte ainsi de questions voisines : le problème initial portait sur l'existence et l'unicité d'un inverse, le suivant fut l'existence et l'unicité de l'élément neutre, le suivant exploitait l'unicité pour caractériser l'élément neutre, caractérisation dont une partie a muté pour déclencher un processus de découverte.

Cette session résulte de l'interaction entre les acteurs de l'activité mathématique : le mathématicien, le langage et les données. Pour ce qui concerne l'importance du langage, l'expression des données amène irrémédiablement le mathématicien à se poser un certain nombre de "questions types" comme ici l'existence ou l'unicité. Mais si le langage et les données qu'il exprime ont joué un rôle déterminant dans ces développements heuristiques, les innovations proviennent surtout de la façon dont le mathématicien interprète le langage et dont il s'en sert.

Le Langage Mathématique n'est pas juste un observable possible de l'activité mathématique : il n'est pas neutre et participe à l'activité mathématique. Aussi les tentatives de formalisation doivent tenir compte de ses caractéristiques si elles veulent préserver la créativité du mathématicien.

Le Langage Mathématique n'est pas complètement formalisable

Le langage n'est pas simplement utilisé comme une simple règle de réécriture. Le mathématicien fait notamment intervenir dans son interprétation du langage le comportement opératif des règles et données disponibles. Cela se traduit par des appels multiples à l'intuition comme facteur d'explication. Citons pèle-mêle lors d'une session de découverte :

- l'énoncé des conjectures qui s'expriment avec les notions de ces représentations que le mathématicien se forge. Ces représentations proviennent des opérations, associations et similarités issues du vécu du mathématicien. Elles utilisent un riche vocabulaire intuitif.
- l'ambivalence d'une expression permettant soit de produire l'élément neutre (qui est utile dans toute manipulation), soit de produire l'élément absorbant (qui est puissant par l'influence qu'il exerce) ;
- les liens d'incompatibilité entre "absorbants" et "inversibles", qui se traduisent par une forte influence dynamique des "absorbants" et implique leur isolement ;
- le besoin et le travail sur des exemples quelconques pour s'interroger sur leur construction (l'aspect opératif vient de cette construction, de plus dans quelle mesure et selon quels critères le mathématicien qualifie-t-il son exemple de quelconque ?) ;

Le sens du langage n'est pas simplement dans une interprétation qui préserve la vérité et les équivalences entre objets mathématiques, mais principalement dans la façon que chaque mathématicien a de s'en servir.

C'est pourquoi, une sémantique dénotationnelle ou fonctionnelle des mathématiques ne suffit pas, il faudrait notamment une sémantique pouvant capturer toutes les interprétations "méta-opératives"[5] possibles...

Pour clore l'impossibilité d'une sémantique mathématique générale comme cela est proposé en logique, une sémantique ne dit rien sur les critères de choix et heuristiques à employer. C'était simple de construire les démonstrations précédentes : il y a peu d'objets pertinents en présence (x,x^{-1},y,y^{-1},e) et nous pouvions envisager de les énumérer tous, pour toutes les règles. Mais les mécanismes sont les mêmes qu'en analyse par exemple où il y a un grand nombre de règles applicables à une infinité d'expressions possibles. De plus des critères similaires s'appliquent à la génération de conjectures, de symboles, de notations, de nouvelles définitions... Les liens de causalité doivent être élucidés sous peine de sombrer dans une complexité indécidable.

Il est adapté à l'usage des mathématiciens

Le fait que le Langage Mathématique soit adapté à l'usage des mathématiciens est souvent mésestimé. Nous avons cherché à savoir comment dans ce panorama. Nous avons aussi cherché à illustrer la faiblesse de notre connaissance actuelle et les limites des solutions inspirées de la logique pour modéliser le Langage Mathématique.

Pour répondre plus complètement aux interrogations de Jean Dhombres, nous avons exploré ensemble les qualités suggestives et descriptives, ainsi que les ressources de la zone élémentaire du Langage Mathématique. Rappelons de plus que les ambiguïtés rencontrées sont utiles pour leur intérêt

---

[5] Nous employons ce néologisme pour indiquer que ces interprétations concernent la façon d'analyser le rapport interne entre un mathématicien et la perception qu'il se forge sur les manipulations mathématiques.

ergonomique direct (e.g. c'est plus simple à manipuler) ou indirect (le mathématicien est globalement bénéficiaire). Elles se retrouvent ainsi tant au niveau des symboles qu'au niveau des notations (multiplicatives notamment). Toutefois, ces ambiguïtés ne se veulent qu'apparentes et doivent être levées par le mathématicien s'il en a besoin.

Cette exploration n'était toutefois que très fragmentaire, et les interrogations de Dhombres restent essentielles. L'apport des Sciences Cognitives, allié à un objectif pratique, pourrait permettre de compléter les réponses ébauchées par une étude théorique basée sur des expériences de manipulation par des mathématiciens.

Nous proposons donc d'étudier le Langage Mathématique, non pas pour remplacer le mathématicien lors de son activité mathématique, mais afin de reproduire l'usage du Langage Mathématique afin **d'assister le mathématicien dans son activité.**

L'enjeu de cette thèse est une meilleure connaissance du langage et de ses possibilités de modélisation informatique. C'est pourquoi nous écartons délibérement la question des capacités de raisonnement et des fonctions créatives du mathématicien pour aborder les questions indispensables à toute activité mathématique pratique :

- **Qu'est-ce qui est important dans le langage pour le mathématicien ?**
- Quelles sont les connaissances dont dispose le mathématicien ?
- Par quels mécanismes une solution est-elle construite par morceaux ?
- Comment représenter un problème mathématique résolu ?
- Comment présenter une démonstration d'une preuve solution ?

Nous avons de ce fait adopté la **démarche suivante pour l'étude du Langage Mathématique** :

a) état du domaine d'application : les mathématiques, l'activité mathématique et les mathématiciens ;

b) description du problème : la connaissance et l'étude de l'activité mathématique sont insuffisantes pour décrire et définir ce que sont l'activité mathématique, le Langage Mathématique et ses productions. Comme tout problème de conception, le processus et les étapes de construction d'une preuve ou d'une démonstration sont essentiellement inconnus ;

c) souhaits : nous avons besoin de modèles précis pour un objectif pratique. Faute d'en trouver, nous avons dû explorer nous-même les points suivants : les processus de l'activité du mathématicien, les objets (concepts, preuves, démonstrations) et les connaissances requises ;

d) état des lieux sur l'activité : le savoir-faire d'analyse d'une activité de conception existe mais n'est guère appliqué aux mathématiques. Les principaux travaux sont issus de la pédagogie ou de la démonstration automatique, et portent principalement sur l'activité de résolution de problèmes et la recherche heuristique ;

e) Pour résoudre le problème b), il faut encourager et regrouper ces travaux d'analyse des mathématiques, en les associant à une **linguistique des mathématiques** ;

f) Pour détailler les modèles que nous avons introduits lors de notre étude pratique de l'activité mathématique, nous proposons de mener conjointement une **étude expérimentale** de l'usage du langage et une étude expérimentale basée sur les données d'observation recueillies par un outil de construction informatique de preuves et démonstrations.

4. ANALYSE LINGUISTIQUE D'ECRITS MATHEMATIQUES

4.1. VERS UNE LINGUISTIQUE DES MATHEMATIQUES

4.1.1. Analyse linguistique des textes de démonstrations

> Lorsqu'il s'agit de mathématique, les concepts de "raisonnement" et de "démonstration" entretiennent manifestement des rapports étroits. Il est donc naturel, pour explorer la polysémie du premier, d'étudier en préalable la pragmatique du second. Autrement dit, nous nous proposons, dans une première étape, d'analyser linguistiquement les passages des textes mathématiques qui sont explicitement présentés comme des démonstrations.

Ce fondement d'une analyse linguistique des mathématiques a été rédigé par Daniel Lacombe pour les journées de l'ARC [Lacombe 84]. Il y présente l'analyse traditionnelle du Langage Mathématique comme :

① des conventions d'écritures, dont des définitions ;

② des énoncés mathématiques, exprimant des propriétés sur les objets mathématiques ;

③ des indicateurs d'inférence, plus ou moins détaillés ;

④ des commentaires "heuristiques" ou "méthodologiques".

Pour cette analyse traditionnelle, une démonstration est essentiellement constituée de composants de type ② et ③. Pour mériter ce nom, une démonstration doit entraîner de proche en proche la **conviction**. Daniel Lacombe relève les insuffisances d'une telle analyse pour laquelle la spécificité majeure du Langage Mathématique se réduit au mélange de langue naturelle et de formules. Il note le caractère spécifique du discours mathématique, le détournement du sens usuel des mots, et l'effacement des marqueurs métalinguistiques. Cette spécificité et le détournement qui en résulte sont aussi observés chez les juristes : "groupe", "loi", "variable", "quelconque"... ont un sens différent dans la vie courante, pour les mathématiciens et les juristes.

Outre l'étude de ces insuffisances, Daniel Lacombe précise l'intérêt d'une analyse plus fine du Langage Mathématique :

- Un texte mathématique réel ne contient en général que des fragments d'énoncés mathématiques et des ébauches de démonstrations mathématiques.
- Par contre (et par conséquent) une grande partie du texte est consacrée à fournir au lecteur des **indications** sur la façon dont il convient de compléter, de rectifier éventuellement et de combiner entre eux ces fragments et ces ébauches afin d'obtenir des affirmations complètes et des preuves valides.

Pour désigner ces indications exprimées dans un langage extrêmement conventionnel (bien que non symbolique), Daniel Lacombe utilise l'adjectif épimathématique. Il subdivise ainsi figure 1.14 le Langage Mathématique en six zones.

Les 6 zones du Langage Mathématique

		langage usuel	langage ad-hoc (symbolique)
mathématique		① fonction fausse deux plus six	② f F 2 + 6 > 3
métamathématique	épi-mathématique	③ prouver d'où ce qui est absurde	④ *très peu utilisée* dém cqfd (3)
	péri-mathématique	⑤ Ce résultat sera démontré au §3.3.	⑥ *très peu utilisée* N.B. [19]

figure 1.14

La majorité du vocabulaire spécialisé se situe dans la zone 1 puisqu'elle inclut tous les concepts des théories mathématiques. Les domaines se prêtant aux manipulations formelles sont représentés dans la zone 2 puisqu'ils disposent de plus d'une écriture symbolique. Un noyau de vocabulaire générique se situe dans la zone 3 épimathématique, car il s'agit de marqueurs et de rupteurs d'inférence employés indépendamment d'un domaine de démonstration. La zone 5 périmathématique est relative aux annotations : elle positionne la présentation d'une démonstration dans un contexte plus large que la résolution du problème. Les aspects ad-hoc du métalangage (zones 4 et 6) ne sont guère représentés autrement que par des marqueurs généraux, auxquels il faudrait ajouter des "astuces" typographiques comme la mise en italique.

Daniel Lacombe note d'ailleurs que :

> C'est au niveau épimathématique que se produisent la plupart des détournements infligés par la langue mathématique aux procédés de la langue usuelle. D'autre part, on constate que l'envahissement - déjà signalé - du discours mathématique par la métalangue se produit lui aussi au niveau épimathématique, et non pas à propos de questions "métamathématiques" (au sens des logiciens) - ce genre de questions étant généralement absent des textes mathématiques.

Pour accomplir sa fonction de communication, il est naturel qu'un "jargon" opératif spécifique se soit développé en mathématiques. Le mélange du niveau mathématique et du métalangage se fait alors naturellement, puisque le but principal du discours porte sur la résolution du problème, et non sur les objets mathématiques et leurs énoncés.

Nous laisserons Daniel Lacombe conclure cette analyse linguistique :

> nous sommes donc en présence de trois sortes d'éléments, de natures totalement différentes :
> - des **objets formels** de nature combinatoire finie ;
> - des **procédés linguistiques** permettant de décrire ces objets, de façon plus ou moins concentrée et plus ou moins complète ;
> - des **processus psycholinguistiques** permettant au lecteur de reconstituer, explicitement ou implicitement, des descriptions complètes à partir de descriptions incomplètes qui lui sont fournies (ces processus n'aboutissant que dans les « bons » cas, c'est-à-dire lorsque l'auteur n'a commis ni faute mathématique ni erreur didactique et que le "niveau" du lecteur est conforme aux prévisions).

4.1.2. Notre objectif pratique d'analyse du langage

Ces analyses du Langage Mathématique permettent d'étudier son évolution, ses rôles et son traitement cognitif. Notre objectif de conception d'un outil d'assistance au mathématicien nous imposent un point de vue pratique :

> Expliciter et spécifier l'usage du Langage Mathématique

Ce point de vue aborde le Langage Mathématique de façon à comprendre la pratique du langage par le mathématicien. Il apparaît que cette réflexion peu explorée fait partie d'une réflexion d'ensemble sur le Langage Mathématique. Pour pouvoir en parler, il faut le connaître !

De nos jours, le support des connaissances et leur modèle de traitement sont souvent informatisés. L'objectif le plus réaliste pour étudier et spécifier l'usage du Langage Mathématique est ainsi d'en construire un **modèle adapté à l'informatique**. L'informatisation oblige notamment à clarifier et détailler les analyses, tout en imposant un spectre plus large et uniforme des composantes analysées. L'exemple du traitement du Langage Naturel montre les bénéfices interdisciplinaires que l'on peut en espérer, ainsi que l'**urgence d'une telle réflexion** en mathématiques. Comme pour le Langage Naturel, la finalité de la recherche peut être "fondamentale" ou "appliquée".

La dualité "fondamental-appliqué" montre l'arbitraire de cette dichotomie lorsqu'on la pense en terme de faisabilité. Si une spécification fondamentale de cette ampleur voit le jour, cela ne peut être que sous l'impulsion de son utilité pratique afin que son développement puisse être effectué et validé en un temps raisonnable. Il lui faut bénéficier d'une synergie de motivations, sciences et techniques interdisciplinaires faisant une large part à l'informatique.

La linguistique des mathématiques n'est pas seule concernée par une recherche fondamentale. La connaissance du Langage Mathématique va de pair avec celle de l'activité mathématique. Une modélisation informatique "peu perturbatrice" du langage permet de recueillir les principales données observables de l'activité mathématique. Elle autorise alors l'expérimentation "fine" des facteurs influents et des processus cognitifs mis en œuvre. L'ensemble des sciences cognitives est alors impliqué.

Le langage Mathématique présente une conjonction de facteurs qui rend son étude particulièrement intéressante d'un point de vue linguistique. Les caractéristiques structurelles d'un texte écrit en Langage Mathématique sont similaires à celles d'un texte en Langage Naturel. On retrouve par exemple le découpage d'un article en sections avec des paragraphes étiquetés sous forme de théorèmes et démonstrations. La différence principale porte sur les règles de construction syntaxique employées. Notamment, la caractéristique fondamentale des textes mathématiques est la structure "rigoureuse" du récit. Cette caractéristique rend possible la représentation détaillée d'une preuve et démonstration, et permet d'envisager la représentation du sens des textes mathématiques à partir de cette structure de base.

Cet aspect formel constitue la différence essentielle dans le traitement du Langage Mathématique par rapport à celui du Langage Naturel[1]. Plus généralement, le Langage Mathématique peut être pensé comme une **opérationnalisation du Langage Naturel** usuel avec compaction des formes de langage et adjonction d'une forte composante graphique (formules, schémas...). Il partage donc ses caractéristiques globales avec les dialectes opératifs du Langage Naturel : importance du contexte, plusieurs canaux d'information disponibles (avec une redondance éventuelle savamment dosée) et une ambiguïté uniquement extrinsèque (ç.-à-d. un sens unique visé).

Il comporte enfin une forte composante argumentative dans son activité de démonstration l'autorisant à paraphraser et lui permettant d'exprimer ses intentions. Le Langage Mathématique respecte ainsi les deux facettes de sa fonction mathématique - l'expressivité et la rigueur[2] - en favorisant la validation et la diffusion des résultats par sa fonction de communication.

---

[1] Ne pas confondre avec un langage formel dont la structure a été choisie a priori afin de faciliter sa définition, son automatisation et les méta-preuves. Le Langage Mathématique est issu d'autres besoins !

[2] Par exemple les analystes du 18ème siècle recherchaient l'expressivité afin d'obtenir de nouveaux résultats, tandis que ceux du 19ème s'attachèrent à donner des définitions précises et des preuves rigoureuses (pour une discussion voir l'article de Judith V. Grabiner [Grabiner 86]).

Une **approche linguistique des mathématiques est nécessairement cognitive**, car l'activité du mathématicien ne peut s'envisager qu'en analysant ses connaissances, ses calculs et ses raisonnements. Nous distinguons trois raisons essentielles nécessitant le développement d'une linguistique spécialisée aux mathématiques :

- parce qu'il existe une expertise spécifique au langage des mathématiques et à son usage ;
- parce qu'une analyse cognitive de l'activité mathématique est indispensable pour proposer des outils d'assistance au mathématicien ;
- parce qu'une linguistique des mathématiques permet d'envisager la modélisation de l'activité mathématique naturelle dans sa complexité de langage et de raisonnement.

4.1.3. Les niveaux de discours des textes mathématiques

L'usage du langage en mathématiques est souvent performatif : "dire, c'est faire". Le Langage Mathématique se trouve habituellement surchagé car il mêle plusieurs niveaux de description : la description des objets de discours, la description des moyens de transformation de ces objets, la description des intentions qui régissent ces transformations, etc.

Nous proposons de partitionner l'analyse de textes mathématiques en deux niveaux de discours selon les ingrédients qu'utilise le mathématicien :

- le niveau de discours mathématique, qui correspond aux fonctions mathématiques utilitaires du langage (descriptives et évaluatives). Il comprend un langage d'expressions mathématiques (langage dit "objet") et un "méta"-langage chargé des manipulations ;
- le niveau de discours métamathématique, qui correspond aux fonctions de communication du langage (epistémiques et explicatives). Il comprend de plus des procédés langagiers capables de décrire le déroulement de l'activité du mathématicien ;

Le **niveau de discours mathématique** se subdivise alors en deux parties :

- Les expressions mathématiques.
  Leur écriture dans un document mathématique s'effectue selon le cas au moyen de langage de Phrases, de Formules ou de Schémas.
- L'argumentation mathématique :
  c'est-à-dire les moyens langagiers de définition de problèmes et de construction de preuve. L'expressivité de l'argumentation est fortement dépendante des connaissances et compétences de raisonnement du destinataire.

Vu l'ampleur et la diversité du niveau mathématique nécessaire à la pratique d'une activité mathématique, il est bon de préciser ce qui n'en fait pas partie : c'est ce qui interpelle directement un autre mathématicien, et qui se situe donc à un niveau métamathématique. Ici encore l'analogie avec un langage de programmation permet de discerner ces niveaux, la difficulté provenant du double rôle d'interprétation du mathématicien à la fois interpréteur et programmeur[3].

Le niveau de discours mathématique représente les instructions nécessaires à l'exécution du programme qui permet de résoudre le problème mathématique. Le niveau de discours métamathématique représente tout ce qui, dans le code du programme, est destiné au programmeur. Nous distinguons deux catégories, en adaptant à notre propos la distinction de Daniel Lacombe entre périmathématique et épimathématique.

Le **niveau de discours périmathématique** est directement concerné par l'homme-mathématicien et contribue activement à son apprentissage. Il sert de liant à l'activité de diffusion du mathématicien, en coordonnant le discours présenté et ses connaissances. Il est notamment constitué de commentaires comportant :

- des informations de **compréhension** (fonction informelle d'un module de code programme, effets d'une transformation, exemple représentatif, etc) ;
- des descriptions issues de la **planification** (liées aux étapes de développement du logiciel) qui décrivent des objectifs et raisonnements issus de la conception de la démonstration ;
- des considérations **heuristiques** (techniques de programmation, critères d'efficacité, etc) ;

---

[3] Cette analogie est fructueuse puisqu'une approche linguistique des langages de programmation est notamment utilisée comme support d'enseignement de la linguistique informatique [Anis 90]. Nous sommes convaincus que l'analyse du langage usuel sera essentielle pour la définition des langages de la Communication Homme-Machine à venir.

- des paramètres de contrôle de l'**interprétation** ("verbose", "debug", "optimised") qui permettent notamment des exercices didactiques ;
- des informations à caractère **pédagogique ou mnémonique** (exemples, notes historiques, anecdotes, métaphores, etc) ;
- des indications de **ressources** connues et à utiliser (ouvrages de résolution d'équations différentielles, tables de calcul d'intégrales, appel à un traitement informatique !, théorie élémentaire des nombres maitrisée, etc) ;
- des instructions de **formatage** et les composantes spécifiquement textuelles du document (page, table des matières, etc) ;

Le **niveau de discours épimathématique** est l'analogue de commentaires décrivant à un autre programmeur comment coder une fonction absente (ou simplement ce qu'elle fait). Il correspond parfaitement à une fonction explicative spécialisée du langage qui ne décrit pas une solution, mais donne des éléments de sa construction.

La nuance entre argumentation mathématique et cette argumentation épimathématique est parfois assez délicate à cerner car elle repose sur la distinction entre les processus mentaux du mathématicien-lecteur et ceux du mathématicien-interpréteur supposés normalisés par tel langage mathématique (comme il existe par exemple une norme FORTRAN-2). Aussi, sur un plan strictement linguistique, l'argumentation mathématique est parfois écartée. Toutefois, cette distinction lève l'amalgame couramment effectué en mathématiques entre les niveaux de discours.

Au lieu de la levée des omissions et ambiguïtés d'un langage, l'argumentation épimathématique fait appel à de véritables capacités de résolution de problèmes. Elle permet au mathématicien d'user d'une véritable programmation allusive au lieu d'une simple programmation descriptive. Elle préfigure en ce sens les langages de communication utilisés dans la programmation à venir.

Le rôle de l'argumentation épimathématique pour la compréhension est si important que la majorité des démonstrations se situent à ce niveau d'abstraction. Cette argumentation fait néanmoins abondamment appel à l'argumentation mathématique pour délimiter des points de passage. Cela se traduit par une trame inter-niveaux. Les démonstrations de ce type contiennent alors à la fois des suggestions et des descriptions d'opérations mathématiques à effectuer pour construire la preuve solution.

4.1.4. Une approche d'analyse

Nous avons ainsi constitué une grille d'analyse permettant la description intégrale du texte d'un document mathématique à partir de son interprétation cognitive. Elle se résume en six points :

niveau métamathématique
- L'argumentation périmathématique
- L'argumentation épimathématique

niveau mathématique direct
- Les expressions mathématiques
- L'argumentation mathématique

niveau mathématique indirect
- Le contexte d'objets
- L'espace des problèmes

L'intérêt principal de cette grille est de fournir un outil préliminaire au sein d'une **méthodologie d'analyse linguistique** spécialisée aux mathématiques. Le dépouillement de nos observations montre que :

- Le niveau de base ("quoi") est bien entendu constitué des expressions mathématiques, elles-mêmes utilisant le contexte d'objets.
- La manipulation de ce niveau est regroupé dans l'argumentation mathématique et épimathématique (le "comment") et dans l'espace des problèmes (le "pourquoi").
- Les textes mathématiques peuvent être scindés en deux catégories en fonction de l'occupation du niveau métamathématique : l'une visant d'abord la validité et la précision mathématique, l'autre visant principalement la compréhension et l'assimilation. Cela dépend uniquement de l'objectif du rédacteur.

Pour ce qui concerne l'argumentation mathématique, une analyse pratique permet de typer son contenu en fonction des trois axes de la figure 1.5 (objets, construction de preuve, résolution) :

- la **désignation** de sous-expressions, de propriétés et de règles de déduction à l'aide des procédés éventuellement allusifs du Langage Naturel.
- **l'utilisation** de règles de construction de preuves telle l'introduction d'objets ou d'hypothèses, la manipulation de problèmes, la substitution, les règles de déduction, etc.
- le **raisonnement mathématique** et sa traduction à l'aide de marqueurs langagiers.
- des **ressources de traitement** tel la compréhension des énoncés, des capacités de perception, des calculs arithmétiques, un savoir-faire d'utilisation ou de mise en œuvre d'une règle, la résolution de systèmes d'équations particuliers, etc.

Les règles et ressources utilisées peuvent être mathématiquement justifiées si besoin. Les ressources de traitement sont utilisées dans tous les points. Elles se trouvent parfois explicitées dans une démonstration pour clarifier l'argumentation.

Le raisonnement mathématique doit être catégorisé en fonction de ses actions sur le problème et sur l'évolution de la preuve. Il se retrouve aussi au niveau épimathématique dans la mesure où les opérations de transformation ne sont pas toujours clairement définies au niveau mathématique. Nous discernons une fonction de jugement (classification d'objet, vérité, évaluation de difficulté, etc.) et une fonction de résolution de problèmes, raisonnement et conduite du raisonnement que nous partitionnons en :

- gestion du contexte d'objets au sein d'un même problème ;
- gestion du problème (transformation de problèmes équivalents, changement de statut tel l'inversion d'hypothèses et conclusions, etc.) ;
- gestion locale de la démonstration (or, d'où...) ;
- gestion globale de la démonstration (résolution à partir des buts et des données, élimination de certaines données intermédiaires après leur introduction, etc.) ;
- gestion des calculs (utilisation des règles en algèbre, utilisation de théorèmes, etc.) ;

Il est possible de synthétiser à partir des points de cette argumentation mathématique une couche mathématique d'intérêt général que la logique a partiellement cherché à formaliser. Il suffit alors d'ajouter en paramètre des théories mathématiques selon le croquis ci-dessous pour aboutir au contexte nécessaire à la description du document.

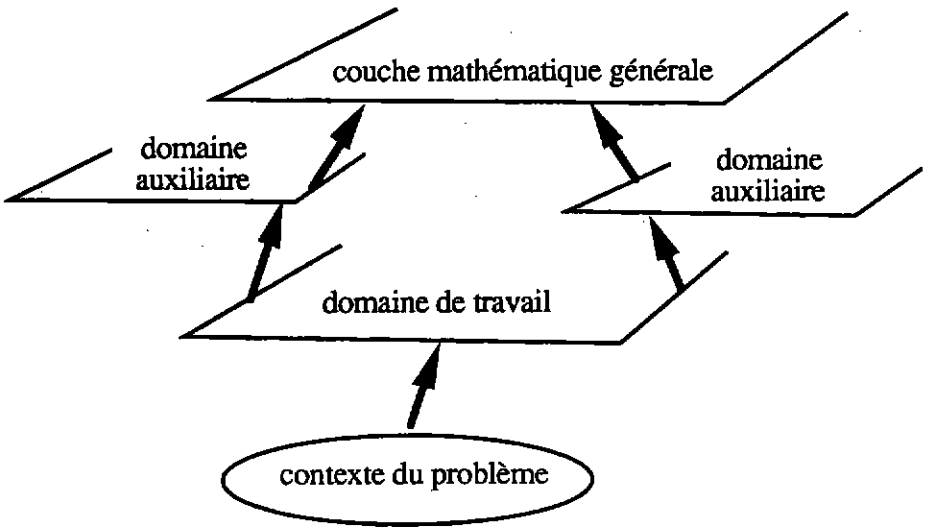


figure 1.15

Un domaine auxiliaire possible est par exemple la théorie des nombres. L'important est toutefois de déterminer la nature exacte des connaissances qu'il comporte. Un tel domaine auxiliaire amène des définitions et théorèmes particuliers, des procédures (pgcd) et heuristiques et des méthodes de raisonnement (analyse modulo n, récurrence).

Cette grille d'analyse permet ainsi de **catégoriser l'argumentation et le contexte de travail du mathématicien**. Son avantage majeur est la simplicité de la catégorisation. Le résultat obtenu est descriptif et non explicatif : il doit donc être soumis à analyse ultérieure selon le besoin. Dans notre analyse pratique du langage, elle contribue à :

- une description des problèmes mathématiques et de leur organisation ;
- un recensement des règles de déduction pour établir un modèle de preuve ;

- un recueil de connaissances de base de l'activité mathématique (ce qu'est une définition, quels sont les théorèmes employés, le savoir-faire d'application et les ressources utilisées, etc) ;

Cette grille est donc à l'usage du linguiste-mathématicien. Elle poursuit le travail d'interrogation effectué par Daniel Lacombe dans [Lacombe 84] sur l'envahissement des mathématiques par sa métalangue. Elle constitue une tentative unique à notre connaissance de discerner et d'expliquer le mélange quasi-inextricable de modalités et de niveaux d'abstraction en mathématiques. Selon ses lectures elle permet de :

- approfondir l'étude de coutumes mathématiques et de phénomènes linguistiques;
- étudier les procédés linguistiques d'argumentation et de description ;
- relier des entités linguistiques aux concepts mathématiques qu'ils représentent ;
- trouver par l'analyse la structure de ces concepts et connaissances mathématiques ;
- permettre la description de ces concepts en terme de tâches-problèmes ;
- recenser les actions du mathématicien et leurs liens avec l'état courant ;
- modéliser le travail du mathématicien et l'état instantané de son contexte de travail.

Cette grille est complétée dans notre étude par :

- une analyse des actions et ressources utilisées par les mathématiciens dans leur argumentation mathématique ;
- une modélisation des problèmes, formules, contexte d'objets, preuves et démonstrations ;
- une analyse et une modélisation de connaissances spécifiques à la couche mathématique générale ;
- une analyse détaillée des manipulations de formules visant à expliquer les lois qui les régissent.

Calculons la surface quand θ varie de 0 à π et doublons :

$$S = 2\int_0^{\pi} \frac{1}{2}\rho^2 d\theta = \int_0^{\pi} a^2(1+\cos\theta)^2 d\theta$$
$$= a^2\left[\int_0^{\pi} d\theta + \int_0^{\pi} \cos^2\theta\, d\theta + 2\int_0^{\pi} \cos\theta\, d\theta\right]$$
$$= a^2\left[0 + \left(\frac{\theta}{2} + \frac{1}{4}\sin 2\theta\right) + 2\sin\theta\right]_0^{\pi}$$
$$= \frac{3\pi a^2}{2}.$$

4° *Surface de la parabole* (fig. 34), $y^2 = 2px$.

— Profitons de cet exemple pour montrer au lecteur comment on trouvé l'équation de la parabole en coordonnées polaires, en prenant le pôle au foyer F.

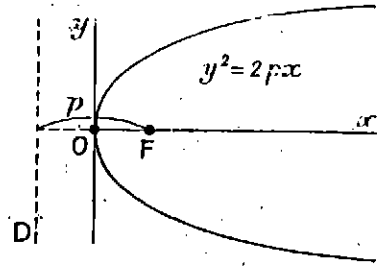


Fig. 34.

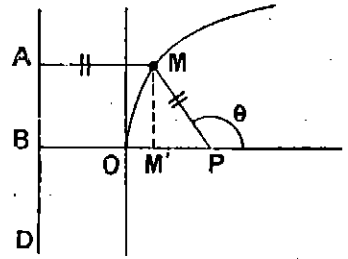


Fig. 35.

Appelons ρ le rayon vecteur MP (fig. 35), on sait que la parabole est le lieu géométrique des points M équidistants d'un point fixe P, appelé foyer, et d'une droite D appelée directrice. La figure donne

$$MA = PB - PM' = p - \rho\cos(\pi - \theta)$$

or $MA = MP = \rho$,

d'où $\rho = p + \rho\cdot\cos\theta$

et $\rho(1-\cos\theta) = p$

$$\rho = \frac{p}{1-\cos\theta} = f(\theta).$$

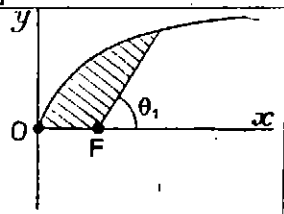


Fig. 36.

La surface du secteur hachuré, (fig 36), sera donc

$$S = \int_{\pi}^{\theta_1} \frac{1}{2}\rho^2 d\theta = \frac{1}{2}\int_{\pi}^{\theta_1} \frac{p^2}{(1-\cos\theta)^2} d\theta.$$

Posons

$$\operatorname{tg}\frac{\theta}{2} = t \qquad \text{avec} \qquad d\theta = \frac{2dt}{1+t^2},$$

On aura

$$S = \frac{p^2}{2}\int_{\pi}^{\theta_1} \frac{\frac{2dt}{1+t^2}}{\left(1-\frac{1-t^2}{1+t^2}\right)^2} = p^2\int_{\pi}^{\theta_1} \frac{(1+t^2)dt}{4t^4}$$
$$= \frac{p^2}{4}\left[\int\frac{dt}{t^4} + \int\frac{dt}{t^2}\right]_{\pi}^{\theta_1} = -\frac{p^2}{4}\left(\frac{1}{3\operatorname{tg}^3\frac{\theta_1}{2}} + \frac{1}{\operatorname{tg}\frac{\theta_1}{2}}\right).$$

On remplacerait θ_1 par la valeur choisie.

5° *Calculer la surface de la boucle du folium de* Descartes.

— On a déjà calculé la surface de la courbe en coordonnées cartésiennes. Pour simplifier les calculs, prenons $a = 1$, d'où l'équation

$$x^3 + y^3 = 3xy.$$

Pour l'exprimer en coordonnées polaires, posons

$$x = \rho\cos\theta \qquad \text{et} \qquad y = \rho\sin\theta$$

d'où

$$\rho^3\cos^3\theta + \rho^3\sin^3\theta = 3\rho^2\sin\theta\cos\theta$$

et

$$\rho = \frac{3\sin\theta\cos\theta}{\sin^3\theta + \cos^3\theta}.$$

Cherchons les limites de θ *de façon que* ρ *parte de* O *et revienne au point* O :

$\rho = 0$ pour $\theta = 0$ (c'est la plus petite valeur de θ)

et $\rho = 0$ pour $\theta = \frac{\pi}{2}$ (c'est la plus grande valeur de θ).

On aura donc

$$S = \frac{1}{2}\int_0^{\frac{\pi}{2}} \rho^2 d\theta = \frac{9}{2}\int_0^{\frac{\pi}{2}} \frac{\sin^2\theta\cos^2\theta}{(\sin^3\theta+\cos^3\theta)^2} d\theta$$
$$= \frac{9}{2}\int_0^{\frac{\pi}{2}} \frac{\sin^2\theta\cos^2\theta}{\cos^6\theta}\cdot\frac{d\theta}{\left(\frac{\sin^3\theta}{\cos^3\theta}+1\right)^2}$$
$$= \frac{9}{2}\int_0^{\frac{\pi}{2}} \frac{\operatorname{tg}^2\theta\cdot d\theta}{\cos^2\theta(1+\operatorname{tg}^3\theta)^2}.$$

4.2. ANALYSE D'UN TEXTE DE RESOLUTION DE PROBLEME

4.2.1. La modélisation des données

4.2.1.1. Le problème dans son contexte

Nous analysons un problème de calcul de surface résolu en exercice dans le traité de J. Quinet [Quinet 62]. Le document avec notre découpage est présenté ci-contre. Le problème dit "Surface de la parabole" donne lieu notamment à un sous-problème qui détermine l'équation en coordonnées polaires de la parabole. La structure de ce document nous parût étrange en première lecture, toutefois les justifications de son organisation se sont révélées après une journée d'analyse.

Il s'agit globalement d'un texte de démonstration qui pose un problème et le résoud à des fins pédagogiques. L'aspect "preuve", c'est-à-dire l'enchaînement mathématique des déductions retenues pour la résolution, y est essentiellement laissé de coté. Nous nous intéressons ici à deux points :

- une analyse du document et de ses enchaînements ;
- une analyse des motivations mathématiques sous-jacentes.

La numérotion "4°" et l'en-tête de mise en page nous rappellent que ce problème s'insère au chapitre 11 intitulé "calcul des surfaces", qui débute par le canevas :

"49. Cas où la courbe est sous la forme y=f(x)." 129
"50. Cas où la courbe est sous forme paramétrique." 134
"51. Cas où la courbe est en coordonnées polaires (fig. 30)." 139
"1° Surface du cercle : r = Cte = R." 140
"2° Surface d'une boucle de la lemniscate (fig. 32)."
$\rho^2 = a^2 . \cos 2\theta$
où θ varie de $-\pi/4$ à $+\pi/4$ 140
"3° Surface de la cardioïde (fig. 33)." 141
"4° Surface de la parabole (fig. 34), $y^2 = 2px$." 142

Ce problème est donc un cas particulier d'**un problème plus général** : Calculer la surface définie par une courbe en coordonnées polaires. Comme cette formulation l'indique, les données principales du problème général sont donc :

- la surface
- la courbe
- le repère

Le problème est donc d'abord géométrique, mais la solution est algébrique : l'exploitation des données constitue donc un transfert progressif d'une représentation géométrique vers une représentation algébrique. Pour cela, le choix d'un repère et d'une courbe détermine une équation de la courbe.

J. Quinet présente p 139-140 la **solution générale** au problème du calcul d'une surface engendrée par une courbe dont l'équation en coordonnées polaires est exprimée sous forme $\rho = f(\theta)$:

$$S = \int_{\theta_1}^{\theta_2} \frac{1}{2} \rho^2 \, d\theta$$

Cette solution générale se comporte véritablement comme un théorème à appliquer pour résoudre le problème. Le mathématicien connaît pour cela l'interprétation géométrique des paramètres.

A partir des données géométriques et algébriques, J. Quinet adopte invariablement dans les problèmes de son traité, une **conduite du raisonnement** respectant la démarche suivante :

1) introduction des données ;
2) choix d'un type d'équation, étude et caractérisation des paramètres manquants ;
3) écriture de la formule de la surface, instanciée avec les paramètres du problème ;
4) calcul de la formule de la surface ;
5) discussion éventuelle de ce résultat (interprétation géométrique, cas particulier, etc.) ;

La caractérisation de la démarche de J. Quinet n'est qu'une première étape de notre analyse. Nous voulons détailler cette démarche de conduite du raisonnement afin de **l'expliquer** en fonction des données d'un problème et de la solution générale.

Nous cherchons donc à mieux connaître l'organisation des données et celle de la résolution du problème. Nous voulons aussi déterminer les principes qui régissent la présentation apparente d'une démonstration. Nous faisons ainsi la nuance entre la ***description*** des données et leur ***présentation*** dans la démonstration. La description concerne le lieu et l'ordre dans lequel les données sont introduites, tandis que la présentation concerne leur mise en forme dans le document.

En connaissant plus finement une démonstration du point de vue du mathématicien qui la produit, nous pourrons préciser les diverses connaissances et leur exploitation pour la résolution, pour la présentation dans une démonstration, ainsi que pour le suivi d'une démonstration.

4.2.1.2. L'énoncé du problème

L'énoncé du problème est principalement résumé dans le titre :

> 4° Surface de la parabole (fig. 34), $y^2 = 2px$.

Ce texte n'est pas un énoncé de problème. De plus, la fig. 34 ne présente aucune surface !

L'énoncé exploite une complémentarité de présentation visuelle et langagière. La formulation de l'énoncé suppose une bonne familiarité du lecteur avec le concept de parabole, l'interprétation des figures, le calcul de surfaces, la correspondance géométrique-algébrique, etc.

Bien qu'elle soit aussi mentionnée dans la figure, la duplication de l'équation algébrique "$y^2 = 2px$" en fin du titre favorise l'indépendance du texte. Elle permet notamment de paraphraser l'énoncé dans une représentation totalement algébrique :

> Soit la parabole d'équation $y^2 = 2px$, calculer la surface ayant la description suivante :

Paradoxalement, cette donnée principale qu'est la surface n'est pas précisée dès le début, mais seulement lorsque le contexte le permettra. La formulation d'un énoncé ou d'un document mathématique suit ainsi une **stratégie de description** privilégiant ici la courbe. Il est notamment envisageable de commencer par la surface.

En attendant la complétion des données du problème, le mathématicien se contente de l'existence d'une surface à calculer qu'il sait être définie à partir de la parabole. Il suffit que J. Quinet décrive la surface avant le point 3) de sa démarche : l'écriture de la formule de la surface. Nous qualifions cet ordre de description d'***opportuniste***, car il gère l'indétermination et s'adapte à la nature des données. Par exemple, la surface est précisée dès le début de l'énoncé si elle peut être décrite à partir de propriétés évidentes, comme dans le cas d'une boucle.

Le mathématicien n'est pourtant pas désorienté par l'incomplétude de l'énoncé. Son savoir-faire lui permet d'envisager un certain nombre de surfaces potentielles, définies à partir des courbes. Dans notre cas l'interprétation de l'énoncé peut, par exemple, correspondre aux choix suivants :

"surfaces" de la parabole

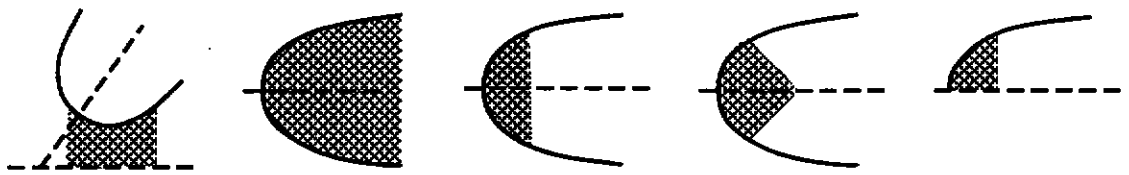


figure 1.16

Le premier choix correspond usuellement à la surface d'une courbe quelconque. Le second correspond à la surface d'une courbe fermée. Les deux autres choix proposent des surfaces réalistes, tandis que le dernier est le choix retenu dans le premier exercice de ce chapitre 11 : son exploitation étant exclusivement algébrique, la surface n'avait même pas été figurée...

Nous retrouvons ici, sous leur forme la plus simple, les ingrédients recensés dans l'analyse des problèmes du chapitre. L'énoncé d'un problème de calcul de surface comporte :

- le nom de la (ou des) *courbe de référence*, avec éventuellement son intérêt pratique et sa construction ;
- une figure représentant les *données géométriques*, avec si besoin la description langagière de la surface à calculer. Elle est caractéristique de l'exploitation d'un raisonnement géométrique dans le problème ;
- l'*équation de la courbe* comportant le nom et les caractéristiques des paramètres. Elle peut éventuellement être calculée à partir des données géométriques ;

La Fig. 34. de l'exemple décrit les données du problème ; elle :

- représente les lieux géométriques représentant la parabole, qu'elle désigne ici par son équation algébrique ;
- définit et désigne un repère (O, x, y) que J. Quinet choisit de façon à simplifier l'équation algébrique de la parabole (il pourrait privilégier la construction et prendre la droite D comme axe (0,y)!) ;
- présente les paramètres principaux de la modélisation d'une parabole.

Cette figure est à notre avis illustrative de l'importance de la courbe et des connaissances qui s'y rattachent dans le contexte d'un problème qui n'est même pas bien défini. Les paramètres présents sur cette figure servent de mise au point didactique et mémorielle pour le lecteur.

En résumé, la présentation d'un énoncé dépend de la nature du problème et de ses données. La **maîtrise de concepts mathématiques usuels** est nécessaire pour l'interpréter. La modélisation des concepts est donc essentielle pour la prise en compte des données mathématiques.

4.2.1.3. Modélisation d'un réseau de concepts

La modélisation des concepts est **particulièrement difficile** car elle est fort imbriquée et de nombreux éléments déterminants restent à expliciter. De plus, la simple définition d'un concept n'est pas immédiate. Par exemple le concept de surface fait appel à notre intuition géométrique des sous-ensembles du plan, mais rien ne précise formellement ce qui définit une surface et sous quelles conditions elle est bien définie.

Nous pouvons ainsi résumer le concept d'équation tel que nous l'utilisons ici, car il illustre que l'essentiel reste à modéliser. Une équation :

- fait partie d'une représentation algébrique ;
- est une relation algébrique entre des variables, une variable privilégiée dite inconnue étant choisie ;
- peut être rattachée à un objet géométrique dont la nature (droite, cercle, etc.) contraint la forme de l'équation ;
- nécessite le choix d'un repère pour être en correspondance bijective avec un objet géométrique ;
- a une forme simplifiée attendue dépendant de l'objet géométrique et du repère ($\rho = f(\theta)$) ;
- sert dans tout usage algébrique de l'objet géométrique correspondant ;

La modélisation des concepts mathématiques est à notre avis une entreprise à long terme nécessitant encore de nombreuses études. Nous ne proposons pas, par exemple, de modélisation de la parabole capable de rendre compte des données et connaissances de ce problème. Nous en constatons la difficulté et en discutons quelques thèmes.

Un ***concept*** appartient à l'univers mathématique, et est indépendant d'un formalisme de mise en œuvre. Il est utilisé pour regrouper des informations qui lui sont rattachées par l'expérience des mathématiciens. Il regroupe ainsi des informations provenant de modes d'utilisation différents. Un ***mode d'utilisation*** correspond à un usage possible du concept.

Ces modes d'utilisation sont de deux types : *descriptif* et *constructif*. Un mode descriptif consiste à étudier un objet. Il s'attache à catégoriser ou à trouver les propriétés d'un objet représentant éventuellement ce concept. Ce n'est donc pas un mode inactif ou statique : il y a actitivé. Le mode constructif s'inscrit dans l'action du mathématicien afin de produire un objet représentant de ce concept. L'activité est ici la production d'une construction à partir d'objets mathématiques.

Une façon d'envisager ces modes d'utilisation est de dire que le mathématicien ne s'intéresse qu'à un aspect du concept à la fois. Par exemple le mathématicien peut adopter un mode d'utilisation correspondant à la description visuelle du lieu géométrique des points de la parabole. Il peut adopter un mode d'utilisation correspondant à la construction géométrique de la parabole. Ce mode se subdivise selon la nature des opérations disponibles (règle et compas, tansformations mathématiques, etc.). Un autre mode d'utilisation constructif consiste par exemple à trouver l'équation de la parabole, ce qui est une façon de la définir algébriquement.

Nous ne sommes plus maintenant au niveau de l'activité du mathématicien mais au niveau (méta) d'une **modélisation de cette activité**. Le concept est alors un regroupement d'informations dont nous distinguons les *aspects statiques* et les *aspects dynamiques* qui servent à la mise en œuvre d'un mode d'utilisation. Ces deux termes sont réservés à la modélisation de l'activité, alors que "descriptif" et "constructif" sont réservés à la nature de l'activité. Nous dirons que lorsque le mathématicien choisit un mode d'utilisation dans son activité, cela se traduit dans la modélisation par l'adoption d'un *point de vue* n'envisageant que certains aspects de la modélisation.

Nous détaillons d'abord les **aspects statiques de la modélisation** que nous réduisons à la donnée de paramètres caractéristiques et de propriétés stratiques. Les paramètres caractéristiques de la parabole sont décrits Fig. 34 :

- le foyer F (qui est un point) ;
- la droite directrice D ;
- la distance de F à D, représentée par le paramètre p ;
- l'axe de symétrie ;
- un point caractéristique appelé sommet ;
- un axe tangent à la courbe au sommet ;
- une équation définie dans un repère (O,x,y) ;

Parmi ces sept points, les trois premiers sont des paramètres de construction de la parabole, alors que les quatre autres sont des paramètres de description. Une particularité de la parabole est que ses paramètres de construction sont visuellement cachés, et donc distincts des paramètres utilisés dans des actions de description.

Pour énoncer ces paramètres, le concept de parabole fait explicitement appel aux concepts de :

point, droite, axe, distance, symétrie, être tangent, équation, repère.

Il fait de plus implicitement appel à des concepts comme :

courbe, courbe ouverte, conique, surface engendrée, plan, ensemble de points, lieu géométrique, point quelconque, repère cartésien, coordonnées, etc.

Ces paramètres caractéristiques s'accompagnent d'une richesse langagière indispensable pour la désignation et la manipulation dans l'activité mathématique, et donc indispensable dans la modélisation du concept de parabole. Il y a notamment ici :

l'équation de la parabole, l'axe de symétrie, le foyer, le paramètre, le sommet, la droite directrice, F, D, p.

Les propriétés statiques énonçables sont de plusieurs types, selon qu'elles sont relatives par exemple à la :

- forme
  continue, connexe, symétrique, fermée, etc.
- mesure
  distance, valeur, égalité via équidistance, angle, etc.
- nature ensembliste
  appartenance, inclusion, intersection, identité, etc.
- transformation
  rotation, homothétie, etc.

Les **aspects dynamiques de la modélisation** du concept de parabole consistent à permettre l'exploitation de ces aspects statiques. Par exemple, ils peuvent servir à la détermination des propriétés géométriques de description de la parabole à partir de la seule donnée de son équation. C'est ce que les mathématiciens appellent habituellement l'étude de la courbe. Ils regroupent des concepts spécifiques et interviennent par exemple pour déterminer l'asymptote d'une branche infinie.

Un autre exemple d'aspect dynamique est celui de l'exploitation des paramètres caractéristiques pour la construction d'un point quelconque de la parabole, et donc virtuellement du lieu géométrique des points de la parabole. Cet aspect dynamique exploite donc une propriété de définition de la parabole via une propriété de chacun de ses points.

Un autre aspect dynamique est alors la gestion de la cohérence entre paramètres caractéristiques. Elle concrétise les liens mathématiques entre ces paramètres. Par exemple, pour déterminer le paramètre p il suffit de connaître l'équation ou la distance de F à D.

Les informations statiques et dynamiques, relatives au concept de parabole ne sont pas toutes spécifiques à la parabole. C'est pourquoi des **hiérarchies conceptuelles dans la modélisation** permettent de rattacher les informations au concept le plus général auxquelles elles s'appliquent.

Considérons une hiérarchie triviale parabole-courbe. Par nature, la détermination des paramètres descriptifs de la parabole sont valables pour toutes les courbes. Ces aspects dynamiques sont donc rattachés au concept de courbe plutôt qu'à celui de parabole. Par contre, le résultat de la description de la parabole est spécifique à la parabole et s'y trouve rattaché.

Les hiérarchies conceptuelles mathématiques sont complexes et introduisent beaucoup de nuances. Par exemple, il peut y avoir un concept intermédiaire : celui de conique. C'est le lieu des points dont le rapport des distances à un point (*foyer*) et à une droite (*directrice*) a une valeur donnée (*excentricité*). La mise à disposition du concept de conique fait migrer la quasi-totalité du rattachement des paramètres descriptifs de la parabole vers le concept de conique. Cela met en évidence des propriétés communes de la parabole, de l'ellipse et de l'hyperbole.

Ce rattachement des paramètres se fait avec extension de leurs caractéristiques. Par exemple, il y a pour la conique la possibilité d'avoir aussi deux sommets. De plus ce rattachement met en évidence dans notre description un paramètre supplémentaire "caché" : l'excentricité, qui a pour valeur 1 dans la parabole. Il y a alors des **principes d'extension et d'utilisation des hiérarchies**.

Ainsi, même s'il connaît cette hiérarchie avec la conique, le mathématicien peut choisir de ne pas s'y référer. C'est même une pratique courante en mathématique où les paramètres à prendre en compte sont réduits au minimum nécessaire. L'excentricité n'est alors pas explicitement abordée et la définition de la construction de la parabole est alors une variante spécifique : le lieu des points équidistants à un point (*foyer*) et à une droite (*directrice*).

Cette complexité des relations hierarchiques entre concepts se retrouve dans les **autres liens d'association** qui existent entre concepts et dépendent d'un mode d'utilisation. Par exemple, nous avons dans notre problème une relation entre les concepts de courbe et de surface engendrée par une courbe, qui traduit un lien de causalité souvent externe aux mathématiques. Par ailleurs, les concepts de courbe et de surface sont respectivement associés à celui de point quelconque, par l'intermédiaire du concept d'ensemble de points.

Ces liens d'association sont le support privilégié de problèmes mathématiques et de connaissances de (ré)solution qui les accompagnent. Nous avons vu précédemment pour la parabole diverses possibilités de surfaces engendrées. Dans un mode d'utilisation où une courbe en coordonnées polaires engendre une surface, l'expression générale de la surface a été calculée. Sinon, les connaissances mathématiques de

résolution de problème examinent la possibilité de se ramener aux cas généraux. Ces liens d'association déterminent enfin la nature et la forme de l'énoncé des problèmes. La typicité de notre énoncé actuel en est un indice indiscutable.

La modélisation du réseau de concepts est donc indispensable pour l'organisation des **données**, pour les **aspects linguistiques** et pour les connaissances de **résolution de problème** utilisées en mathématiques.

Elle doit malgré tout se contenter d'expédients faute de simplement modéliser correctement le mathématicien dans son activité. Notre savoir-faire actuel limite les modélisations envisageables, et ce faisant les applications possibles.

4.2.1.4. Manipulation textuelle des concepts

J. Quinet "profite" du problème pour introduire un sous-problème a priori inopportun : "trouver l'équation de la parabole en coordonnées polaires, en prenant le pôle au foyer F". Ici l'auteur :

- pose implicitement le sous-problème ;
- utilise un alibi didactique pour présenter une méthode : "Profitons de cet exemple pour montrer au lecteur comment..." ;
- précise un des paramètres géométriques nécessaires à cette transformation ;

Ce sous-problème constitue le développement principal du point 2) de la démarche de J. Quinet :

2) choix d'un type d'équation, étude et caractérisation des paramètres manquants ;

Le lecteur doit utiliser ici sa **connaissance préalable du réseau de concepts** et des liens entre courbe et surface. Ce sous-problème s'explique par le développement de la solution générale d'une surface engendrée par une courbe en coordonnées polaire. En fait, l'introduction de ce sous-problème est nécessaire car l'équation $\rho = f(\theta)$ n'est pas connue.

L'objectif suivi et les connaissances indispensables à cette résolution sont connues, car introduites en début de la section 51 de notre exemple :

"51. Cas où la courbe est en coordonnées polaires (fig. 30)." 139

Cette Fig. 30 permet de recenser les composantes mathématiques indispensables pour le **mode d'utilisation** d'une "courbe en coordonnées polaires" ainsi que leurs conventions de désignation :

- une *courbe* regroupant des points et définie par une équation. La courbe n'a pas forcément de nom spécifique mais est désignée par C si besoin ;
- un *point quelconque* de la courbe, référencé par un symbole. J. Quinet emploie indifféremment P ou M dans son traité, et utilise des "prime" comme dans P' et P" pour marquer une correspondance ;
- un *point fixe* appelé pôle et référencé par convention par le symbole O ;
- un *rayon vecteur* reliant ces deux points, dont la longueur est notée ρ ;
- une Origine qui est un *axe de coordonnées* issu du point fixe O (attention, pour l'auteur l'Origine est un axe) ;
- un *angle* formé par le rayon vecteur avec l'Origine, et désigné par θ.

Cette description est intéressante par ailleurs, car elle montre comment le réseau de concepts est étendu à partir d'un réseau antérieur. Les quatre derniers points sont spécifiques et caractérisent l'appellation "être en coordonnées polaires" pour une courbe. Ils sont construits intégralement à partir du réseau antérieur. Les autres points sont alors partagés et récupérés du réseau antérieur. Des particularités sont néanmoins rajoutées, puisque l'équation de la courbe s'écrit maintenant $\rho = f(\theta)$. Les extensions proposées le sont par besoin pratique et non nécessairement par volonté de généralisation. Par exemple, l'équation pourrait s'écrire ici $R(\rho,\theta)=0$ (relation implicite) mais $\rho = f(\theta)$ suffit à caractériser la classe de problèmes qui intéresse l'auteur.

L'instanciation des composantes de ce mode d'utilisation "courbe en coordonnées polaires" permet **d'évaluer l'état de l'introduction des données dans cette démonstration.** L'examen des paramètres indispensables connus à cet endroit donne :

- courbe
  Elle est spécialisée au concept de parabole, et instanciée pour tout le problème à la parabole décrite selon un autre point de vue : "courbe en coordonnées cartésiennes". L'équation et le système de coordonnées qui lui correspond sont ainsi à transformer en l'équation en coordonnées polaires attendue.
  Les **composantes dynamiques de la modélisation** sont alors spécialisation, instanciation et gestion de points de vues différents sur un objet.
- point fixe
  Le point fixe est ici précisé grâce à son appellation "pôle", issu du point de vue "courbe en coordonnées polaires", relativement à un objet géométrique existant "foyer F", issu de la modélisation de la parabole.
  La **terminologie** joue donc un rôle indispensable dans les correspondances conceptuelles requises pour l'exploitation des composantes dynamiques de la modélisation.
- les quatre autres restent à préciser...

C'est justement l'objet de la phrase suivante, que nous avons qualifié de données du sous-problème. Ces données sont introduites avec leurs symboles de désignation, car elles bénéficient d'une présentation mixte géométrique-algébrique.

La donnée principale, ou inconnue du problème, est le rayon vecteur. Il est introduit immédiatement dans le texte, tandis que les données secondaires nécessaires à sa caractérisation sont introduites avec leurs propriétés au moyen de la Fig. 35. Cette Fig. 35 représente les points appartenant aux diverses courbes, et montre en particulier que AM⊥D, MM'⊥(O,x) et AM=MP.

Les quatres marqueurs caractéristiques du point de vue de "courbe en coordonnées polaires" : le pôle, le rayon vecteur, l'Origine et l'angle formé par le rayon vecteur avec l'Origine sont ainsi introduits dans des situations différentes (corps du sous-problème, texte ou figure décrivant les données). Cela indique une **collaboration entre ces champs textuels**, ainsi qu'un raisonnement du mathématicien respectivement sur le texte, sur les figures et sur leurs correspondances.

La déclaration « Appelons ρ le rayon vecteur MP » est alors un moyen d'introduire la description des données du sous-problème de l'équation, tout en préparant l'introduction de la figure comme support de présentation privilégié. Cette déclaration permet d'omettre ρ sur la Fig. 35 alors que θ y est indiqué. La figure représente alors un choix d'informations utiles dont le dosage s'apparente aux principes énoncés par H.P. Grice [Grice 75].

La connaissance du réseau de concepts permet donc de **déterminer finement les données manquantes** du problème à un emplacement de la démonstration. Par contre, elle ne dit rien sur la résolution de problème ni sur la stratégie d'introduction des données, qui privilégie ici la description complète de la courbe. Nous qualifions la stratégie adoptée pour cette démonstration d'opportunisme excessif, car le problème n'est toujours pas complètement posé !

4.2.2. La structure d'une démonstration

4.2.2.1. Structure apparente et structure profonde

Nous qualifions de **structure profonde de cette démonstration**, le couple composé du problème bien posé avec toutes ses données, et de sa résolution en terme de tâches de résolution de problème, exploitant ici la solution générale. Nous qualifions de **structure apparente de cette démonstration** la façon dont le mathématicien choisit d'exposer son problème en introduisant progressivement les données, en ne présentant que certaines tâches, et en choisissant une mise en forme textuelle particulière.

Nous avons envisagé prioritairement jusqu'ici la structure apparente de la démonstration. Les données présentées dans le document peuvent s'expliquer à partir d'un réseau de concepts. Elles correspondent à la complétion du mode d'utilisation d'une "courbe en coordonnées polaires". Toutefois, cette explication est insuffisante.

Par exemple, l'analyse de la structure apparente de la démonstration montre que la justification de AM=MP (relation indiquée Fig. 35) s'effectue par un rappel du procédé général de construction d'une parabole : « la parabole est le lieu géométrique des points M équidistants d'un point fixe P, appelé foyer, et d'une droite D appelée directrice. ». Ce rappel suit immédiatement la référence à la Fig. 35 et vient compléter sa description. Il fournit un argument manquant au raisonnement géométrique apparent.

L'intérêt du rappel d'une telle définition peut être :

- argumentatif : justifier que AM = MP.
  Cela est indiqué sur la figure mais l'argument étant strictement constructif, il n'est pas déductible des propriétés géométriques apparentes (dites descriptives).
- pédagogique : rappeler une propriété de la parabole.
  Cette seule composante présenterait peu d'intérêt vu le niveau supposé du lecteur et les fortes allusions à cette propriété déjà effectuées figures 34 et 35.
- terminologique : introduire la définition d'un foyer et de la directrice.
  Ces termes ont déjà été utilisés précédemment...
- désignatif : définir le sens des symboles M, P et D.
  Cela permet de compléter la définition de ρ sur un plan textuel afin d'être indépendant de la description présentée Fig. 35. Ce serait contradictoire avec la suite : « La figure donne ».

Aucun de ces arguments n'est réellement satisfaisant. En effet, l'intérêt principal de ce rappel ne peut être défini à partir du document, mais à partir des tâches de résolution du problème par le mathématicien. Ces tâches permettent un autre abord de la démonstration. Il faut repartir pour cela de la solution générale aux données du problème :

$$S = \int_{\theta_1}^{\theta_2} \frac{1}{2} \rho^2 \, d\theta$$

L'exploitation de cette solution générale engendre les tâches suivantes :

- déterminer θ_1 tel que θ_1 soit l'angle... ;
- déterminer θ_2 ;
- déterminer $f(\theta)$ tel que $\rho = f(\theta)$;
- produire une formule nouvellement instanciée ;
- calculer la formule instanciée ;

La tâche de difficulté principale est alors de déterminer $\rho = f(\theta)$. Elle se décompose en :

- trouver une relation entre ρ et θ ;
- mettre la relation sous forme $\rho = f(\theta)$

Le mathématicien a deux grandes façons de procéder pour trouver une relation entre ρ et θ :

- algébrique, en effectuant un changement de variables à partir de l'équation de la parabole en coordonnées cartésiennes ;
- géométrique, en utilisant directement les propriétés géométriques de construction de cette parabole.

J. Quinet choisit la voie géométrique, qui est plus "astucieuse", ne nécessite pas au préalable l'équation en coordonnées cartésiennes, et relève de plus d'une méthode générale aux coniques. Cette résolution du sous-problème utilise donc un raisonnement géométrique pour aboutir à un résultat algébrique.

Il est alors possible de justifier la démonstration par sa structure profonde : l'exploitation de la relation AM = MP. La **démarche de raisonnement** du mathématicien est ainsi :

- expression de la relation géométrique : la parabole est le lieu des points équidistants à un point (*foyer*) et à une droite (*directrice*) (cette connaissance conceptuelle se traduit en langage écrit) ;
- instanciation de la relation au moyen d'une figure, puis de désignations langagières M, P, A : cela correspond à une introduction de données auxiliaires similaire à l'élimination d'un quantificateur existentiel ;
- expression de la relation entre ρ et θ :
  - première façon d'exploiter AM = MP ;
  - deuxième façon d'exploiter AM = MP ;
  - synthèse des résultats obtenus.

La structure profonde de la démonstration à l'aide de tâches de résolution de problèmes est alors la raison principale de l'expression de la relation géométrique du document : « la parabole est le lieu géométrique des points M équidistants d'un point fixe P, appelé foyer, et d'une droite D appelée directrice. ».

4.2.2.2. L'obtention d'une relation algébrique

Le raisonnement pour trouver l'expression de la relation entre ρ et θ est ici entériné par "La figure donne". La démarche de raisonnement adoptée pour le choix de la présentation apparente traduit un **transfert d'influence progressif** du raisonnement visuel vers le raisonnement langagier. Cela donne :

- raisonnement géométrique visuel, sur la figure et ses données géométriques ;
- transcription dans un langage textuel avec des données géométriques (MA, PB, etc.) ;
- transcription dans un langage textuel avec les données algébriques correspondantes ;
- raisonnement déductif, aboutissant à une relation algébrique ;

La **transcription** constitue un changement de représentation ou de langage qui requiert des inférences. Le cas le plus simple est celui de la transcription géométrique-langagière, qui fait correspondre une représentation géométrique intuitive (distance de M à A) à sa transcription langagière (MA).

La transcription du raisonnement géométrique donne MA=M'B=PB-PM'. Il serait abusif de supposer que l'égalité M'B=PB-PM' puisse se dispenser de la figure, car elle permet la justification de sa validité (P, M' et B sont alignés) comme la motivation de l'introduction de P.

L'étape suivante de la transcription algébrique est la plus délicate :

- la transcription de MP en ρ est l'exploitation d'une donnée textuelle précédente ;
- la transcription de PB en p se fait par correspondance conceptuelle entre le paramètre p et la distance du foyer à sa projection orthogonale sur la droite D. Cette transcription est facilitée par la correspondance visuelle entre la Fig. 34 et la Fig. 35 ;
- la transcription de PM' en coté du triangle rectangle (M, M', P) pour aboutir en $\rho\cos(\pi-\theta)$ requiert un raisonnement géométrique sur les angles et leur interprétation algébrique.
- la transcription de PB-PM' en $p-\rho\cos(\pi-\theta)$ requiert de plus des inférences sur le devenir et la correspondance des sous-termes respectifs.

Chaque égalité traduit ainsi des inférences que le mathématicien ajuste en fonction des compétences supposées du lecteur. Ces égalités sont alors exploitées au niveau de l'argumentation mathématique. Elles y sont introduites ("La figure donne") et corrélées (or, d'où).

Si la syntaxe des notations aide beaucoup dans certains problèmes algébriques, les notations utilisées ici exploitent essentiellement la **compréhension sémantique du mathématicien** dans leur utilisation. Cette compréhension a permis au mathématicien de partir des points M et P, et transiter par leur distance afin d'aboutir à une valeur ρ.

Cette compréhension sémantique explique mieux la tolérance aux ambiguïtés syntaxiques et autres exceptions dont voici quelques apercus dans le document :

- l'usage simultané de A, B et D pour désigner des objets de nature différente ;
- aucune corrélation ici entre f et F car ils sont sémantiquement sans rapport ;
- l'ambiguïté de la notation MP qui désigne à la fois un rayon vecteur et une distance (ceci n'a plus lieu dans les notations géométriques utilisées actuellement dans les collèges et lycées) ;
- une erreur de choix de notation de l'auteur, qui désigne le point fixe appelé foyer par P alors qu'il utilise aussi F ;
- l'usage d'un même symbole θ_1 dans deux contextes différents amenés à se correspondre (borne inférieure de la formule générale de la surface, et borne supérieure de la parabole) ;
- une forme d'équation "ρ=f(θ)" ou "y=f(x)" qui fait une exception pour l'équation de la parabole "$y^2=2px$". La forme réduite "y=" aurait nécessité deux cas...

4.2.2.3. La structure apparente de la démonstration

Ce passage de recherche de la relation sous forme ρ = f(θ) est aussi intéressant pour la **gestion locale de démonstration**. Lorsque l'on ne conserve que les extrémités, la gestion locale de la démonstration produit des "déductions logiques" comme :

$MA = p-\rho\cos(\pi-\theta)$
or $MA = \rho$,
d'où $p-\rho\cos(\pi-\theta) = \rho$.

L'utilisation d'une telle règle dans notre problème requiert, ici encore, des inférences qui en constituent la difficulté majeure. Outre la connaissance des règles possibles, il faut ici, pour assurer le **suivi de la démonstration** :

- isoler les extrémités des égalités successives ;
- reconnaître que les termes initiaux sont communs ;
- diagnostiquer la règle employée ;
- déduire la conclusion de la règle (après "d'où") ;
- comparer la conclusion attendue "$p-\rho\cos(\pi-\theta) = \rho$" à la conclusion effective :

  $\rho = p+\rho\cdot\cos\theta$

- trouver l'inversion et les différences respectives ;
- justifier ces différences par des inférences à trouver, ici trouver notamment que $\cos(\pi-\theta) = -\cos\theta$.

La donnée des règles de déduction ne suffit pas,

l'essentiel des inférences est consacré à leur mise en œuvre locale.

Cette gestion locale de démonstration obéit à des motivations plus générales. Pour ce qui est de l'ordre d'introduction, J. Quinet pourrait avoir préféré une inversion des deux premières lignes :

$MA = MP = \rho$
or $MA = PB-PM' = p-\rho\cos(\pi-\theta)$,
d'où $\rho = p+\rho\cdot\cos\theta$.

Cette présentation offre en effet l'avantage d'éviter une inversion de l'ordre d'apparition des résultats (ρ puis $p-\rho\cos(\pi-\theta)$). Par contre, elle met au second plan le raisonnement géométrique principal, introduit par "La figure donne". Cet argument explique l'inversion choisie par l'auteur.

Mais les explications majeures d'un choix de règle de déduction sont relatives à la structure profonde de la démonstration. Par exemple, pourquoi aussi ne pas avoir écrit tout simplement :

$$\rho = MP = MA = PB-PM' = p-\rho\cos(\pi-\theta)$$

Cette solution supprimerait le recours au rupteur d'inférence "or". Toutefois, cette démarche d'inspiration algébrique est à contre courant de la méthode choisie et de la gestion des tâches précédemment décrite qui décompose MA=MP de deux façons.

Finalement, nous aurions préféré, pour la **mise en évidence de cette structure profonde**, une présentation comme :

on a la relation :
$MA = MP$,
or la figure donne :
$MA = PB-PM' = p-\rho\cos(\pi-\theta)$,
d'autre part, on a par définition :
$MP = \rho$,
d'où $\rho = p+\rho\cdot\cos\theta$.

Le reste des calculs correspond à une conduite du raisonnement mieux connue et plus facilement maitrisée. Les principes stratégiques dégagés par [Bundy 75] s'appliquent ici à partir de $\rho = p+\rho\cdot\cos\theta$:

- collection des termes contenant l'inconnue ρ dans le membre de gauche, afin de préparer la forme simplifiée attendue $\rho=f(\theta)$;
- réduction de ses occurences par une mise en facteur afin d'obtenir la formule du document : $\rho(1-\cos\theta) = p$;
- isolement de l'inconnue ρ et simplification éventuelle du membre droit.

La formule finale $\rho = \frac{p}{1-\cos\theta} = f(\theta)$ présente une caractéristique intéressante : pourquoi rajouter "$= f(\theta)$" ?

L'intérêt d'une telle notation est souvent une définition implicite de la fonction f. f est ici utilisé nulle part ailleurs. Il s'agit ici de marquer la composante stratégique implicite qui a guidé ce sous-problème. J. Quinet ne cherche pas à éclairer un lecteur distrait en lui signalant qu'il a obtenu la forme désirée. En

explicitant ainsi l'équation cherchée, J. Quinet marque la fin du suivi des calculs et indique la remontée des centres d'intérêt au niveau global du problème !

Un choix de description fait ainsi intervenir des critères d'ordre créatif, c'est-à-dire difficilement modélisables. De plus, cet aspect est très peu étudié. Des raisons possibles d'un tel choix sont :

- la difficulté heuristique qui entraîne une démarche exploratoire, comme développer une égalité par les deux cotés ;
- des justifications argumentatives, comme privilégier MA=MP alors qu'un parcours direct depuis ρ était possible ;
- des critères de présentation et de proximité d'arguments, comme "la figure donne" ;
- la longueur du raisonnement, qui entraîne un calcul long ou trivial, et autorise ou non des points laissés en suspens dans la gestion des tâches.

Cette résolution du sous-problème a mis en évidence l'importance de la **démarche du mathématicien dans la conduite du raisonnement**. Cette démarche est définissable en fonction de la nature des données mathématiques concernée. Elle conjugue des objectifs conceptuels plus ou moins profonds en fonction de ces données.

4.2.2.4. La résolution du problème principal

La phrase d'introduction "La surface du secteur hachuré, (fig 36), sera donc" marque la transition entre la description des données et le début de la résolution. Elle est l'analogue de "La figure donne" du sous-problème. Le futur "sera", rare en mathématiques, marque ici le début d'un calcul long concernant un même objet.

La description de la surface ayant été différée jusque là, l'auteur profite de cette phrase introductive pour la compléter. L'examen de la Fig. 36 permet de remarquer que :

- elle représente la surface comme un secteur de courbe jusqu'à un angle θ_1 ;
- elle se présente comme un agrandissement de la Fig. 34 (x, y, O, F, et leurs deux points gras) ;
- elle exploite la connaissance de la Fig. 35 par le choix d'un angle qui aurait très bien pu être intérieur au secteur hachuré, et comme la Fig. 35, elle ne représente que la demi-parabole utile ;
- elle ne présente aucune information inutile à ce stade, comme un point M ou la droite directrice.

Le calcul des bornes du paramètre principal est traditionnellement inclus à ce stade de la description. Il est ici omis car un raisonnement géométrique et visuel simple donne π et θ_1.

Les points à compléter de la démarche de J. Quinet sont maintenant :

3) écriture de la formule de la surface, instanciée avec les paramètres du problème ;
4) calcul de la formule de la surface ;
5) discussion éventuelle de ce résultat (interprétation géométrique, cas particulier, etc.) ;

La première ligne correspond au point 3), et la dernière au point 5). Le rappel de l'écriture de la formule se fait selon un **schéma de présentation** intangible :

- symbole de désignation de la surface (S =)
- formule générale à utiliser avec les bornes instanciées ;
- formule avec les inconnues substitués (ici ρ^2) et les coefficients simplifiés et groupés en tête ;

La suite de la résolution comporte des calculs jusqu'à ce que tous les termes puissent être intégrés et le résultat obtenu simplifié. Ici aussi, la **conduite des calculs** est assez bien connue. Nous ne préciserons ici que certains traits caractéristiques :

- les changements de variables éventuels :
  Leur détermination est la difficulté mathématique majeure de la résolution, un mauvais choix entraînant des complications calculatoires importantes. La maitrise des "bons " changements de variables pour les fonctions trigonométriques fait l'objet du chapitre VIII du traité de J. Quinet. Le choix effectué est explicité et donne lieu à des préparatifs auxiliaires afin d'en tirer les conséquences.
- le suivi des calculs :
  Il impose que les règles de transformation utilisées et leurs conditions d'utilisation soient reconnaissables pour un mathématicien à la seule lecture du texte. La détermination de la validité des formules et des calculs est souvent laissée à la charge du lecteur (par ex. vérifier que $1-\cos\theta \neq 0$).
- le découpage des tâches :

J. Quinet présente le déroulement des transformations de la formule de la surface. Les calculs auxiliaires sont isolés (changement de variable), omis ou repoussés. Par exemple, les bornes de $t=tg(\theta/2)$ ne sont pas calculées ; ce calcul serait repoussé en fin de résolution si un changement de variable réciproque ne le rendait inutile.

Une caractéristique importante de toute résolution mathématique est que la **longueur d'une résolution dépend des données du problème et des connaissances mathématiques** disponibles. Considérons la transition :

$$\frac{1}{2}\int_{\pi}^{\theta_1} \frac{p^2}{(1 - \cos\theta)^2}\, d\theta = -\frac{p^2}{4}\left(\frac{1}{3tg^3(\theta_1/2)} + \frac{1}{tg(\theta_1/2)}\right)$$

Elle peut être résolue en une étape si l'expression à intégrer est simple ($R^2\, d\theta$ pour un cercle), s'il s'agit d'une "règle" d'intégration connue, ou si le mathématicien qui l'effectue dispose d'une table d'intégrales ou d'un système de calcul spécialisé. La difficulté intrinsèque des données est alors relative aux moyens disponibles.

Si ces moyens sont trop faibles, le calcul est découpé à volonté. Sa longueur dépend ainsi d'une difficulté relative aux moyens de résolution. Cette hétérogénéïté potentielle engendre ainsi une description très diverse en fonction des données de problèmes a priori proches.

La présentation d'une résolution vient à son tour surimposer des principes analogues à ceux de la résolution effectuée, puisqu'elle cherche à s'adapter aux possibilités de suivi du lecteur. Cette variété de situation renforce l'intérêt des mathématiques.

4.2.3. Synthèse de l'analyse du document

4.2.3.1. Les connaissances mises en œuvre

Ce problème est intéressant car il est proche d'une situation de problème telle que le mathématicien la rencontre. Il montre de ce fait le chemin à parcourir entre une intention intuitive (calculer la surface d'un secteur polaire) et sa formulation en un énoncé mathématique complet. L'analyse du document a en outre montré la **densité et la diversité de connaissances mathématiques requises**. Cela a des conséquences importantes sur la faisabilité pratique d'un système d'assistance.

Les limites de la réalisation d'un tel système sont alors l'incomplétude qui va survenir dans la modélisation du domaine et la quantité des connaissances nécessaires. Une approche d'assistance dans un domaine assez bien formalisé permet de cantonner ces limites à un seuil raisonnable. La phase de résolution du problème peut alors être envisagée.

La nature et la gestion des tâches découlent principalement de connaissances mathématiques du domaine d'étude en laissant une faible part à des principes heuristiques généraux. La compréhension de ce document fait aussi une part importante aux connaissances introduites ailleurs dans le traité. Cet exemple illustre ainsi la **prépondérance de la modélisation du domaine** dans une activité de raisonnement.

La similitude entre données mathématiques et les besoins de concision font que les documents mathématiques utilisent des **procédés linguistiques complexes à interpréter**. Le mathématicien utilise ici ses connaissances pragmatiques pour interpréter une ellipse de langage. Par exemple, tous les exercices d'application du chapitre 11 (p236-238) ont des énoncés du type "Surface entre la courbe..." ; seul le premier précise que le problème est de "Calculer la surface comprise entre la courbe...". L'information sur la nature de la surface est suffisante puisque l'objet du chapitre est commun : le calcul de surfaces.

Plus généralement les mathématiques utilisent couramment des procédés d'abréviation dans les énoncés et leur résolution. Un principe de communication voudrait que le document ne précise que les éléments d'informations nécessaires à la complétion d'un canevas déjà connu du mathématicien-lecteur. L'interprétation des documents mathématiques fait alors appel à des informations implicites essentielles, de nature linguistique et mathématique. Nous qualifions ce procédé de description de *paresseux*, car il ne précise que les informations jugées utiles à la continuation du raisonnement. Nous pouvons donc résumer les procédés de description mathématiques comme étant **paresseux et opportunistes**.

Nous pouvons également ébaucher les préliminaires d'une organisation des connaissances mathématiques :
- le mathématicien exploite plusieurs **représentations mathématiques** au cours de son activité ;
- une représentation mathématique constitue un univers de raisonnement organisé, toutefois il existe des **correspondances** permettant le changement d'interprétation et la coopération entre représentations ;
- les connaissances sont organisées autour de **concepts**. Les informations rattachables à un concept proviennent de sa modélisation individuelle (parabole), de concepts plus généraux (courbe), ou d'un mode d'utilisation par association de plusieurs concepts (surface et courbe) ;
- la **modélisation opératoire** d'un concept regroupe les informations statiques et dynamiques permettant son utilisation dans l'activité mathématique. La terminologie joue un rôle important dans cette utilisation ;

Plus concrètement, un **théorème** comme celui de la formule générale de la surface s'accompagne :
- d'un savoir-faire d'utilisation (quel théorème est utilisable, lequel utiliser, et que faire après) ;
- d'un savoir-faire d'interprétation (quelle est la borne inférieure dans mon problème) ;
- d'un savoir-faire de mise en œuvre (que substituer et comment) ;
- d'un savoir-faire de résolution de sous-problème (comment faire pour trouver $\rho = f(\theta)$);
- d'un schéma de présentation privilégié pour son utilisation (S=...=..., puis calculer l'intégrale).

4.2.3.2. Les aspects pédagogiques

Les aspects pédagogiques sont souvent introduits lorsque le sujet n'est pas un objet mathématique. Cela traduit une ébauche de dialogue personnalisé à travers une relation argumentaire entre pédagogue et élève. Il y a ici :

Profitons, on trouve, Appelons, on sait que, Posons, On aura, On remplacerait.

Ce dialogue a pour objectif l'entraînement à la résolution de problèmes. "On remplacerait" précise comment ce problème s'applique à des données plus instanciées. "Profitons" suivi d'une référence explicite au lecteur sert d'introduction à une nouvelle connaissance. "comment on trouve" indique qu'il s'agit d'une méthode de résolution.

Le travail du lecteur n'est pas à sous-estimer pour autant. La démonstration proposée n'a rien d'une méthode, et le lecteur doit savoir diagnostiquer l'élement essentiel de la démonstration (l'usage de la relation géométrique) pour mémoriser cette méthode. La généralisation éventuelle de cette méthode de la parabole aux coniques est latente et laissée au lecteur s'il y pense.

La généralisation couramment utilisée en mathématiques surpasse donc une description littérale de la résolution. La réutilisation de résolutions effectuées est sinon d'une portée trop limitée. Un processus d'apprentissage symbolique véritable ne peut donc se limiter à la structure apparente de la démonstration.

Les marqueurs de dialogue sont utilisés pour expliciter l'intervention du mathématicien, ici :
- l'introduction d'une nouvelle connaissance ;
- le rappel d'une connaissance ;
- la conduite d'un raisonnement ;
- un choix du mathématicien ;
- le résultat d'une action du mathématicien ;
- l'action à effectuer en fonction d'un objectif.

Ces marqueurs ont ainsi un rôle pédagogique certain, une tournure sans intervention du mathématicien étant toujours possible. La nature de cette intervention détermine de plus leur bon usage : il est erroné par exemple d'intervertir "Posons" et "On aura" dans le changement de variable $t = tg(\theta/2)$.

Plus généralement, la structure apparente d'une démonstration correspond à de nombreux choix, notamment dans l'introduction des données. Il semble alors irréaliste de vouloir modéliser tous ces choix et les structures qui les supportent. Ils se comportent comme une connaissance de mise en forme apparente à partir de la structure profonde.

Il semble alors souhaitable pour un système qui construit une démonstration de respecter la structure profonde de celle-ci. La structure apparente d'une démonstration devrait alors trouver des formes d'expression plus simples. Notre solution consiste à poser d'abord le problème, à mieux expliciter les motivations des actions effectuées tout en utilisant des tournures de raisonnement traditionnelles.

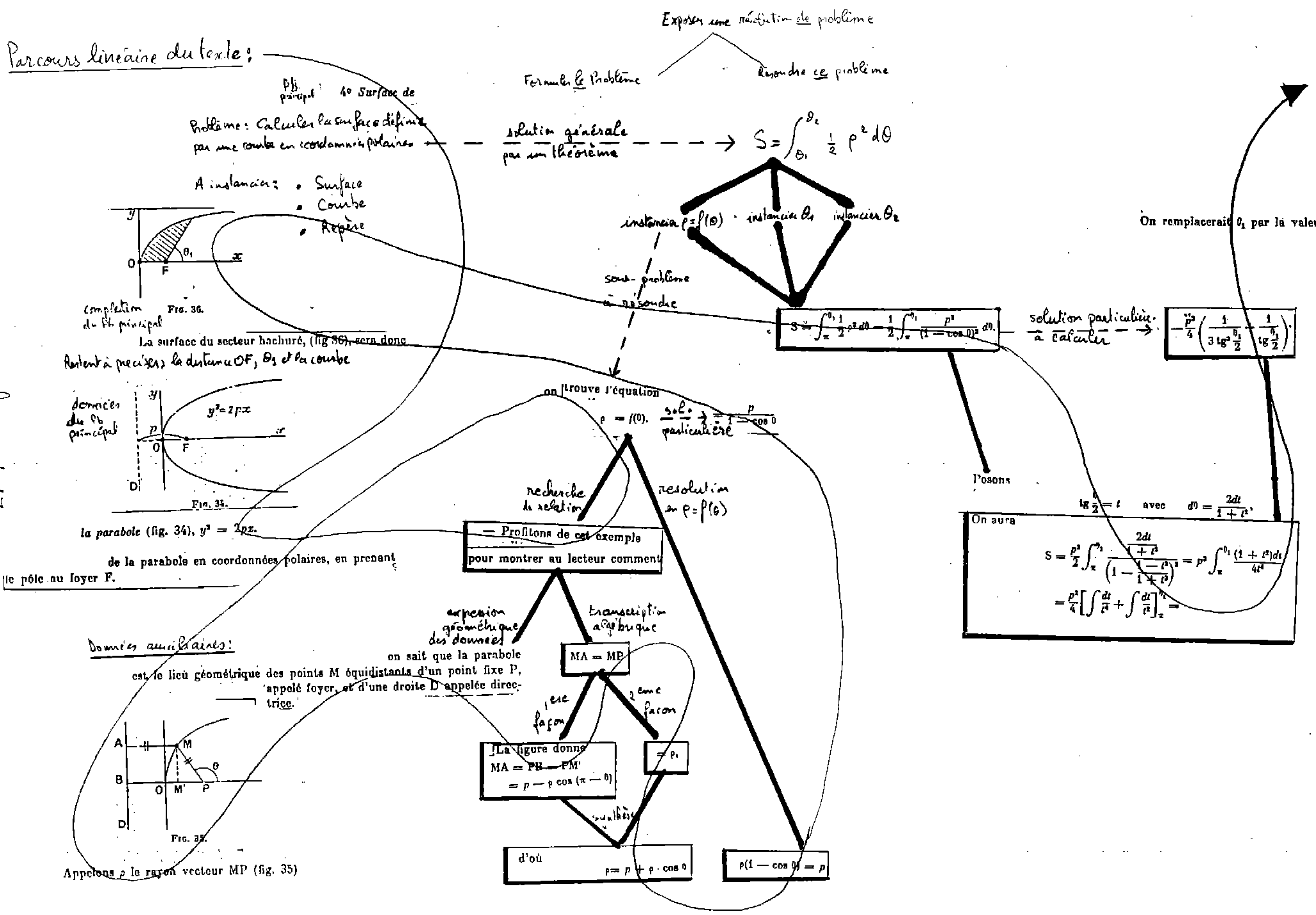

Parcours linéaire du texte :
Exposer une résolution de problème
Formuler le Problème
Résoudre ce problème
Pb. principal
4° Surface de
Problème : Calculer la surface définie par une courbe en coordonnées polaires
solution générale par un théorème
$S = \int_{\theta_1}^{\theta_2} \frac{1}{2} \rho^2 d\theta$
A instancier : • Surface • Courbe • Repère
instancier $\rho = f(\theta)$
instancier θ_1
instancier θ_2
On remplacerait θ_1 par la valeur choisie.
FIG. 36.
complétion du Pb principal
La surface du secteur hachuré, (fig 36), sera donc
Restent à préciser la distance OF, θ_1 et la courbe
sous-problème à résoudre
$S = \int_{\pi}^{\theta_1} \frac{1}{2} \rho^2 d\theta = \frac{1}{2} \int_{\pi}^{\theta_1} \frac{p^2}{(1 - \cos \theta)^2} d\theta.$
solution particulière à calculer
$-\frac{p^2}{4}\left(\frac{1}{3 \operatorname{tg}^3 \frac{\theta_1}{2}} - \frac{1}{\operatorname{tg} \frac{\theta_1}{2}}\right)$
données du Pb principal
$y^2 = 2px$
FIG. 34.
on trouve l'équation
$\rho = f(\theta)$. sol. particulière $\rho = \frac{p}{1 - \cos \theta}$
Posons
$\operatorname{tg} \frac{\theta}{2} = t$ avec $d\theta = \frac{2dt}{1 + t^2}$,
On aura
$S = \frac{p^2}{2} \int_{\pi}^{\theta_1} \frac{\frac{2dt}{1 + t^2}}{\left(1 - \frac{1 - t^2}{1 + t^2}\right)^2} = p^2 \int_{\pi}^{\theta_1} \frac{(1 + t^2) dt}{4t^4}$
$= \frac{p^2}{4} \left[\int \frac{dt}{t^4} + \int \frac{dt}{t^2} \right]_{\pi}^{\theta_1} =$
recherche de relation
résolution en $\rho = f(\theta)$
la parabole (fig. 34), $y^2 = 2px$.
— Profitons de cet exemple pour montrer au lecteur comment
de la parabole en coordonnées polaires, en prenant le pôle au foyer F.
expression géométrique des données
transcription algébrique
Données auxiliaires :
on sait que la parabole est le lieu géométrique des points M équidistants d'un point fixe P, appelé foyer, et d'une droite D appelée directrice.
MA = MP
1ère façon
2ème façon
La figure donne
MA = PB — PM'
$= p - \rho \cos (\pi - \theta)$
$= \rho$
synthèse
FIG. 35.
d'où $\rho = p + \rho \cdot \cos \theta$
$\rho(1 - \cos \theta) = p$
Appelons ρ le rayon vecteur MP (fig. 35)

figure 1.17

4.2.3.3. Une arborescence de tâches

Au ... de distinctions trop arbitraires entre les langages d'expression, nous avons pu discerner deux types de ...sages dans ce document, concernant respectivement la :

- formulation du problème :
  comprenant l'énoncé du problème avec la construction de ses données ;
- résolution du problème :
  comprenant la conduite effective de sa résolution.

Nous proposons une autre présentation de la démonstration qui se rapproche de la structure profonde (figure 1.17 ci-contre). La partie gauche regroupe les données, et la partie haute l'abstraction en un problème général. Le parcours détaillé adopté par J. Quinet dénote un comportement paresseux et opportuniste qui le rend fort complexe.

Les **données du problème** sont organisées plutôt comme des schémas à instancier. L'aspect utile des données est surtout les noms, c'est-à-dire les moyens de désignation. Une fois introduites, les données constituent un contexte dont les informations sont utilisées en parallèle. Ce contexte est détaillé en introduisant des informations implicites de façon opportuniste en fonction des besoins. C'est ce qui explique l'introduction des données du sous-problème en bas à gauche.

L'examen de la figure montre le glissement progressif géométrique-algébrique, qui se traduit par la diminution du flux d'utilisation des données du problème. La modélisation de ce glissement est importante car elle se retrouve dès que coexistent plusieurs représentations. Elle joue donc un rôle essentiel lors des applications physiques des mathématiques.

Lorsque nous considérons cette démonstration pour son raisonnement mathématique, nous observons deux grands enchaînements de déductions :

- le calcul de la surface S = ...
- l'exploitation de la relation de la parabole.

Les tâches abordées pour le parcours sont dans l'ordre :

- l'introduction des données du problème ;
- la construction géométrique de la relation particulière à la parabole ;
- le raisonnement géométrique donnant la transcription algébrique de cette construction ;
- la résolution de cette transcription algébrique ;
- l'utilisation de l'équation polaire obtenue dans la formule de la surface ;
- la résolution de la formule de la surface.

Les caractéristiques spécifiques notables lors de l'exécution de ces tâches sont alors :

- l'instanciation, lorsqu'il existe un cadre plus général ;
- l'extension du contexte ;
- la décomposition et synthèse de tâches ;
- la gestion des continuations en fin de tâche ;
- l'ordre de parcours des tâches pour réaliser une tâche plus générale.

Ces caractéristiques permettent de déterminer l'essentiel du raisonnement du mathématicien. Elles n'ont guère été étudiées à notre connaissance sur des cas réalistes. Nous nous contentons de constater ici que les **structures doivent être très souples dans leur décomposition et leurs possibilités de parcours**. C'est une des caractéristiques essentielles recherchée par les preuves à granularité variable. Notons toutefois qu'une preuve s'intéresse prioritairement aux productions et actions effectuées. Nous serons donc amenés à étudier, dans le chapitre suivant concernant les preuves à granularité variable, la preuve sous-jacente à ce problème.

4.3. ANALYSE DIRECTE DE TEXTES D'UN OUVRAGE MATHEMATIQUE

4.3.1. L'interprétation des écritures

Nos analyses se focalisent maintenant sur le texte des sections 4 et 5 du chapitre 8 du traité d'algèbre de Roger Godement [Godement 66, pp. 144-147]. Ce texte présente la formule du binôme. Les coefficients du binôme sont introduits par l'énoncé suivant :

$$\binom{n}{p} = \frac{n\,(n-1)\ldots(n-p+1)}{1.2\ldots p} = \frac{n!}{p!\,(n-p)!} \quad (0 \le p \le n).$$

La notation introduite "$\binom{n}{p}$" est donc égale à deux expressions distinctes, désignées respectivement par ① et ②. Cela signifie notamment qu'il peut y avoir deux définitions algébriques des coefficients binomiaux en fonction de deux modes de représentation algébrique :

- une représentation intuitive ①, où le mathématicien raisonne sur des termes dont il s'agit de décrire le contenu et les liens à l'aide d'une description intuitive ;
- une représentation formelle ②, où les mathématiciens ont cherché à formaliser les descriptions intuitives par des notations et des lois de manipulation.

Dans la **représentation intuitive** ① l'entité de référence est le terme, qui est construit à l'aide d'un petit groupe d'objets mathématiques atomiques souvent regroupés par des parenthèses. Dans une représentation intuitive, **la signification d'une écriture doit être extraite de façon non triviale du sens de ses constituants**. Le dénominateur se lit alors littéralement "1 multiplié par 2 et coetera jusqu'à p". Il s'agit donc d'un produit dont le premier terme est "1", le second "2", et le dernier "p". Les capacités de généralisation du mathématicien sont alors utilisées pour inférer qu'il s'agit des p premiers entiers non nuls.

Le numérateur est un produit dont le premier terme est "n" le second "n-1" et le dernier "n-p+1". Il peut donner lieu à trois interprétations équivalentes :

- Si le mathématicien interprète les termes lus comme d'autres termes mathématiquement équivalents : "n-0", "n-1" et "n-(p-1)", il s'agit du produit de p termes de la forme "n-r" tels que "$(0 \le r \le p-1)$".
  L'interprétation de la notation est une fonction de l'espace des termes dans l'espace des termes mathématiquement équivalents.
- Si le mathématicien interprète les termes par les entiers qu'ils représentent, il s'agit du produit de p entiers en "descendant à partir de n".
  L'interprétation de la notation est une fonction de l'espace des termes vers l'espace des entiers (c.-à-d. vers le Modèle mathématique associé).
- Si le mathématicien interprète les termes comme une séquence descendante, il s'agit du produit des entiers "en descendant à partir de" n, auquel on a enlevé les n-p derniers entiers.
  L'interprétation de la notation est une fonction de l'espace des termes vers l'espace des opérations intuitives sur des séquences d'entiers.

Nous adoptons cette troisième interprétation dans la suite.

Dans la **représentation formelle** ②, les coefficients du binôme sont écrits à l'aide de la factorielle. Si la factorielle dispose d'une interprétation directe, l'interprétation de la notation est simple. La signification de cette écriture est alors principalement dépendante de son usage dans les manipulations. Cet usage faisant intervenir la factorielle, sa signification est construite à partir de celle de la factorielle. Il n'y a plus qu'à donner une définition récursive de la factorielle pour rester au sein d'une représentation algébrique formelle :

$$n! = n\,(n-1)! \quad \text{si } n > 1,$$

$$0! = 1! = 1.$$

Une représentation formelle des notations cherche à formaliser les règles de manipulation des notations, de façon à tendre vers un système formel capable de les contrôler. Par contre, une représentation intuitive cherche à profiter des capacités d'interprétation des mathématiciens pour améliorer l'expressivité des manipulations. Les représentations intuitives sont préférées pour cela dans les traités mathématiques.

La **correspondance** entre la représentation intuitive ① et la représentation formelle ② des coefficients du binôme peut aussi être établie et exploitée. Il suffit d'utiliser la représentation intuitive des factorielles, à la façon de Roger Godement qui écrit page 97 :

où, pour tout entier naturel n, on pose

$$n! = 1.2.3...n \text{ si } n \neq 0, \quad \text{et} \quad n! = 1 \text{ si } n = 0.$$

Cette correspondance requiert une analyse de la part du mathématicien, en comparant les deux définitions des coefficients du binôme. Une fois le dénominateur de ① interprété comme p!, le numérateur peut être identifié à l'interprétation intuitive de n!/(n-p)! pris globalement.

4.3.2. La maîtrise des écritures

L'interprétation d'une notation, et plus généralement d'une écriture mathématique ne se résume donc pas à reconstituer littéralement cette écriture. Elle s'accompagne de plus d'un travail d'analyse et de modélisation de l'écriture à des fins opératoires. L'interprétation d'une notation s'accompagne donc d'une **maîtrise de l'écriture permettant de reconstituer l'activité** codée par l'écriture lors de la lecture.

Une définition est ainsi mémorisée dans un mode opératoire permettant de l'appliquer à des données spécifiques. Avec la représentation intuitive ①, le mathématicien reconstitue le numérateur puis le dénominateur. Il choisit par exemple pour reconstituer le numérateur de :

- partir du premier terme du coefficient du binôme ;
- d'aboutir au terme construit en prenant la partie supérieure moins la partie inférieure plus un.

La représentation formelle ② se prête plus à une mémorisation par cœur de la définition comme : (le coefficient du binôme avec n en haut et p en bas égale) "factorielle n sur factorielle p sur factorielle n-p". Son application se fait alors par récitation de la séquence et substitution par les données actuelles. Une telle méthode est inadaptée pour la représentation intuitive où les termes ont un rôle différent.

Tout résultat mathématique se traduit ainsi en une dynamique d'utilisation à l'aide de contraintes de dépendances (ne serait-ce que pour son évocation). Cette maîtrise de l'écriture s'accompagne aussi d'une **pragmatique d'utilisation** regroupant des caractéristiques non précisées dans les définitions.

La notation "..." illustre plus que tout autre comment, en mathématiques, la forme de la présentation traduit l'intention du mathématicien. Elle donne d'ailleurs lieu à un développement sur son interprétation dans l'ouvrage [Graham 89]. Par exemple en prenant "$p=2^r$" dans l'exemple trivial "1.2...p", on obtient la suite des "2^r" premiers entiers non nuls. Si le mathématicien écrit au contraire "$2^0.2^1...2^r$" l'interprétation est toute autre... Le mathématicien a rendu explicite la forme sur laquelle porte la généralisation à effectuer. Ce présupposé d'interprétation est l'une des règles associée à la notation "..." .

Cette forme de la présentation utilise aussi la position respective des termes comme indice d'interprétation. Par exemple, dans la même page 145 de Roger Godement, le dernier terme du numérateur "(n - p + 1)" est écrit dans un ordre différent dans "x^{n-1-p}" (à ±1 près). Dans ce dernier cas, l'intention du mathématicien est claire : il s'agit de "n-1" moins p.

Dans ces deux exemples, lorsqu'il cherche à clarifier son intention, le mathématicien s'éloigne donc d'une "forme normale par défaut" résultant des usages d'écriture afin de marquer son intention par l'écriture. A l'extrême, il recopie "n-p" comme "(n+1) - (p+1)" afin de clarifier son intention d'interprétation de la forme simplifiée "n-p". De même, Roger Godement aurait pu noter "n-0" au lieu de "n" s'il avait voulu marquer une expression de la forme "n-r" dans l'interprétation des coefficients du binôme. Il aurait pu aussi l'indiquer plus subtilement en notant "n (n-1)...(n-(p-1))", c'est-à-dire en l'indiquant uniquement sur la forme considérée comme "la plus générale" de la séquence de termes.

La maîtrise d'une écriture consiste à reconstituer **l'utilisation, la provenance et le devenir** de cette écriture dans la (les) représentation(s) de raisonnement du mathématicien. Dans cette optique d'interprétation, la non-simplification ou l'absence de conformisme aux usages du reste du texte traduisent

une intention. Les écritures comportent alors des signaux que nous nous attachons par la suite à décoder en fonction des connaissances et des traitements d'un mathématicien.

Nous abordons d'abord l'usage macroscopique du langage pour permettre une description panoramique de l'activité mathématique. Nous décrivons ainsi les représentations utilisables dans une argumentation, nous précisons les connaissances et capacités de traitement utilisées chez le lecteur, puis nous précisons le rôle d'une démonstration.

La section suivante s'attache plus à l'analyse et à la production d'étapes de démonstration. Elle détaille les mécanismes de production (et d'interprétation) depuis le problème jusqu'à la gestion du niveau basique - et même microscopique ! Cela aboutit ultérieurement à la description d'un modèle de preuve à granularité variable chargé de représenter les opérations de construction de preuve tels que les effectue un mathématicien.

4.3.3. Diversité des représentations utilisables dans une argumentation

Les représentations utilisées par un mathématicien sont essentielles pour connaître les mécanismes de guidage et de découverte des démonstrations. La diversité des représentations disponibles est bien connue lors d'un usage "appliqué" des mathématiques (par ex. les problèmes de poulies...). Elle est d'une richesse similaire pour la géométrie. Cette diversité est moins reconnue à l'intérieur des autres branches des mathématiques. Aussi, pour l'illustrer dans ce contexte, nous proposons d'examiner la demi-page 146 du traité de Roger Godement. Elle est consacrée uniquement aux diverses justifications d'une relation simple!

L'examen du tableau précédent suggère la formule

$$\binom{n}{r} = \binom{n}{n-r} ;$$

la vérification de celle-ci est immédiate puisque

$$\binom{n}{r} = \frac{n!}{r!\,(n-r)!}, \qquad \binom{n}{n-r} = \frac{n!}{(n-r)!\,r!} ;$$

mais la véritable raison de la formule en question réside dans le fait que l'expression

$$\sum_{p=0}^{p=n} \binom{n}{p} x^{n-p}\, y^{p}$$

ne doit pas changer — vu sa signification — si l'on permute x et y ; il est donc naturel que les coefficients dans cette expression des «monômes» $x^{n-r}\, y^{r}$, $x^{r}\, y^{n-r}$ soient égaux, puisque ces monômes se déduisent l'un à l'autre par l'échange de x et y.

On peut aussi observer que, X désignant un ensemble à n éléments, $\binom{n}{r}$ est le nombre de parties à r éléments de X ; en associant à une telle partie Y son complémentaire X-Y, on obtient une bijection de l'ensemble des parties à r éléments de X sur l'ensemble des parties à n-r éléments de X; d'où la relation $\binom{n}{r} = \binom{n}{n-r}$.

Nous organisons cette analyse en :

- <u>représentation</u> :
  la représentation qu'utilise la justification de la relation ;
- <u>genèse</u> :
  l'origine de la production de la relation à partir de la représentation ;
- <u>moyens</u> :
  les moyens utilisés pour produire cette justification ;
- <u>vérification</u> :
  le type de justification produite ;

L'examen du tableau précédent suggère la formule

$$\binom{n}{r} = \binom{n}{n-r} ;$$

- représentation : le tableau référencé est le triangle de Pascal. Il s'agit donc d'une organisation des extensions, c'est-à-dire des nombres qui interprètent ces coefficients.
- genèse : dûe à un procédé de calcul récursif des extensions.
- moyens : observation des régularités de l'organisation produite.
- vérification : empirique, réfutation possible via un contre exemple expérimental.

la vérification de celle-ci est immédiate puisque

$$\binom{n}{r} = \frac{n!}{r!\,(n-r)!}, \qquad \binom{n}{n-r} = \frac{n!}{(n-r)!\,r!} ;$$

- représentation : algébrique formelle, il n'y a pas besoin de Modèle d'interprétation.
- genèse : procédé de manipulation de formules utilisant la définition formelle.
- moyens : se ramener à une identité évidente.
- vérification : règles syntaxiques de calcul (preuve mécanique).

mais la véritable raison de la formule en question réside dans le fait que l'expression

$$\sum_{p=0}^{p=n} \binom{n}{p} x^{n-p}\, y^{p}$$

ne doit pas changer — vu sa signification — si l'on permute x et y ; il est donc naturel que les coefficients dans cette expression des «monômes» $x^{n-r}\, y^{r}$, $x^{r}\, y^{n-r}$ soient égaux, puisque ces monômes se déduisent l'un à l'autre par l'échange de x et y.

- représentation : algébrique intuitive.
- genèse : relation dans laquelle le coefficient du binôme intervient : elle doit rester invariante vu son interprétation intuitive.
- moyens : effectuer des recoupements entre plusieurs niveaux d'interprétation.
- vérification : règles de cohérence entre interprétation (justification globale ou indirecte).

On peut aussi observer que, X désignant un ensemble à n éléments, $\binom{n}{r}$ est le nombre de parties à r éléments de X ; en associant à une telle partie Y son complémentaire X-Y, on obtient une bijection de l'ensemble des parties à r éléments de X sur l'ensemble des parties à n-r éléments de X; d'où la relation $\binom{n}{r} = \binom{n}{n-r}$.

- représentation : ensembliste "naive".
- genèse : multiples points de vue dans l'interprétation de relations dans un Modèle.
- moyens : argumentation à l'aide de fonctions sur les ensembles.
- vérification : règles de correspondance ensemblistes.

$$(x+y)^{n} = \underbrace{(x+y)\,(x+y)\ldots(x+y)}_{n \text{ facteurs}},$$

Nous y ajoutons une argumentation issue de [Graham 89, p163]. Elle explique que le nombre de termes avec r facteurs de x et n-r facteurs de y est le coefficient de $x^{r}\, y^{n-r}$ après combinaison des termes identiques. Or c'est exactement le sens de $\binom{n}{r}$. La symétrie est alors invoquée pour justifier la relation à montrer.

- représentation : combinaison des deux représentations précédentes : algébrique intuitive pour le développement de la puissance, et ensembliste "naive" pour l'interprétation du nombre de possibilités de choisir des parties à r éléments dans un ensemble à n éléments.
- genèse : interprétation intuitive d'une relation dans laquelle le coefficient du binôme intervient, en allant un cran plus loin que dans la formule avec le Σ.

- moyens : combiner de façon complexe plusieurs niveaux de représentation. L'argument de symétrie est toutefois employé de façon plus directe qu'avec le Σ.
- vérification : lorsque l'arsenal des mathématiques est en jeu, elle ne peut se faire que relativement aux représentations intuitives et formelles utilisées.

Ces représentations servent pour exprimer des méthodes de calcul. Par exemple, Roger Godement écrit p. 147 pour la formule du binôme généralisée à un produit $\prod(x_i + y_i)$:

> pour multiplier des sommes les unes par les autres, on choisit, de toutes les façons possibles, un terme dans chacune de ces sommes, on effectue le produit des termes ainsi choisis, et on additionne les résultats ainsi obtenus pour tous les choix possibles.

Un objectif de la linguistique des mathématiques consiste à programmer des tâches et des méthodes à ce niveau épimathématique. Plus encore que la production d'une maquette réaliste, l'intérêt de cet objectif serait d'étudier les phénomènes afin d'en tirer des conséquences pour la programmation informatique. Il serait souhaitable de comprendre ainsi la méthode de production du triangle de Pascal [Godement 66, p. 146] :

> la méthode pour calculer le p° terme de la n° ligne, consiste à additionner le p° et le (p-1)° termes de la ligne précédente.

La palette d'utilisation des représentations est donc extrêment vaste en mathématiques. Vouloir ne garder que des représentations formelles revient à s'amputer volontairement d'une grande part des mathématiques. Prétendre automatiser les changements d'interprétation permettant ces argumentations est irréfléchi. Chercher à **modéliser certaines représentations intuitives** et certaines règles de changement d'interprétation est plus réaliste mais encore inexploré.

Les Sciences Cognitives ont ainsi de nombreux champs d'exploration en mathématiques. Par exemple un ouvrage comme [Janvier 87] s'attache spécifiquement à une première approche des représentations en mathématiques, mais à des fins essentiellement didactiques. Une idée intéressante est celle de représentation opaque ou transparente. Une *représentation opaque* masque son interprétation d'origine et permet de travailler uniquement sur cette représentation. Par contraste une *représentation transparente* travaille sur un Modèle via un langage qui sert uniquement d'outil de manipulation.

Pour ce qui concerne les outils de travail du mathématicien, les représentations employées par le mathématicien ont une double incidence :

- une représentation formelle est indispensable pour pouvoir effectuer des vérifications, or elle n'est pas souvent la plus intéressante à manipuler ;
- dans tous les traitements humains, les représentations intuitives servent à la découverte et au guidage du raisonnement lors de la production de preuves.

Les facilités permises dans une représentation intuitive sont souvent primordiales pour **emporter la conviction.** Un des points forts du travail de Laurent Chaudron [Chaudron 89] est d'ailleurs la recherche des éléments pertinents pour emporter la conviction d'un autre mathématicien. Ces éléments pertinents interviennent aussi dans la mémorisation de la démonstration, d'après la propriété "pars pro toto" qui permet de retrouver le tout à partir de la donnée d'une de ses parties.

En conséquence, il est vain de prétendre automatiser l'activité d'un mathématicien hors des domaines où des procédures de calcul réalistes ont pu être mises en évidence (par le travail des mathématiciens!). Il est aussi souvent biaisé de prétendre automatiser un domaine à partir d'un prototype trop réduit. C'est pourquoi, nous prônons une **approche d'assistance au mathématicien associée à une étude expérimentale des mathématiques.**

4.3.4. Compétences et connaissances du lecteur

Dans cette analyse, le texte des sections 4 et 5 du chapitre 8 du traité d'algèbre de Roger Godement [Godement 66, pp. 144-147] est découpé en blocs selon la nature de son contenu. Nous présentons ici les premiers blocs, chaque bloc est scindé en cinq parties :

- bloc mathématique n°i,
  avec en note la nature de la liaison avec le bloc précédent (au cours de cette analyse, le mot "précédent" désigne le bloc précédent) ;
- panorama :
  synthétise les caractéristiques du contenu du texte ;
- lecteur :
  illustre la diversité et la complexité de la modélisation du lecteur
- analyse :
  certains traits de l'activité mathématique
- résultats partiels :
  une première mise en valeur des résultats de l'analyse du bloc considéré.

BLOC MATHEMATIQUE N°1 (nouvelle section)

4. Formule du binôme
Les « identités remarquables » qu'on démontre dans l'enseignement secondaire lorsqu'il s'agit de nombres réels, sont pour la plupart encore valables dans tout anneau (en supposant parfois que les éléments x, y, ... figurant dans ces identités commutent deux à deux).

a) PANORAMA
Niveau de discours : périmathématique
Représentation exploitée : intuitive
Objectif principal : donner les moyens de manipulation d'expressions mathématiques dans des anneaux quelconques.
Problème traité : étendre les identités familières aux anneaux

b) LECTEUR
Capacités de traitement : compréhension de phrases en Langage Naturel.
Connaissances utilisées : manipulations d'expressions mathématiques sur les ensembles de nombres usuels (N, Z, Q, R, C).
Effet attendu : étendre la validité des identités familières au cadre général des manipulations dans un anneau quelconque, sous réserve éventuelle que les éléments manipulés commutent deux à deux.

c) ANALYSE
Procédés langagiers : l'auteur reste très vague dans sa description.
Organisation des connaissances : il s'agit d'un procédé didactique et naturel de généralisation de connaissances par simple extension de leur domaine d'application, moyennant une condition de validité supplémentaire dans certains cas.
raisonnement mathématique : RAS.

d) RESULTATS PARTIELS
La **modélisation des connaissances** du mathématicien permet ce type d'extension par réutilisation d'un ilot de connaissances dans des situations plus générales, moyennant une procédure d'adaptation servant d'interface à cette extension. Pour l'instant, cette extension n'est que potentielle, et doit être validée cas par cas.

Il s'agit ici de l'introduction d'une **étape d'apprentissage**, dont l'analyse et la modélisation sont un enjeu essentiel des Sciences Cognitives.

La présentation d'un discours mathématique ressemble par moments à celle d'un **discours en Langage Naturel** portant sur un tout autre sujet, néanmoins familier au lecteur destinataire.

BLOC MATHEMATIQUE N°2 (même ligne)
Par exemple, soient x, y deux éléments d'un anneau K, et calculons

$$(x + y)^2 = (x + y)(x + y) = x^2 + xy + yx + y^2 ;$$

on voit que, si x et y commutent, on retrouve l'identité $(x + y)^2 = x^2 + 2xy + y^2$.
On a alors

$$(x + y)^3 = (x^2 + 2xy + y^2)(x + y) = x^3 + 3x^2y + 3xy^2 + y^3$$

comme le montre un calcul trivial.

a) PANORAMA
Niveau de discours : mathématique, à exploitation périmathématique
Représentation exploitée : algébrique
Objectif principal : illustrer l'objectif précédent sur deux identités remarquables.
Problème traité : retrouver des identités familières et montrer le rôle de la condition sur x et y.

b) LECTEUR
Capacités de traitement : capacités de perception, conduite de calculs, suivi de calculs et compréhension d'énoncés en Langage Mathématique.
Connaissances utilisées : définition d'une puissance, manipulation de sommes et produits
Effet attendu : valider l'affirmation précédente.

c) ANALYSE
Procédés langagiers : l'auteur indique la nature de la transition à ce bloc par un marqueur langagier "Par exemple". Il s'attache à préciser la nature des objets mathématiques manipulés (x,y et K). Le fait que K soit inutilisé dans la suite du bloc montre que la nécessité de son introduction procéde d'une exigence d'argumentation mathématique. L'auteur distingue les calculs (une ligne particulière introduite par "calculons"), de leur résultat (une identité) qui est inséré dans le texte comme les autres descriptions. L'intention périmathématique de ces calculs fait que l'auteur commente la preuve et la validité de ces déductions ("on retrouve", "comme le montre un calcul trivial"). Il gagne ainsi en expressivité ce qu'il perd en concision.
Organisation des connaissances : importance de la prise en compte des cas particuliers. Il y a ici introduction de deux règles de développement de puissances. Elles sont accompagnées de leur contexte d'application (explicité au niveau langagier, d'où l'importance de "K") et des calculs qui leur servent de validation.
raisonnement mathématique : Il fait ici appel à des calculs dont les étapes sont susceptibles d'être justifiées et dont le résultat est établi en règle. Il met l'accent sur une propriété particulière qui permet de justifier une étape indispensable. L'usage langagier des mathématiques voulant que la dernière ligne corresponde au résultat, le lecteur établit ici des inférences implicites pour trouver que "$x^2 + xy + yx + y^2$" n'est qu'un résultat partiel peu intéressant alors que "$x^2 + 2xy + y^2$ " est un résultat notable. La justification "on voit que, si x et y commutent" est la synthèse des inférences implicites d'identification des sous-formules "xy + yx" et "2xy".

d) RESULTATS PARTIELS
Il existe des contraintes langagières provenant de **lois de présentation** de l'argumentation mathématique. Ces procédés langagiers influent sur la conduite des déductions. La règle est de présenter à gauche la situation nouvelle (ici la puissance d'une somme) afin de l'exprimer en fonction d'éléments isolément plus simples.

Tout calcul ou raisonnement, aussi élémentaire soit-il, requiert des **capacités de perception.** Il s'agit notamment de :
- l'**isolation** (point de vue sélectif, permettant de n'envisager qu'un aspect des choses) ;
- l'**évocation** (reconnaissance d'une forme proche déjà vue) ;
- l'**analyse de traits** (description d'une forme selon des critères jugés pertinents) ;
- la **généralisation** d'une forme (pour la classifier ou la situer parmi d'autres formes) ;
- la **comparaison** de deux formes (production de similitudes et différences) ;
- l'**identifiabilité** de deux formes (génération d'hypothèses qui les rendraient identiques) ;

Ces points peuvent être mis en œuvre en plusieurs niveaux imbriqués. Nous résumons usuellement cette vue personnelle des capacités de perception en parlant de capacités de reconnaissance et d'association. Pour l'illustrer dans le développement de "$(x + y)^3$", "$(x^2 + 2xy + y^2)$" est identifiée comme une somme de facteurs qu'il a fallu parenthéser. L'origine de cette expression est alors elle aussi identifiée comme le

développement de "$(x + y)^2$", soit par calcul, soit par reconnaissance d'un résultat précédemment exposé, soit encore par les deux processus simultanément.

Ces capacités de perception sont ainsi constamment sollicitées. De plus, la nature élémentaire des capacités de perception est justifiée par le "on voit que" pour le cas de "xy + yx" et "2xy". C'est pourquoi, les capacités de perception sont implicites et ne sont éventuellement citées que de façon allusive.

Les capacités de **conduite de calculs** consistent à appliquer les règles de définition des opérations élémentaires (addition, multiplication, puissance, et autres lorsqu'elles sont définies...), et à manipuler et réassembler les sous-expressions que cela comporte. L'opération de base : appliquer une règle, est parfois qualifiée de calcul. La conduite de calculs consiste alors à savoir effectuer et enchainer ces opérations de base. Lors de cette conduite de calculs, certaines règles usuelles comme "xy = yx" peuvent être neutralisées sans remettre en cause le processus général. Ces capacités de conduite de calcul font donc appel à la mise en œuvre de procédures et de raisonnements paramétrés par les règles disponibles.

Les capacites de **suivi de calculs** regroupent des capacités de perception et de maîtrise du langage. La maîtrise du langage consiste notamment à savoir découper et analyser les expressions, par exemple à identifier que "$(x + y)\ (x + y)$" est le produit de la somme de x et de y par une somme identique. Les capacités de conduite de calculs sont alors utilisées pour obtenir l'égalité avec les formules qui l'encadrent.

Enfin, ces capacités suggèrent des capacités d'**écriture de calculs**, afin de produire des expressions bien formées, en adéquation avec leur sens, et respectant les intentions de l'auteur quant à la qualité du suivi des calculs. Nous regroupons ces trois dernières capacités sous le vocable : **capacités de calcul**.

BLOC MATHEMATIQUE N°3 (nouveau paragraphe)

Plus généralement :

Théorème 3. Soient K un anneau, x et y deux éléments de K, et supposons que x et y commutent. On a alors, pour tout entier $n \geq 1$, la relation

$$(x + y)^n = \sum_{p=0}^{p=n} \binom{n}{p} x^{n-p}\, y^p$$

où l'on pose

$$\binom{n}{p} = \frac{n\,(n - 1)\ldots(n - p + 1)}{1.2\ldots p} = \frac{n!}{p!\,(n - p)!} \quad (0 \leq p \leq n).$$

a) PANORAMA

Niveau de discours : mathématique
Représentation exploitée : algébrique
Objectif principal : introduire un théorème utile dans un anneau quelconque
Problème traité : développement d'une puissance

b) LECTEUR

Capacités de traitement : analyse tolérante aux nouveautés, compréhension de notations semi-formelles, construction d'un contexte hypothétique.
Connaissances utilisées : définition et interprétation intuitive du Σ, manipulation de termes indicés, sens intuitif des "..." et définition de la factorielle.
Effet attendu : ajout de ce théorème et de la définition qui l'accompagne avec les connaissances déjà présentes dans la rubrique "anneau quelconque".

c) ANALYSE

Procédés langagiers : Le théorème comprend la description du contexte, la description de la relation, et une définition annexe. Il est annoncé par deux marqueurs. L'un est lié à l'argumentation périmathématique ("Plus généralement"), l'autre au procédé langagier de désignation ("Théorème 3"). L'introduction des objets x,y et K diffère de celle du bloc précédent par l'introduction préliminaire de K. Elle s'accompagne d'une hypothèse sur x et y explicitement mise en évidence. Par contre, l'entier n est introduit séparément alors qu'il aurait pu l'être comme x et y. L'usage mathématique veut qu'un objet manipulé soit catégorisé (entier, relation, etc), et qu'il ne soit introduit que lorsque le besoin s'en fait sentir (d'où la définition finale de $\binom{n}{p}$ postérieure à son introduction). Cette définition est accompagnée de ses conditions de validité ("$0 \leq p \leq n$") introduites ici a posteriori.

Organisation des connaissances : Un théorème est rédigé afin de faciliter son utilisation. Il présente dans son contexte la situation dans laquelle le mathématicien se trouve lors de son utilisation, et distingue par une hypothèse la propriété à vérifier pour la validité de l'application (x et y commutent). Un théorème peut alors être étiqueté par son rôle principal dans l'action : développer la puissance d'une somme.
raisonnement mathématique : de nombreuses inférences implicites dans l'interprétation du Σ. Outre l'interprétation de la notation Σ (que faire de "p=n" ?), il y a l'identification d'une notation non encore définie, et le choix des exposants "$x^{n-p}\ y^{p}$" au lieu de "$x^{p}\ y^{n-p}$", afin de retrouver l'ordre des polynômes correspondant au développement du Σ. De même, dans le développement de $\binom{n}{p}$: il faut savoir que n et p sont des entiers, identifier "1.2...p" à "p!" (via le produit des p premiers entiers qui est une définition de la factorielle de p), et surtout reconnaître que "$n\,(n-1)\ldots(n-p+1) = \frac{n!}{(n-p)!}$", ce qui est beaucoup plus complexe...

d) RESULTATS PARTIELS
Un théorème a une **structure mathématique interne** explicable en terme de son utilisation. Une modélisation des connaissances doit permettre d'en extraire une procédure d'utilisation : indexer un théorème par l'opération à effectuer ou par son contexte d'application et vérifier que ses conditions de validité sont respectées.

Un théorème a pour objet une **relation principale** aussi générale que possible que le mathématicien est chargé d'analyser. Cette relation est définie si toutes les sous-expressions qui la composent sont définies. Le rôle du théorème est d'introduire les objets atomiques avec leurs relations, et celui du mathématicien est de vérifier que les autres sous-expressions sont définies. Il faut ici vérifier que n est un entier ; que (K,+) est un groupe commutatif et que "$n \geq 0$" pour que le Σ soit défini ; que "n-p" et "p" et surtout que $\binom{n}{p}$ sont des entiers. Une remarque au théorème vient ainsi justifier ce résultat implicite nécessaire mais non trivial. Notons enfin que x^0 est supposé défini et égal à l'élément unité de K : "I", supposé lui-même exister d'après la définition choisie pour un anneau. Une fois cette relation définie, le mathématicien peut vérifier sa validité moyennant l'examen de sa démonstration.

4.3.5. Rôles possibles d'une démonstration

Cette analyse de quelques lignes d'un texte mathématique a montré que les sujets de recherche et les connaissances à fournir sont nombreux et variés. Ce ne serait pas trahir les mathématiques que de déclarer qu'elles sont au service des démonstrations.

Le **rôle d'une démonstration** a été longuement discuté en didactique des mathématiques (par ex. [Balacheff 87]). Les fonctions utilitaires du langage mathématique permettent d'en illustrer les composantes :

- fonction mathématique descriptive :
  présenter les arguments employés pour aboutir au résultat mathématique;
- fonction mathématique évaluative :
  décider de la validité d'une preuve ;
- fonction de communication explicative :
  convaincre l'interlocuteur de la validité du résultat ;
- fonction de communication épistémique :
  servir de référence communautaire, pour l'importance du résultat, pour le fil conducteur ou pour les moyens techniques employés dans la démonstration ;

Une démonstration a des objectifs tout à fait différents et le point fort de la didactique est de chercher à désacraliser les démonstrations. Il y a des démonstrations pour acquérir le besoin de prouver, pour apprendre comment justifier, pour apprendre la dialectique, pour apprendre les procédés heuristiques, pour comprendre les arguments principaux, pour prouver la rigueur de la démarche, pour convaincre un autre expert, pour avancer un argument convainquant qui servira ultérieurement dans la mémorisation[1], etc.

[1] C'est l'une des usages possibles du raisonnement approché tel qu'il est étudié par L. Chaudron [Chaudron 89].

Ainsi, **une démonstration varie selon les buts poursuivis par le mathématicien** qui l'écrit. Notre opinion est que les démonstrations couramment proposées sont trop souvent influencées par les besoins et les usages de rédaction d'une école mathématique dominante.

Le choix du mathématicien lors de la présentation de sa solution se réduit globalement à l'alternative :

- **privilégier l'intuition** et avancer une argumentation allant à l'essentiel, quitte à ce que le destinataire n'ait que des présomptions sur la validité effective du résultat.
- **privilégier la validité** d'une preuve, en acceptant les contraintes garantes de cette rigueur langagière.

Une démonstration qui privilégie l'intuition aide aussi à trouver une démonstration privilégiant la validité. En effet, elle peut le faire de deux façons :

- en abrégeant le détail des opérations : il est alors possible de détailler la preuve.
- en utilisant une représentation plus intuitive ne disposant pas d'opérations ayant une validité reconnue : il existe toujours une subtile relation d'adéquation qui permet de guider la preuve dans une représentation plus "formelle" (par ex. un guidage utilisant des similarités intermédiaires).

La démonstration par récurrence de la formule du binôme présentée dans le livre de Roger Godement privilégie l'intuition. Nous présentons ultérieurement une autre démonstration de ce résultat privilégiant la validité. Conformément au reste de son traité, Roger Godement met en avant sa volonté de clarté didactique, sans sacrifier bien entendu à la rigueur des résultats présentés. Cette rigueur se traduit principalement dans l'exposé des énoncés et de leur organisation. Par contre, les démonstrations de ce traité sont essentiellement des indications sur les éléments pertinents, à l'intention du lecteur. Nous les avons qualifiées précédemment d'argumentation épimathématique.

Cette démonstration utilise essentiellement la représentation intuitive de la sommation Σ. Par opposition à une représentation formelle constituée de règles de manipulation du Σ, une représentation intuitive considère que le Σ n'est qu'un procédé de notation pratique pour représenter des termes similaires. L'essentiel de la démonstration est alors :

> (...) Or, en multipliant par x+y la relation précédente, il vient
>
> $$(x+y)^n = \sum_{p=0}^{p=n-1} \binom{n-1}{p} x^{n-p} y^p + \sum_{p=0}^{p=n-1} \binom{n-1}{p} x^{n-1-p} y^{p+1} ;$$
>
> si r est un entier tel que 0<r<n, il y a au second membre de la relation précédente deux termes contenant le monôme
>
> $$x^{n-r} y^r ;$$
>
> le premier s'obtient en prenant p=r dans la première somme, ce qui introduit un facteur égal à
>
> $$\binom{n-1}{r},$$
>
> et le second s'obtient en prenant p=r-1 dans la seconde somme, ce qui introduit un facteur égal à
>
> $$\binom{n-1}{r-1};$$
>
> pour achever la démonstration, il reste donc à vérifier la relation
>
> $$\binom{n}{r} = \binom{n-1}{r} + \binom{n-1}{r-1}.$$

Le mathématicien est conscient des limites d'une telle argumentation, mais est surtout intéressé par sa pertinence. Les règles de la représentation formelle de Σ sont bâties de façon à respecter les propriétés de la représentation intuitive, tout en assurant l'impossibilité d'une intuition trompeuse. C'est pourquoi **la représentation intuitive constitue une synthèse pertinente pour trouver la démonstration et en communiquer les caractéristiques essentielles.**

Au lieu d'adopter un point de vue global s'intéressant à tous les termes, Roger Godement choisit ici une typologie des termes (ceux contenant le monôme "$x^{n-r} y^r$"). Il n'a pas ici les moyens mathématiques formels permettant une telle extraction ou la justification de l'invariance des termes relativement à cette typologie. Pourtant chaque mathématicien connaît la pertinence de cette décomposition, ainsi que les moyens d'accéder aux termes de cette forme à partir du Σ (prendre p=r ou p=r-1 ou p=? selon l'analyse du contenu du terme principal).

Roger Godement "démontre" par ce biais que la relation est vérifiée pour les n-1 termes tels que 0<r<n. Même si ce type d'argumentation était mathématiquement formalisé, cette démonstration serait **incomplète** puisqu'elle omet le cas des deux termes extrêmes du résultat final. Ce cas des limites est subsidiaire pour l'explication qui vise à convaincre le lecteur de la validité de la relation proposée.

4.4. ANALYSE DE QUELQUES ETAPES DE MANIPULATION DE FORMULES

4.4.1. Production du plan d'une démonstration

Un **plan de démonstration** est fréquemment utilisé pour la conduite du raisonnement. Il est organise des actions et des tâches possibles. Lors de son exécution, des décisions sont prises afin d'exécuter certaines actions ou certains tâches. Nous parlons d'*élémenis de décision* pour les données et les connaissances intervenant dans la décision, et d'*embarras* lorsqu'ils ne suffisent pas à privilégier nettement l'un des choix.

Nous voulons analyser ici les mécanismes de production d'un plan de démonstration. Dans notre exemple, Roger Godement déclare vérifier la relation :

$$\binom{n}{r} = \binom{n-1}{r} + \binom{n-1}{r-1}.$$

Toute conduite des calculs est complexe. Pour la mener à bien, les objectifs du mathématicien sont de :
- déterminer les connaissances disponibles en fonction des données ;
- trouver un plan de démonstration ;
- réaliser le guidage des opérations susceptibles de réaliser ce plan.

L'étape préliminaire consiste en une analyse de la relation afin de déterminer la nature de ses constituants, et les informations qui sont susceptibles d'y être rattachées. Cette relation est une égalité. Il n'y a aucune information utile sur les coefficients du binôme hormis leurs deux définitions : toute opération concernant ces coefficients doit y recourir. Chaque coefficient doit donc être transformé en utilisant l'une des deux définitions disponibles, de préférence la même pour rester homogène. La définition à choisir est à la discrétion du mathématicien.

La **stratégie de raisonnement** dirigée par les données est alors :
- évaluer les données du problème ;
- analyser les connaissances disponibles sur ces données ;
- choisir les connaissances à utiliser.

Il existe plusieurs **schémas de démonstration** envisageables pour démontrer une égalité (par équivalence, par double inégalité, etc.). Si le mathématicien choisit par exemple un schéma de démonstration par égalité, il lui reste à décider de quel membre partir (gauche, droite, les deux) et dans quel ordre procéder. Nous appelons cela l'**ordonnancement de démonstration**.

Vu l'absence d'information sur les relations directes entre coefficients du binôme, l'analyse préliminaire a montré qu'il faudra de toute façon développer chacun des deux membres. Une bonne **heuristique** consiste à le faire simultanément et envisager le problème ainsi transformé.

Une fois cela mentalement ou physiquement effectué, la poursuite de la stratégie de raisonnement consiste à évaluer les possibilités de continuation pour rendre les deux membres identiques. Ainsi :

$$\frac{n\,(n-1)\ldots(n-r+1)}{1.2\ldots r} \text{ est comparé avec } \frac{(n-1)\,(n-2)\ldots(n-r)}{1.2\ldots r} + \frac{(n-1)\,(n-2)\ldots(n-r+1)}{1.2\ldots(r-1)}.$$

Le membre de gauche ne donne pas d'indication de continuation, tandis que le membre de droit suggère une tâche générale de simplification de fractions. Cela requiert une réduction au dénominateur "1.2...r" suivie d'une simplification du numérateur ainsi obtenu. Une telle conduite des opérations est évidente pour un mathématicien. L'objectif final n'a d'ailleurs pas besoin d'être connu, ce qui évite en première analyse de procéder séparément par les deux membres.

Le mathémat[illegible]a donc hésité dans notre scénario pour trouver l'ordonnancement de la démonstration par égalité. La re[illegible]e d'éléments de décision lui a fait développer les données en parallèle jusqu'à trouver une piste intér[illegible]te à poursuivre isolément. L'ordonnancement choisi dans la démonstration omet cette démarche de re[illegible]erche heuristique : il peut donc y avoir des **travaux préparatoires découplés de la présentation finale.**

Cette analyse produit finalement le plan de démonstration suivant :

- démontrer par égalité à partir du membre droit :
  c'est le schéma de démonstration avec l'ordonnancement retenu ;
- développer en utilisant la définition des coefficients :
  la tâche résultant des contraintes de l'analyse initiale ;
- réduire au même dénominateur ;
- simplifier le numérateur ;
  les deux étapes du plan de simplification de fraction
- aviser !

Aviser indique qu'il faut coordonner le résultat obtenu avec le problème. Le savoir-faire du mathématicien indique que ce plan doit normalement aboutir - si la relation et la démonstration sont correctes - à la formule dépliée de la définition du coefficient de droite, puis se réduire à l'expression $\binom{n}{r}$. La simplification du numérateur vise donc un produit de facteurs et non des termes développés. Il y a donc une **connaissance "approchée" des manipulations** qui permet de :

- prédire grossièrement l'effet des manipulations (une fraction) ;
- savoir si cela correspond à des conclusions attendues (une autre fraction) ;
- déduire quel type de suite est nécessaire dans le plan (quelques ajustements mineurs ici vu que ce sont deux fractions, mais un problème ultérieur à résoudre si le résultat était une intégrale) ;
- tenir compte des conclusions attendues pour adapter les manipulations selon leur effet (préférer garder le numérateur factorisé) ;
- diagnostiquer des erreures éventuelles (une apparition qui ne s'intègre pas dans cette connaissance "approchée" paraît suspecte cf. [Cipra 83]).

Outre cet aspect très intuitif qui vient conforter la pratique du mathématicien, nous avons vu que **le mathématicien planifie sa conduite du raisonnement** en bâtissant un plan de démonstration partiel en fonction des données du problème. Le plan débute par un choix de schéma de démonstration et de son ordonnancement. Le mathématicien utilise pour cela une stratégie de résolution qui évalue les données puis analyse et choisit les connaissances.

4.4.2. Le raisonnement lors du déroulement des calculs

La planification et la conduite du raisonnement proprement dite apparaissent rarement dans une démonstration. La mathématicien lecteur est amené éventuellement à les reconstituer à partir d'indications fragmentaires. Cela correspond au rôle principal d'une démonstration qui est avant tout de présenter une solution.

La planification permet néanmoins d'expliquer la conduite du raisonnement observé dans la démonstration. Nous observons ici **l'instanciation du plan dans la démonstration**. Le déroulement des calculs de cette vérification d'une relation donne :

$$\binom{n}{r} = \binom{n-1}{r} + \binom{n-1}{r-1}.$$

Or le second membre est égal à

$$\frac{(n-1)\,(n-2)\ldots(n-r)}{1.2\ldots r} + \frac{(n-1)\,(n-2)\ldots(n-r+1)}{1.2\ldots(r-1)}$$

$$= \frac{(n-1)\ldots(n-r+1)\,(n-r) + (n-1)\ldots(n-r+1)\,r}{r!}$$

$$= \frac{[(n-r)+r]\,(n-1)\ldots(n-r+1)}{r!} = \frac{n\,(n-1)\ldots(n-r+1)}{r!} = \binom{n}{r}$$

Roger Godement choisit ici pour sa "vérification" la notation intuitive des coefficients du binôme, ce qui correspond à l'esprit de la "démonstration" dans laquelle cette vérification intervient. Nous raisonnons alors sur cette représentation algébrique intuitive pour décrire les propriétés qu'il utilise implicitement dans ses calculs.

La phrase de transition "Or le second membre est égal à" utilise le rupteur d'inférence "or" qui suggère une remarque. Ce rupteur pouvant être absent sans nuire au suivi de la transition, nous l'interprétons comme une **indication de l'analyse initiale** de la relation. Cette analyse a conduit le mathématicien au plan évoqué précédemment, qui commence la démonstration par égalité à partir du second membre.

Ce rupteur d'inférence "or" s'accompagne de plus habituellement d'une synthèse introduite par "d'où", "donc", "il en résulte que", etc. L'absence de synthèse tient ici à l'évidence visuelle de la conclusion. Cette **synthèse implicite** est néanmoins effectuée : elle consiste à remonter au niveau du problème posé et à le qualifier de résolu.

La conduite des calculs commence alors par la tâche de développement du second membre. Le mathématicien qui n'a pas la définition sous les yeux procède éventuellement par reconstitution mentale à partir du modèle opératoire qu'il a choisi pour la définition, par exemple : prendre la partie supérieure moins la partie inférieure du coefficient du binôme, puis ajouter un. Cela donne respectivement comme extrémités "(n-1)-r+1" et "(n-1)-(r-1)+1". Un petit calcul mental vient simplifier le résultat obtenu.

L'étape suivante du plan consiste à : "réduire au même dénominateur". La procédure à suivre pour cela résulte de l'expérience du mathématicien. Elle consiste ici à :

- chercher quel va être le dénominateur commun, si possible la plus petite expression englobante, en :
  - reconnaissant des facteurs communs dans les dénominateurs ;
  - gérant les facteurs ainsi distingués pour le dénominateur ;
  - mémorisant ces facteurs à l'intention des coefficients du numérateur ;
- écrire le dénominateur commun dans le résultat en cours de construction ;
- recopier chaque numérateur en le multipliant par les coefficients requis.

La reconnaissance de facteurs communs entre le produit de séquences d'entiers revient, dans la représentation intuitive, à déterminer les entiers communs à ces séquences. Cette comparaison est analogue à une comparaison d'intersections d'intervalles. Pour le dénominateur, il faut extraire les bornes des deux séquences ; comme les bornes inférieures sont identiques, il faut comparer les bornes supérieures et choisir la plus grande pour la plus petite expression englobante, et la plus petite pour les facteurs communs. Il sait alors que le premier terme n'a pas besoin de coefficient correctif, alors que le second doit être multiplié par r ...

Nous sommes ici à un niveau de détail très fin de la tâche. On pourrait croire que toutes la modélisation des opérations impliquées est directe, vu que les opérations sont simples à réaliser pour un mathématicien. Il n'en est rien ! La simplicité de l'éxécution des opérations masque la grande sensibilité du mathématicien aux données qu'il manipule. Une prise en compte astucieuse des données dans un cadre général de réduction au même dénominateur nécessite une algorithmique complexe.

Notre analyse de ces **opérations mentales** est que les traitements du mathématicien :

- s'adaptent à la nature des données (opportunistes)
- optimisent la mémoire de travail ;
- varient selon la présence d'un support visuel initial ou intermédiaire (partie du résultat écrit) ;
- possèdent souvent d'autres propriétés intéressantes : incrémentalité, optimisation de l'ordre de prise en compte des données, etc.

L'analyse de cette démonstration a notamment montré comment le mathématicien gère des tâches macroscopiques qui se décomposent jusqu'à des opérations mentales très détaillées que nous qualifions de microscopiques : nous parlons alors de ***micro-tâches*** et de ***micro-opérations***. Des exemples de micro-tâches sont alors chercher la plus petite expression englobante ou chercher les facteurs communs. Des exemples de micro-opérations sont d'écrire le dénominateur commun ou d'effectuer un calcul mental.

Cette notion de micro-opération est essentielle puisqu'elle décrit la façon dont les formules sont écrites. Une micro-opération permet la mise en œuvre d'un résultat mathématique, d'une action de présentation, etc. Elle ne se cantonne pas nécessairement à l'écrit directement observable et cherche à modéliser les actions mentales du mathématicien.

Les micro-tâches et les micro-opérations de transformation d'une formule s'appuyent essentiellement sur la forme visuelle de la formule. Pour rendre compte de l'importance de ces repères spatiaux dans la mise en œuvre et le suivi des transformations, nous introduisons la notion de ***structure prégnante*** qui traduit la forme (et le positionnement spatial) des parties principales de la formule.

Se repérant sur cette structure prégnante, le mathématicien gère successivement la réalisation des micro-tâches précédemment évoquées :

- écrire le dénominateur commun dans le résultat en cours de construction ;
- recopier chaque numérateur en le multipliant par les coefficients requis, qui se décompose à son tour...

Nous qualifions de ***zone d'intérêt*** la (ou les) zone(s) respectivement considérée(s) dans la réalisation de ces micro-tâches. Nous disposons ainsi d'un **modèle d'analyse des formules** proche des procédés employés par le mathématicien.

Notre propos n'est pas de découvrir le modèle mental exact du mathématicien, avec toutes ses subtilités, mais simplement de proposer un modèle de traitement susceptible de simuler un comportement, c'est-à-dire susceptible d'être présenté et accepté comme explication à un autre mathématicien. Pour résumer, **nous cherchons la plausibilité cognitive et non la validité cognitive.**

4.4.3. Présentation des étapes de démonstration

Nous nous sommes ainsi principalement intéressé à la production d'une preuve solution. Nous allons voir maintenant comment des micro-opérations sont insérées parmi celles issues d'une planification générale. Leur justification peut d'ailleurs donner lieu à une véritable expertise permettant une mise en forme fine des étapes afin de faciliter le suivi d'une démonstration.

La fraction obtenue par le procédé issu littéralement de la planification serait alors :

$$= \frac{(n-1)\,(n-2)\ldots(n-r) + (n-1)\,(n-2)\ldots(n-r+1)\,r}{1.2\ldots r}$$

Or Roger Godement écrit :

$$= \frac{(n-1)\ldots(n-r+1)\,(n-r) + (n-1)\ldots(n-r+1)\,r}{r!}$$

L'intention du mathématicien se traduit ici dans la **présentation des étapes de démonstration**. Un texte mathématique comporte alors une succession d'indices permettant de reconstituer les opérations effectuées et à venir, ainsi que leurs liens avec la résolution du problème.

Pour comprendre les intentions que cette présentation met en évidence, analysons les différences observées tout en dégageant des règles qui les expliquent. Rappelons que les "..." utilisent une représentation intuitive mieux appréhendée en terme de séquences et de bornes entières. Nous analysons toutefois ici que les micro-opérations aboutissant à la formule présentée.

- le **dénominateur** est réécrit selon une autre notation :
  il s'agit de simplifier la présentation d'un terme dont le sens n'intervient ultérieurement que lors de la transition aboutissant au résultat final.
  R0 : une fois le dénominateur écrit, le centre d'intérêt du mathématicien est le numérateur qu'il s'agit de simplifier ;
  R1 : un terme n'intervient pas directement dans une transformation lorsque le centre d'intérêt du mathématicien est fixé sur une autre zone (il reste alors invariant) ;
  R2 : un terme n'intervient pas directement dans des transformations lorsqu'il n'intervient pas directement durant plusieurs transformations successives ;
  R3 : lorsqu'un terme n'intervient pas directement dans des transformations, la préservation de son apparence syntaxique suffit à justifier son invariance ;
  R4 : l'analyse d'une forme syntaxique est facilitée par sa concision ;
  R5 : une bonne présentation cherche à faciliter dès que possible la concision ;
  R6 : les équivalences de notation bien maitrisées sont utilisées pour améliorer la présentation ;
  R7 : une équivalence de notation bien maitrisée peut-être utilisée sans gêne si l'état initial et l'état résultant sont explicites dans la transformation.

R8 : une équivalence de notation moins bien maitrisée est explicitée dans une transformation (elle fait l'objet principal de la transition ou est justifiée verbalement) ;
R9 : lorsqu'un terme n'intervient pas directement dans des transformations, les équivalences de notations bien maitrisées sont utilisées dès la première transformation pour améliorer la concision ;

Il y a donc des zones d'intérêt secondaire, tel le dénominateur, qui se contentent d'être recopiées et peaufinées sous certaines conditions. Les zones plus actives resteraient-elles intangibles ?

- le **terme de droite du numérateur** est réécrit de façon plus concise :
il s'agit de simplifier la présentation d'un terme en utilisant une forme plus abrégée du même procédé de notation, ici en supprimant le facteur intermédiaire "(n-2)".
R'0 : tout terme mathématique peut être analysé selon sa provenance, son utilisation dans la formule, et son devenir ;
R'1 : le terme de droite provient de la définition des coefficients du binôme ;
R'2 : toute utilisation d'une définition commence par une instanciation littérale au cas considéré (l'instanciation littérale comporte trois facteurs dont le second est "((n-1) -1)") ;
R'3 : lorsque l'application d'une règle doit être reconnue, seules quelques simplifications triviales sont autorisées (d'où "(n-2)") ;
R'4 : dans une transformation de développement par dépliage de définitions, les définitions utilisées doivent être reconnues ;
R'5 : une notation est utile pour signifier une intention et pour présenter des éléments pertinents dans les manipulations ;
R'6 : le deuxième facteur intervient pour clarifier l'intention de la notation et non comme élément pertinent dans les manipulations ;
R'7 : une fois le sens d'un terme reconnu, la clarification de l'intention devient secondaire par rapport à la lisibilité des manipulations ;
R'8 : la notation "..." peut être abrégée en explicitant uniquement le facteur initial et le facteur final ;
R'9 : la suppression de "(n-2)" augmente la lisibilité des calculs ultérieurs sans nuire à la provenance du terme ni à la clarté de l'interprétation de la notation ;

La notation des séquences d'entiers avec "..." est donc de deux termes initiaux lors de l'introduction d'une séquence[2], puis du seul terme indispensable après. Nous expliquons cela par le fait que la représentation intuitive est construite et présente dans l'esprit du mathématicien, et que des indices écrits suffisent ultérieurement. Nous avons ainsi observé quelques mathématiciens effectuant la simplification à partir de la donnée isolée de la formule précédente. La simple analyse de la formule et de ses groupements pose problème alors qu'elle devient évidente avec la donnée de la formule précédente.

Conformément à R'0, cette analyse traduit l'influence de la provenance et de l'utilisation d'un terme dans la formule ; il reste à illustrer l'influence de son devenir :

- le **terme de gauche du numérateur** est réécrit différemment :
il combine la suppression précédente de "(n-2)" avec l'introduction de l'avant-dernier terme "(n-r+1)".
R"0 : les notations employées en mathématiques ont pour but de faciliter le travail du mathématicien de façon à :
  - permettre l'interprétation aisée du sens de la notation si besoin ;
  - offrir des lois de manipulation de la notation simples à utiliser ;
  - permettre, grâce à leur forme visuelle, l'introduction aisée des éléments pertinents aux manipulations ;

R"1 : une étape d'une démonstration est une synthèse entre la clarification des transformations passées, celle de l'état actuel, et la préparation des transformations à venir ;
R"2 : lorsque la suite des transformations doit exploiter une propriété (ici une mise en facteur), il est judicieux de rendre visible la propriété commune ;
R"3 : la suite des transformations cherche à faire apparaître les facteurs "(n-1)...(n-r+1)" dans le terme "(n-1)...(n-r)" ;
R"4 : par la représentation intuitive qu'elle utilise, la notation "..." permet la visualisation de toute sous-séquence par visualisation de ses extrémités ;
R"5 : la notation "..." permet l'introduction de tout facteur intermédiaire selon le procédé bien connu qui consiste à ...
R"6 : la préparation de la transformation suivante requiert l'introduction du facteur intermédiaire "(n-r+1)" dans le terme de gauche du numérateur.

---

[2] Dans son traité, Roger Godement s'est permis d'aller jusqu'à trois pour la définition de la factorielle.

Cette analyse permet de mesurer la complexité des lois qui régissent la présentation de formules lors d'une démonstration. Elle montre que le mathématicien exploite implicitement une véritable ***expertise de présentation de formules et de preuves*** dans sa pratique quotidienne. Elle illustre de plus que cette expertise n'est pas inaccessible et qu'elle peut être formalisée, notamment à partir d'une analyse linguistique détaillée des démonstrations mathématiques.

Les règles présentées dans cette analyse d'une étape de transformation de formule n'ont aucune prétention de modélisation complète et valide des lois qui régissent la présentation des formules dans une démonstration. La verbalisation de l'expertise ainsi explicitée se prête d'ailleurs elle-même à être travaillée de la sorte.

4.4.4. Le suivi des étapes de démonstration

L'analyse de l'étape précédente a permis de montrer comment la simplification de la formule avait été préparée par la mise en évidence du facteur commun du numérateur pour obtenir :

$$= \frac{(n-1)\ldots(n-r+1)\,(n-r) + (n-1)\ldots(n-r+1)\,r}{r!}$$

La micro-tâche principale est alors de simplifier le numérateur (zone d'intérêt principal). L'analyse du numérateur (micro-tâche d'analyse obligatoire) précède toute autre micro-tâche ou micro-opération. Elle peut s'expliquer récursivement à l'aide de la notion de structure prégnante et de zone d'intérêt à l'intérieur du numérateur. La structure prégnante est alors "AAA BBB + CCC DDD", les zones d'intérêt locales s'arrêtant sur les termes parenthésés.

Le résultat de cette analyse, inséré dans la structure prégnante générale donne :

$$\frac{\text{COMMUN}\ (n-r) + \text{COMMUN}\ r}{\text{XXX}} \quad \text{avec COMMUN représentant } (n-1)\ldots(n-r+1)$$

Les abréviations "COMMUN" et "XXX" indiquent :
- que le mathématicien ne s'intéresse plus à leur contenu ;
- qu'il se contente ultérieurement dans l'action d'identifier leur forme et de les recopier ;

Une fois cette analyse réussie, les mathématiciens observés respectent une séquence d'écriture nullement obligatoire, et obtiennent les formules ci-dessous en suivant strictement l'ordre d'écriture de chaque caractère (voire "(n-r+r)" s'ils abrègent déjà) :

$$\frac{\text{COMMUN}\ [(n-r)+r]}{\text{XXX}} \quad \text{qui est simplifié en} \quad \frac{\text{COMMUN}\ n}{\text{XXX}}$$

L'écriture de la transformation effectuée peut alors se résumer par les micro-opérations suivantes :
- recopier la partie commune "COMMUN" ;
- prendre successivement les termes à regrouper et les écrire ;
- tirer un trait et recopier le dénominateur "XXX" ;

L'influence du contexte général de la démonstration intervient puisque Roger Godement écrit d'emblée la mise en facteur à gauche et non à droite comme cela se produirait s'il n'y avait inversion. Il écrit :

$$= \frac{[(n-r)+r]\,(n-1)\ldots(n-r+1)}{r!} = \frac{n\,(n-1)\ldots(n-r+1)}{r!} = \binom{n}{r}$$

Ce résultat met en évidence que même les micro-opérations effectuées par le mathématicien sont influencées par les buts ultérieurs qui ne se concrétiseront ici qu'à la dernière étape de la démonstration.

Il illustre aussi que **le suivi d'une démonstration nécessite un recalage permanent** des attentes du lecteur par rapport aux productions proposées par l'auteur. Le lecteur doit ici prendre en compte la formule produite par l'auteur afin de constater l'inversion. Ce recalage est bien évidemment inutile pour l'auteur d'une démonstration qui n'a qu'à verbaliser son état mental.

Une **explication possible de cette inversion observée** est que les contenus cognitifs de "COMMUN" et "XXX" sont présents dans la mémoire de travail du mathématicien lecteur et que ce dernier n'a plus besoin de renouveler une analyse précédemment effectuée. Ainsi, le mathématicien est conscient de leur rôle dans sa globalité, et en particulier de leur devenir. C'est pourquoi, il sait que le "n" va devant "COMMUN" et non après, afin de compléter la séquence. Le fait qu'un mathématicien ne s'intéresse pas au contenu de "COMMUN" est donc à relativiser.

L'analyse de cette séquence de simplification met toutefois en évidence trois possibilités de positionnement du terme simplifié issu de "[(n-r)+r]", selon qu'il est inversé à gauche immédiatement, dans la formule intermédiaire, ou pas du tout. Le fait qu'il soit inversé à gauche immédiatemment dans l'exemple nous permet de conclure que **l'inversion immédiate était préférable.** Cela entraîne une comparaison avec les deux autres solutions.

Ecrire $\frac{(n-1)\ldots(n-r+1)\,n}{r!} = \binom{n}{r}$

entraîne notamment une réorganisation mentale du numérateur après avoir déplié mentalement le numérateur attendu dans le coefficient du binôme.

Ecrire $\frac{[(n-r)+r]\,(n-1)\ldots(n-r+1)}{r!} = \frac{(n-1)\ldots(n-r+1)\,n}{r!}$

entraîne un effet de disparition du premier facteur, disparition que l'on retrouve après simplification et analyse du numérateur du membre de droite.

Dans ces deux cas, le coût mental du suivi de la transformation est relativement grand. Il se traduit notamment, par rapport à la solution de référence, par un surcoût dû à une analyse détaillée du numérateur. En revanche, grâce à une préparation judicieuse de l'étape précédente, la transformation :

$$\frac{(n-1)\ldots(n-r+1)\,(n-r) + (n-1)\ldots(n-r+1)\,r}{r!} = \frac{[(n-r)+r]\,(n-1)\ldots(n-r+1)}{r!}$$

ne demande guère de surcoût mental. L'analyse détaillée du numérateur du membre gauche est déjà effectuée par le mathématicien pour l'effectuation des micro-tâches. Il ne lui reste plus, lorsqu'il cherche le point d'arrivée du regroupement de "(n-r)" et "r", qu'à le situer à gauche au lieu du point plus probable à droite.

L'**importance de la structure prégnante** se traduit donc dans la complexité de la micro-tâche d'analyse de la formule. Une modification de la structure prégnante entraîne une analyse détaillée considérant la provenance des termes. Une mauvaise gestion de ces modifications a pour conséquence une surcoût d'analyse superflu.

4.5. LA PRODUCTION DE PREUVES ET DEMONSTRATIONS

4.5.1. La conduite des calculs

Nous adoptons maintenant la représentation formelle des coefficients du binôme pour aborder cette preuve, non plus comme explication d'une démonstration, mais dans l'optique de sa production. La relation s'écrit maintenant :

$$\binom{n}{r} = \binom{n-1}{r} + \binom{n-1}{r-1}.$$

L'étape initiale de planification reste inchangée et trois dépliages de la définition choisie pour les coefficients du binôme amène la mathématicien à comparer :

$$\frac{n!}{p!\,(n-p)!} \quad \text{avec} \quad \frac{(n-1)!}{(p-1)!\,(n-p)!} + \frac{(n-1)!}{p!\,(n-1-p)!}$$

Ici encore, seul le membre de droit donne des indications de continuation afin de retrouver le plan précédemment énoncé :

- réduire au même dénominateur ;
- simplifier le numérateur ;
- aviser !

La structure prégnante du membre droit est la forme générale :

$$\frac{N_1}{D_1} + \frac{N_2}{D_2}$$

La micro-tâche "réduire au même dénominateur" :
- analyse les deux zones d'intérêt D_1 et D_2 lors de sa réflexion actuelle ;
- compare leurs facteurs ;
- met en évidence leurs facteurs communs ;
- trouve et recopie le plus petit facteur englobant ;

La structure prégnante de D_1 et D_2 est "X Y" et se détaille afin de comparer les facteurs. Le ***thème d'intérêt*** qui justifie ce détail est ici la nature de son contenu. Il arrive que le thème d'intérêt soit uniquement la structure, qu'il soit le degré de la variable x pour un monôme, ou qu'il soit la recherche des occurences de variables lors d'un changement de variables.

Le résultat de l'analyse de chaque terme selon ces thèmes d'intérêt constitue le ***contenu caractéristique*** du terme considéré. Une telle recherche de contenu caractéristique pourrait être modélisée récursivement par micro-tâches et zones d'intérêt, mais en dessous d'un seuil de complexité, le résultat émerge dans la mémoire de travail. De façon similaire avant ce seuil, l'ordonnancement éventuel entre micro-tâches perd toute rigidité.

L'analyse détaillée requise par la comparaison des deux dénominateurs : (p-1)! (n-p)! et p! (n-1-p)!
distingue ici comme contenu caractéristique respectif : "p-1" "n-p" et "p" "n-1-p".

La suite des opérations consiste ici à examiner le degré de proximité de ces ingrédients. "p-1" est associé à "p" tandis que "n-p" est associé à "n-1-p". Notons que cette proximité joue principalement sur la forme actuelle des ingrédients en préservant toutes les interprétations, sans envisager le cas trop particulier où p est égal à n-p.

Une telle **analyse qualitative** n'est utile qu'en collaboration avec les règles de manipulation des expressions. La proximité du contenu des factorielles n'est intéressante que parce qu'il existe une règle permettant de les relier par une ou quelques applications successives de : "n! = n (n-1)!". Les deux associations précédentes vont pouvoir être concrétisées par l'application simple de cette règle. Nous qualifions les inférences implicites que cela implique de ***micro-inférences***.

La mise en évidence des facteurs communs obtient donc un résultat de la forme :

A (n-p) B et p A B

Le mathématicien est alors en mesure de trouver le plus petit facteur englobant et l'exploiter de plusieurs façons :

- Il peut le faire directement en écrivant le dénominateur du résultat à venir :
  $\frac{}{A\ (n\text{-}p)\ B\ p}$ (par ex., choix des facteurs visibles par balayage de zone d'intérêt de gauche à droite).

- Il peut mémoriser cette analyse et remonter au niveau de la structure prégnante générale pour exploiter une fraction de la forme :
  $\frac{N}{A\ (n\text{-}p)\ B} + \frac{N}{p\ A\ B}$ et commencer par écrire le numérateur N p + N (n-p) par zones d'intérêt "croisées".

Dans ces deux cas, s'il ne simplifie pas simultanément le numérateur, il obtient :

$$\frac{(n-1)!\ p + (n-1)!\ (n-p)}{(p-1)!\ (n-p)\ (n-p-1)!\ p}$$

Pour faire face à cette diversité de situations et de choix possibles, il est intéressant de disposer d'un **modèle de traitement mental** susceptible de décrire le raisonnement du mathématicien, ses compétences, ses préférences. Un tel modèle doit permettre de :

- proposer des étapes de démonstration acceptables (suivi et qualité des explications) ;
- évaluer la difficulté cognitive de production et de suivi d'étapes de transformation de formules ;
- expliquer le choix d'étapes et les causes d'erreurs de calcul éventuelles .

Le principal résultat que nous retiendrons de cette analyse des transformations de formules est qu'il n'y a pas d'opération "simple" en mathématiques. La conduite des calculs de cette réduction au même dénominateur et les micro-opérations qu'elle requiert le montrent clairement. Il reste donc beaucoup de travail à effectuer avant de produire un modèle de traitement mental satisfaisant.

Nous avons proposé un modèle de traitement mental par micro-tâches et micro-opérations, qui permet par exemple une analyse des formules à l'aide de structures prégnantes et de zones d'intérêt. Ce modèle permet de décrire et de justifier partiellement les transformations de formules et est susceptible d'évaluer la difficulté cognitive de leur suivi et de leur production.

4.5.2. La détermination des étapes de preuve

Une preuve constitue un **réseau de dépendances**. La planification et la conduite des calculs choisies ont une importance cruciale sur le contenu des étapes produites. Réciproquement, l'action sur l'une des étapes visibles a des incidences sur la cohérence globale de la preuve.

La diversité des formes possibles, ainsi que la variabilité de l'évolution des formules font qu'il existe une grande variété de démonstrations. De nombreuses transitions simples effectuent en effet plusieurs opérations simultanées. Rien n'empêche a priori de les effectuer séparément, et cela dans un ordre relativement libre. Si l'on rajoute la possibilité d'intervertir l'ordre des termes, l'espace des formules possibles est démesuré.

Pourtant cette diversité est réduite par le fait que les preuves retenues sont optimales, selon des lois de régularité assez imprécises. La granularité choisie dans la preuve **régule la complexité** des calculs. Ainsi, les étapes proposées par différents mathématiciens pour compléter une preuve sont peu nombreuses, parce qu'elles résultent déjà d'une **organisation des calculs**.

L'organisation d'une preuve à l'aide de formules est ainsi trop grossière pour traduire les véritables raisons qui déterminent la formule suivante. Le travail du mathématicien consiste précisement à regrouper les opérations de façon à obtenir une progression homogène à la place d'une succession d'opérations trop fines. C'est pourquoi, nous avons introduit des aspects microscopiques pour expliquer le traitement mental d'un mathématicien.

L'état de la formule globale n'est souvent qu'une progression parallèle mais indépendante de certaines tâches jusqu'à un rendez-vous nécessitant une synthèse. Ainsi par exemple, la simplification du numérateur et celle du dénominateur s'effectuent en parallèle mais de façon indépendante à partir de :

$$\frac{(n-1)!\ p + (n-1)!\ (n-p)}{(p-1)!\ (n-p)\ (n-p-1)!\ p}$$

Rien n'empêche a priori de simplifier le numérateur en laissant le dénominateur inchangé, et créer ainsi une étape inédite dans une preuve réaliste. Même si les simplifications sont menées de front, il est aussi possible que l'une soit nettement plus longue, obligeant par là même à dupliquer inconsidérement la forme simplifiée de l'autre branche.

Des principes ergonomiques viennent régir les étapes de preuves. Ces principes peuvent être expliqués en terme de coût des opérations de traitement.

Un principe ergonomique essentiel est d'éviter de dupliquer des expressions compliquées. De nombreux choix de production (et de présentation) peuvent s'analyser grâce à lui. Il justifie par exemple la nécessité de certaines dérivations dans le déroulement d'une preuve. C'est le cas lorsqu'une grosse partie de formule reste longtemps inchangée.

Une variante de ce principe incite à regrouper des termes égaux ou semblables. Ce regroupement de termes évite une duplication en production et diminue le complexité d'analyse. Par exemple, il vaut mieux utiliser la règle, sauf autre raison, :

$$\frac{A}{D}+\frac{B}{D} \quad \text{donne} \quad \frac{A+B}{D}$$

Un autre principe ergonomique consiste à remarquer que lorsqu'un mathématicien utilise une formule compliquée ou peu familière, il a besoin de la visualiser sous une forme proche. Ainsi dans notre démonstration initiale Roger Godement écrit p145 :

> Le résultat à établir est trivial pour n=1 ; il suffit donc de prouver que la formule
>
> $$(x+y)^{n-1} = \sum_{p=0}^{p=n-1} \binom{n-1}{p} x^{n-1-p} y^{p}$$
>
> implique la formule analogue pour l'exposant n. Or, en multipliant par x+y la relation précédente, il vient

Il est facile de modifier les phrases précédentes de façon à supprimer le rappel de cette formule. Toutefois, son absence sera préjudiciable pour le suivi du lecteur qui devra évoquer mentalement la formule en question. L'étape suivante n'étant pas ici d'un secours suffisant pour la reconstituer aisément, le lecteur devra aller explicitement rechercher la relation initiale dans l'énoncé du théorème.

La production d'étapes de preuve est donc régie par certaines **lois de régularité** :

- la détermination d'un seuil de complexité acceptable pour enchainer les étapes de la preuve ;
- le découpage des transformations d'une formule en micro-tâches distinctes avec des points de rendez-vous ;
- les micro-opérations visibles localement se parallélisent dans chaque zone d'intérêt de façon à respecter un seuil de complexité de la transformation de l'étape ;
- des micro-opérations sont enchaînées si possibles dans une zone d'intérêt tant que leur succession respecte le seuil de complexité de la transformation de l'étape ;
- lorsqu'une zone d'intérêt est la seule à évoluer durant plusieurs étapes, les calculs sont séparés de la preuve principale de façon à ne pas alourdir le tout inutilement.

Rappelons enfin que la complexité d'une étape de preuve est une complexité cognitive relative à la charge de travail du mathématicien. L'utilisation de règles ou de théorèmes mathématiques puissants permet une évolution importante des calculs au prix d'une complexité de transformation souvent minime. La production d'une preuve consiste bien souvent à chercher à employer ces règles puissantes.

La **gestion des transformations de formule** est ainsi similiaire à un processus de planification, d'ordonnancement de plan (par instanciation en respectant les contraintes de parallélisme, précédence, etc.) et de suivi de son exécution. Elle permet de trouver un chemin critique où les ressources sont exploitées au mieux de leur efficacité.

La différence majeure avec les problèmes de productique et de robotique est la souplesse de ce processus. Il dispose en effet de nombreuses qualités souhaitées (détermination du chemin critique, instanciation partielle, réactivité aux données, incrémentalité, opportunisme, récupération de plan, etc.).

4.5.3. Plusieurs voies possibles dans une preuve

Nous reprenons la preuve de la relation entre coefficients binomiaux afin d'examiner les voies de développement envisageables de ces manipulations de formules. Pour cela, nous adoptons une approche orientée vers la découverte d'une preuve. Rappelons que le problème consiste à établir la relation :

$$C_n^p = C_{n-1}^{p-1} + C_{n-1}^{p}$$

Afin d'identifier les diverses voies possibles, il faut revenir au début de l'analyse du problème, après développement du membre droit. Le problème issu de la planification est de "mettre sous forme de fraction simplifiée" un terme dont le mathématicien a identifié la forme prégnante générale, qu'il va

ultérieurement raffiner selon les données observées et les règles envisageables. Le mathématicien commence par examiner la formule :

$$\frac{(n-1)!}{(p-1)!\,(n-p)!} + \frac{(n-1)!}{p!\,(n-1-p)!} \text{ de la forme générale } \frac{N_1}{D_1} + \frac{N_2}{D_2} \text{ (somme de deux fractions)}$$

Comme toujours lors de la résolution d'un problème, il y a deux grandes voies possibles :

- **résoudre directement**, c'est-à-dire mettre en œuvre des opérations permettant de ramener le problème à une construction ;
- **simplifier le problème**, c'est-à-dire le transformer en un autre problème (ou plusieurs) que l'on espère résoudre ;

Une solution pour le résoudre directement consiste à réduire les fractions au même dénominateur, après mise en évidence de ses facteurs communs. La règle de déduction est alors :

$$\frac{N_1}{B\,D_1} + \frac{N_2}{B\,D_2} = \frac{N_1\,D_2 + N_2\,D_1}{B\,D_1\,D_2}$$

La deuxième voie consiste à simplifier le problème. Une méthode de simplification consiste à réduire la taille de ses arguments : c'est la fonction principale de la mise en facteurs. La règle de distributivité "xy + xz = x (y+z)" s'applique ici sous la forme adaptée :

$$\frac{A\,N_1}{B\,D_1} + \frac{A\,N_2}{B\,D_2} = \frac{A}{B}\left(\frac{N_1}{D_1} + \frac{N_2}{D_2}\right)$$

Nous dénombrons alors deux voies de preuve de la relation entre coefficients du binôme, selon qu'il y a mise en facteur générale ou non (cf. figure 1.18).

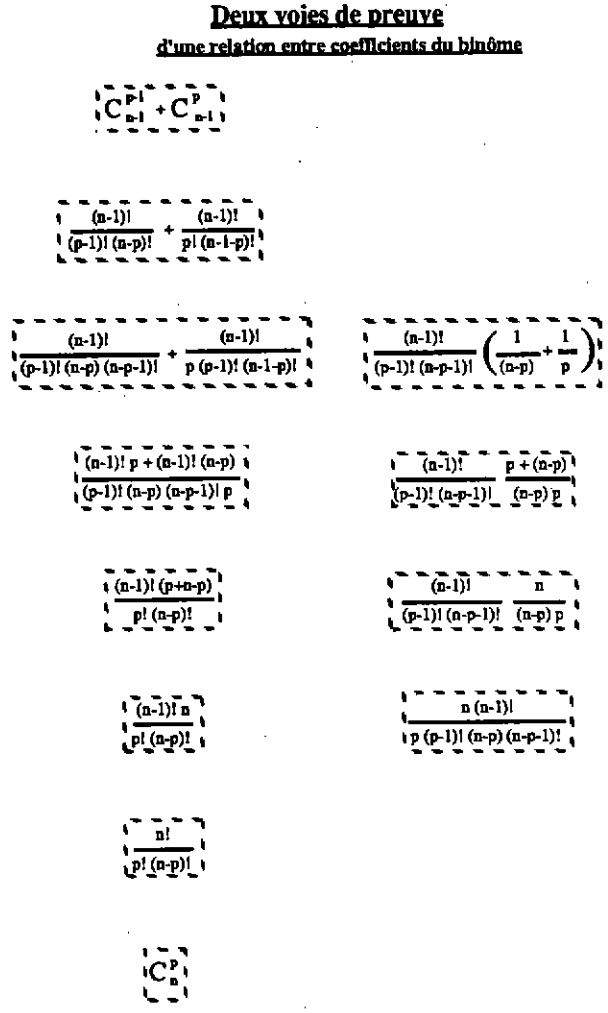


figure 1.18

4.5.4. Des parcours selon plusieurs niveaux de détail

Quelques parcours possibles de la preuve

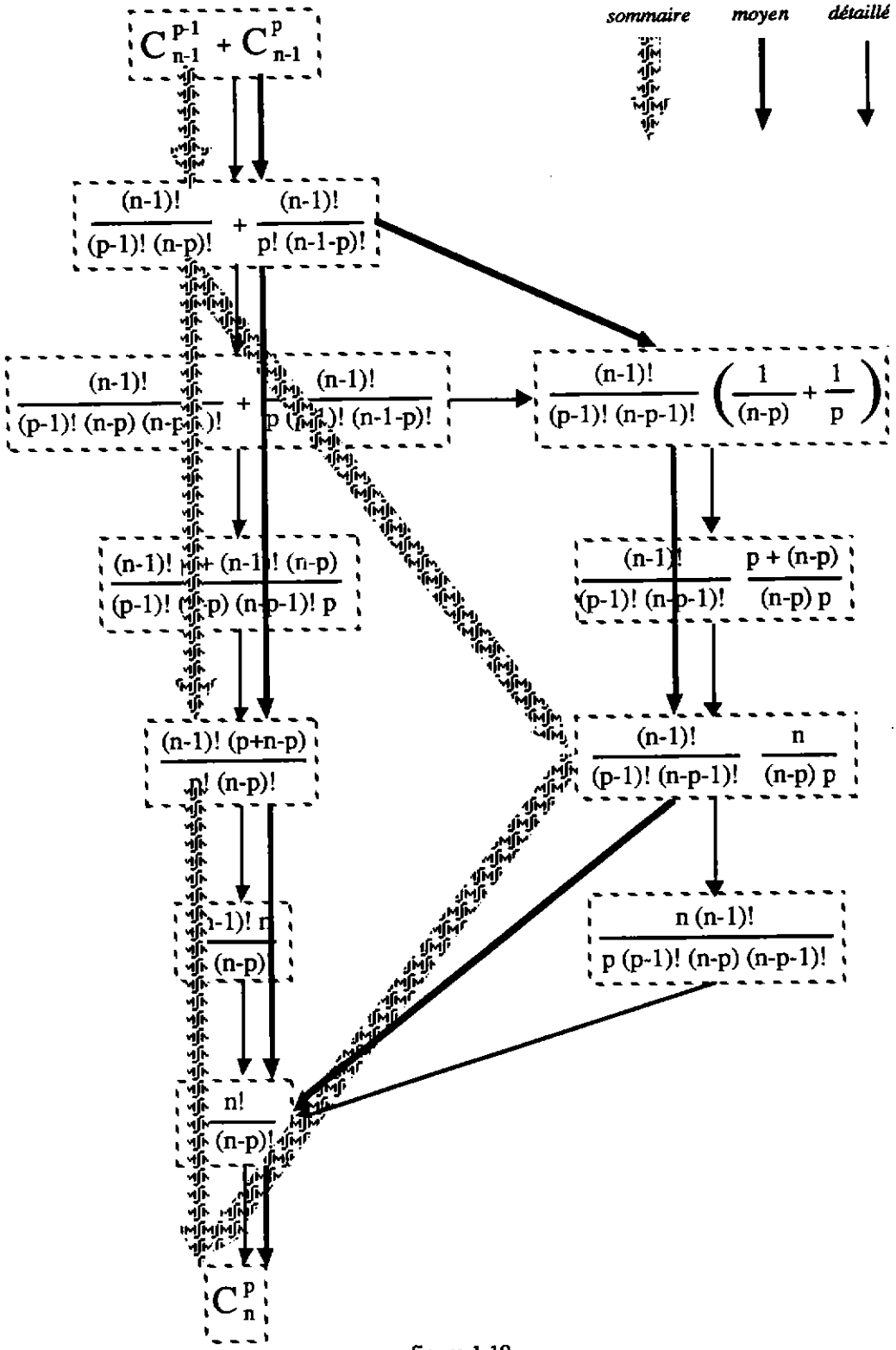


figure 1.19

Nous appelons *parcours* d'une preuve, la succession d'étapes suivie par le mathématicien. Quelle que soit la voie choisie, le mathématicien a le choix entre plusieurs niveaux de détail dans la description, le niveau zéro étant celui de l'énoncé de la relation, c'est-à-dire de l'état initial et de l'état final (cf. figure 1.19).

Le choix d'un parcours dépend de nombreux paramètres. Dans la mesure où les diverses possibilités sont détectées et estimées, le mathématicien cherche à **privilégier le suivi des transformations.**

Pour un parcours moyen ou sommaire, nous préférons la voie de droite qui met mieux en évidence les éléments caractéristiques des transitions (les facteurs communs). Pour un parcours détaillé, nous préférons la voie de gauche plus courte et aussi explicite. Cette concision est due à une règle de réduction au même dénominateur efficace grâce à la prise en compte partielle de facteurs communs.

Des lois de régularité dans l'évolution des transformations permettent d'expliquer correctement le choix des étapes. Pour illustrer leur influence, notons que le raccordement final de la voie de droite n'a qu'une seule possibilité par niveau de détail, car les autres liaisons envisageables correspondraient à une évolution locale ou globale irrégulière. Il serait par exemple inopportun de ne pas simplifier "n (n-1) !" en "n!" au numérateur et de l'effectuer au dénominateur.

D'autres critères interviennent aussi, comme l'intention du mathématicien ou la prise en compte du lecteur. L'absence d'une règle sophistiquée de réduction au même dénominateur chez un apprenant ferait préférer une description détaillée par la voie de droite (le parcours par la gauche devrait sinon être rallongé).

4.5.5. Habillage d'une preuve en démonstration

L'exemple de la relation entre coefficients du binôme nous a montré comment une preuve pouvait être décrite à des niveaux différents ou avec des notations différentes. Le niveau minimal est, bien sûr, celui de l'énoncé de la relation. Dans ce cas, la preuve peut-être laissée "en exercice" ou supposée effectuée par ailleurs, seul le résultat étant utilisé.

$$\binom{n}{r} = \binom{n-1}{r} + \binom{n-1}{r-1}.$$

Ce résultat se démontre trivialement en développant les coefficients.

On vérifierait par récurrence que :

$$C_n^p = C_{n-1}^{p-1} + C_{n-1}^p$$

Quelques **démonstrations présentant des preuves différentes** de cette relation sont disponibles ci-après. Nous remarquons en particulier que l'habillage choisi dépend notamment du niveau de granularité adopté. Lorsqu'une démonstration est succinte, le mathématicien a besoin (et peut se permettre) d'un habillage plus riche afin d'améliorer la compréhension.

$$\binom{n}{r} = \binom{n-1}{r} + \binom{n-1}{r-1}.$$

Or le second membre est égal à

$$\frac{(n-1)\,(n-2)\ldots(n-r)}{1.2\ldots r} + \frac{(n-1)\,(n-2)\ldots(n-r+1)}{1.2\ldots(r-1)}$$

$$= \frac{(n-1)\ldots(n-r+1)\,(n-r) + (n-1)\ldots(n-r+1)\,r}{r!}$$

$$= \frac{[(n-r)+r]\,(n-1)\ldots(n-r+1)}{r!} = \frac{n\,(n-1)\ldots(n-r+1)}{r!} = \binom{n}{r}$$

En développant le second membre, on obtient :

$$C_{n-1}^{p-1} + C_{n-1}^{p} = \frac{(n-1)!}{(p-1)!\,(n-p)!} + \frac{(n-1)!}{p!\,(n-1-p)!}$$

$$= \frac{(n-1)!}{(p-1)!\,(n-p)\,(n-p-1)!} + \frac{(n-1)!}{p\,(p-1)!\,(n-p-1)!}$$

$$= \frac{(n-1)!\,p + (n-1)!\,(n-p)}{(p-1)!\,(n-p)\,(n-p-1)!\,p}$$

$$= \frac{(n-1)!\,(p+n-p)}{p!\,(n-p)!}$$

$$= \frac{(n-1)!\,n}{p!\,(n-p)!}$$

$$= \frac{n!}{p!\,(n-p)!}$$

$$= C_n^p$$

Une remarque permet d'illustrer les avantages et les limites d'une telle preuve détaillée : qu'ai-je réellement lu dans cette preuve et quelles ont été les synthèses que j'ai effectué lors de la lecture ? Nous retrouvons ici la problématique de la lecture et de la compréhension de texte, la lecture anticipant l'essentiel au point de ne pas lire des erreurs ou des duplications du texte (non, il n'y en a pas ici, mais cette vérification nécessite une seconde lecture pour le garantir !).

Montrons la relation : $C_n^p = C_{n-1}^{p-1} + C_{n-1}^{p}$

Le second membre donne : $\frac{(n-1)!}{(p-1)!\,(n-p)!} + \frac{(n-1)!}{p!\,(n-1-p)!}$

en factorisant on obtient : $\frac{(n-1)!}{(p-1)!\,(n-p-1)!}\,\frac{n}{(n-p)\,p}$

ce qui est le développement de C_n^p .

Développons la relation, on a :

$$C_{n-1}^{p-1} + C_{n-1}^{p} = \frac{(n-1)!}{(p-1)!\,(n-p)!} + \frac{(n-1)!}{p!\,(n-1-p)!}$$

$$= \frac{(n-1)!}{(p-1)!\,(n-p-1)!}\left(\frac{1}{(n-p)} + \frac{1}{p}\right)$$

$$= \frac{(n-1)!}{(p-1)!\,(n-p-1)!}\,\frac{n}{(n-p)\,p}$$

$$= \frac{n!}{p!\,(n-p)!}$$

$$= C_n^p$$

4.6. SYNTHESE DE NOTRE ANALYSE DU LANGAGE

Notre objectif pratique d'expliciter et de spécifier l'usage du Langage Mathématique nous a fait étudier en détail des textes produits par des mathématiciens. Il s'avère que **l'usage du langage reflète la conduite de l'activité cognitive du mathématicien.** Cela est particulièrement vrai en mathématiques car les mathématiciens ont développé, lors d'une maturation millénaire, une argumentation commune et reproductible reposant sur des concepts écrits à visée universelle.

Nous avons analysé lors de ce chapitre :

- un texte de **résolution de problème** (surface de la parabole) afin de disposer d'une vue globale de la phase d'utilisation ;
- des passages discursifs permettant de mettre en évidence les ingrédients de la **maîtrise des écritures** par les mathématiciens ;
- une démonstration basée sur des **manipulations de formules** afin de cerner la complexité des paramètres en présence.

Cette analyse a illustré la pertinence de notre **distinction entre preuve et démonstration.** Cette distinction revient à séparer les aspects textuels "de surface", des aspects mathématiques "profonds" qui assurent la cohérence et explique la forte variabilité apparente. Parmi ces aspects profonds, nous situons le consensus de l'argumentation entre mathématiciens, dont la preuve constitue une trace. Ces aspects profonds servent de support à une **description**, puis à une **présentation** textuelle ou graphique.

Notons que cette analyse ne s'intéresse pas aux heuristiques de résolution et de découverte. Il nous importe, en effet, de rester proche de la ligne de démarcation du langage afin d'étudier un **corpus de connaissances presque suffisant à la production de preuves et démonstrations.** L'impression principale à l'issue de notre analyse est que la nature, la diversité et la quantité de connaissances nécessaires à cela étaient largement sous-estimées.

En particulier, nous avons isolé l'importance des **capacités de perception et de calcul** dans la maîtrise de l'écriture, c'est-à-dire dans :

- la façon dont le mathématicien raisonne ;
- les représentations qu'il utilise ;
- les inférences qu'il effectue ;
- les connaissances qu'il exploite.

Nous avons illustré le rôle des aspects "microscopiques" dans une **modélisation de l'activité mathématique** chargée de produire et d'interpréter des démonstrations. Nous avons introduit les éléments d'une expertise de manipulation de formules et de construction de preuves. En particulier, l'intérêt essentiel d'un modèle du mathématicien consiste à proposer des critères de décision pour cette modélisation détaillée de l'activité.

Nous avons préféré effectuer des synthèses partielles à proximité des passages mathématiques et de leur analyse. Cela permet de bien discerner la provenance des résultats en vue d'une extension ou d'une réfutation. Les analyses de textes sont amenées à être poursuivies et les résultats produits complétés. Notre analyse a pour objectif de mettre en évidence l'intérêt et l'urgence d'une **approche linguistique du Langage Mathématique** afin d'aboutir à une maîtrise partielle des paramètres qui interviennent dans la production de démonstrations et dans la modélisation de l'activité du mathématicien.

La partie II présente l'intérêt externe de ce travail, ainsi que le support informatique nécessaire à une symbiose entre cette recherche linguistique et sa modélisation informatique.

5. L'USAGE DES PREUVES MATHEMATIQUES

5.1. L'INTERET DE PREUVES A GRANULARITE VARIABLE

5.1.1. La diversité des preuves possibles

Les étudiants français produisent annuellement plusieurs millions de constructions baptisées "démonstration". Leur structure reste toutefois essentiellement empirique et leur validité ne repose que sur l'expérience de leurs professeurs. Beaucoup de recherches se sont focalisées sur l'art de trouver des preuves, alors que beaucoup de progrès restent à faire sur la façon de construire, représenter et manipuler ces preuves et les connaissances qu'elles utilisent.

Nous nous sommes décidés pour cela à choisir un traité d'exercices et chercher à représenter le texte d'une démonstration, les déductions qu'elle exprime et les opérations nécessaires à leur réalisation. Certaines démonstrations ne se modélisent que très imparfaitement : les liens entre déductions paraissant complexes ou incomplètement exprimés. Une démonstration utilise en effet des procédés de descriptions langagiers pour donner des indications sur une construction que le mathématicien doit être capable d'effectuer.

Nous avons introduit la dichotomie entre preuve et démonstration afin de clarifier l'analyse : la preuve renvoit à l'objet que le mathématicien est sensé construire pour résoudre son problème, et la démonstration à l'objet textuel chargé de présenter cette construction de façon fragmentaire. Une démonstration est donc pour nous une présentation, un habillage textuel par lequel le mathématicien communique sa preuve solution.

Cette séparation entre une preuve et sa présentation permet d'envisager d'autres présentations d'une preuve utilisant des procédés de description plus complets et mieux structurés que ceux des démonstrations usuelles. C'est pourquoi, nous nous intéressons ici à la nature et à la construction des preuves, sachant qu'elles constituent un modèle "profond" permettant de bâtir aussi des démonstrations.

Aboutir d'un problème à une preuve varie en fonction de grandes orientations (résolution géométrique ou analytique), de choix d'objets (parmi un jeu d'objets équivalents comme pour les fonctions trigonométriques) ou de notations, et surtout de la prise en compte des données. Le développement d'une preuve donne ainsi lieu à de multiples possibilités.

La **diversité des preuves envisageables** dépend de trois paramètres :
- l'espace des possibilités ;
- les circonstances ;
- l'usage escompté.

L'espace des possibilités est souvent démesuré en mathématiques, les possibilités réalistes parmi un groupe homogène de mathématiciens sont nombreuses... Deux mathématiciens ne produisent alors des preuves identiques que dans des circonstances bien déterminées.

Par ailleurs, une preuve peut être construite dans des circonstances très différentes :
- lors de sa découverte par un mathématicien (construction heuristique) ;
- ibid., mais avec une grande maîtrise heuristique (construction d'application) ;
- lors d'un dialogue explicatif (construction à la "carte") ;
- lors de la lecture d'une présentation de preuve existante (construction guidée, la preuve étant reconstituée par le lecteur), etc.

Ces circonstances très différentes font que des preuves identiques sont issues de **processus de construction** très variés. De plus, comme cela a été indiqué précédemment pour les démonstrations, il y a plusieurs usages conjoints des preuves :
- descriptif : capturer la progression de la pensée du mathématicien ;
- évaluatif : adapter une preuve de façon aussi détaillée que nécessaire pour juger sa validité ;
- explicatif : choisir un point de vue, comme par exemple détailler certains points ;
- épistémique : construire une preuve destinée à un usage communautaire.

L'usage descriptif permet avant tout de capturer une progression : en utilisant ça et ça comme cela, j'obtiens cette proposition. L'usage évaluatif cherche à préciser tous les éléments de cette progression en utilisant des opérations "rigoureuses" afin de juger la nature de la proposition ainsi obtenue (cette proposition est suffisante). Il suggère donc un certain recul par rapport à la progression.

Nous partitionnons ces usages des preuves selon deux types de tendances :
- trouver-aménager ;
- vérifier-communiquer ;

En fonction de ces tendances, les usages précédents correspondent grossièrement à :

tendances retenues	*trouver*	*aménager*
vérifier	usage descriptif	usage évaluatif
communiquer	usage épistémique	usage explicatif

5.1.2. Nos objectifs de modélisation d'une preuve

Nous avons cherché à **adapter la notion de preuve aux besoins des mathématiciens.** Pour cela, nous avons cherché à conserver les informations utilisables et utiles ultérieurement, dans les preuves que le mathématicien est amené à construire. La construction incrémentale (c.-à-d. progressive par morceaux) et la réutilisation partielle sont alors des thèmes d'intérêt majeurs de la construction d'une preuve.

Nous sommes fréquemment en présence de plusieurs preuves d'un même résultat, chaque preuve résultant d'un processus de construction différent. De plus, le mathématicien a fréquemment besoin de preuves différentes selon ses usages. Même lorqu'il aménage une preuve pour un autre usage, la preuve d'origine peut souvent être amenée à être utilisée ultérieurement.

Nous avons alors abandonné un abord strict de la notion de preuve pour chercher à regrouper sous l'appellation de preuve à granularité variable des informations "voisines", susceptibles d'être utilisées en fonction des besoins du mathématicien dans l'exploitation d'un résultat mathématique. L'objectif dérivé des besoins du mathématicien est alors :

Permettre au mathématicien des utilisations multiples d'une preuve.

La notion de preuve correspond ainsi à une structure discrète partielle dont certaines parties sont incomplètes ou à détailler, d'autres sont abandonnées, d'autres sont nouvellement construites, d'autres enfin sont redondantes et correspondent à plusieurs usages. L'argumentation du mathématicien est alors susceptible d'emprunter l'un des chemins possibles.

Cette diversité d'utilisation est le pendant de la diversité de construction d'une preuve. Nos objectifs détaillés pour la modélisation d'une preuve sont alors de pouvoir **s'inspirer de l'activité des mathématiciens** afin de :
- détailler ou abstraire à volonté certains passages ;
- autoriser des preuves incomplètes et leur combinaison ;
- tolérer et regrouper des variantes de présentation ;
- utiliser librement les procédés de description mathématiques ;
- faire collaborer des abords intuitifs ou plus formels ;
- produire une démonstration à partir d'une preuve ;

Une preuve à granularité variable cherche à concilier ces objectifs afin de s'adapter à la pratique des mathématiques. Une preuve est ainsi "*à granularité variable*" selon qu'elle est raffinée ou abstraite à l'aide d'opérations de déduction plus ou moins détaillées. Nous avons alors introduit une terminologie associée, nous appelons :
- *parcours* d'une preuve, la succession d'étapes suivie par le mathématicien ;
- *niveau de détail*, un choix homogène, plus ou moins détaillé, des transitions du parcours ;
- *voie choisie*, un faisceau de parcours utilisant des arguments mathématiques semblables ;
- *choix de description* d'une preuve, le choix guidant un parcours, comme par exemple présenter le résultat avant d'en détailler les étapes ;
- *présentation*, la concrétisation d'une partie d'une preuve, par exemple sous forme d'une structure interactive susceptible d'être détaillée à volonté.

Nous obtenons ainsi une schématisation d'une preuve mathématique à granularité variable (figure 1.20).

Preuve mathématique à granularité variable

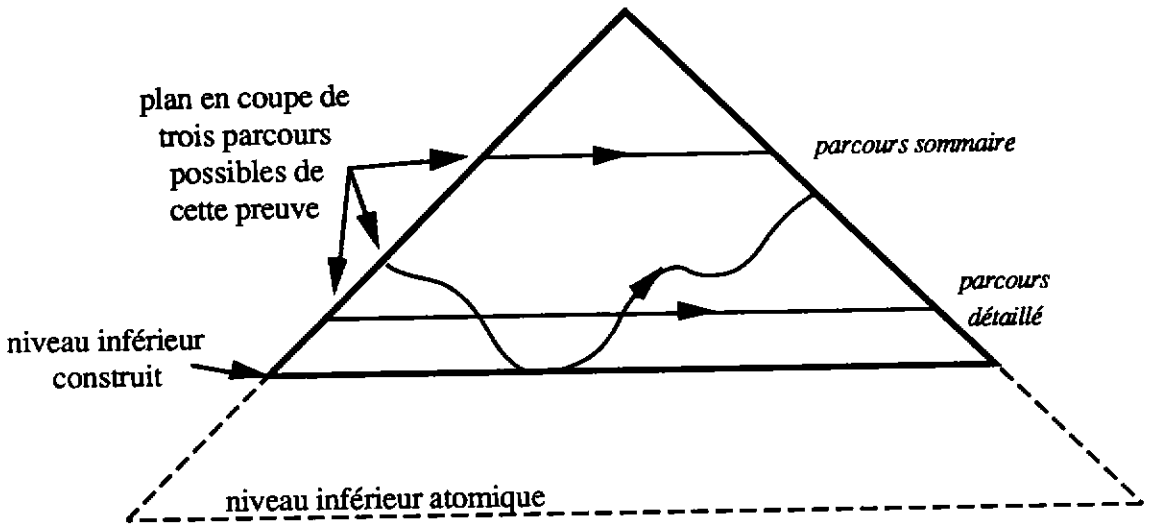


figure 1.20

Le niveau de détail du parcours d'une **preuve mathématique à granularité variable** peut ainsi varier jusqu'au niveau inférieur de sa construction. Ce niveau peut toujours être affiné tant qu'il n'est pas considéré comme atomique. En pratique le mathématicien se contente d'un niveau inférieur tel que l'évidence permette de considérer la preuve comme valide.

5.1.3. Un modèle de preuve à granularité variable

Le but d'une preuve à granularité variable est de pouvoir respecter le procédé de construction du mathématicien. Cela implique une **structure souple** afin de capturer le jeu incessant du changement de points d'intérêt du mathématicien. Cette structure est donc un graphe peu homogène.

Le détail de la structure est complexe à établir car les états sont eux-mêmes disloqués et regroupés, et les opérations de déduction utilisées pour cela ne sont pas reconnues en tant que telles. De plus certaines opérations de déduction sont l'application de résultats qui donnent eux-même lieu à la création d'une preuve. Ces autres preuves ont été construites ailleurs (théorèmes), précédemment (dans le but de former cette opération de déduction) ou le seront peut-être ultérieurement.

L'analyse de l'activité du mathématicien débouche sur un constat essentiel :

Les états sont précis, tandis que les transitions qui les relient sont flous.

Le mathématicien qui construit une preuve fait évoluer des objets qui en constituent la trame et marquent des points de passage du raisonnement. Cette façon de procéder allie **souplesse et précision**. La précision des états choisis est celle des objets mathématiques de la trame. La souplesse provient des transitions qui peuvent être imparfaitement décrites et de la possibilité d'étendre ou restreindre le nombre de points de la trame.

Les états et les transitions sont ainsi des ingrédients essentiels d'une preuve. Les *tâches* et le *contexte* sont deux autres ingrédients associés à la notion de preuve. Le mathématicien introduit ou fait référence à des objets et des relations dont l'ensemble est appelé *contexte*. Son évolution se fait conjointement à la

construction de la preuve et un contexte local est rattachable à chaque état selon le niveau de détail. Le contexte est ainsi une utilisation de la théorie qui ne s'intéresse qu'aux objets utilisés dans la preuve.

Une transition est complètement définie par :
- son point de départ :
  contexte de départ de l'état et conditions d'accès sur le contenu de l'état ;
- leur équivalent pour le point d'arrivée ;
- les moyens de transformation :
  preuve, règle et théorèmes avec les liens de correspondance utilisés pour rendre les contextes compatibles. Des conditions annexes peuvent aussi être à vérifier (tel $a \neq 0$).

Les moyens de transformations étant chargés d'assurer la souplesse de la preuve, ils peuvent à leur tour se décomposer en moyens mis en parallèle ou en séquence. Ils peuvent aussi se résumer plus simplement par une tâche à effectuer.

Une transition peut de plus être étiquetée par plusieurs sortes d'informations comme celles relatives aux parcours possibles (niveau de détail, etc.), la nature mathématique du lien (égalité ou équivalence), la nature intuitive ou formelle de la transition, la tâche qu'elle réalise, etc.

Il existe alors dans une preuve **deux sortes de niveau de détail**, selon qu'il agit sur les états ou les transitions :
- un *niveau de profondeur*, en fonction de l'introduction d'états intermédiaires ;
- un *niveau de décomposition*, en fonction du détail des composantes intervenant dans les transitions. Cette décomposition utilise par exemple le niveau de détail de preuves annexes.

Un des points importants dans notre modélisation d'une preuve à granularité variable est la détermination des états. Lorsque les états sont tous de même nature, comme cela est fréquent dans les calculs, le déroulement de l'activité du mathématicien est essentiellement séquentiel. Lorsque les règles ne relient que deux états, nous obtenons un ***segment de preuve***. Dans l'exemple de la relation entre coefficients du binôme, un état est alors une formule alors qu'une transition correspond à l'utilisation de règles et de théorèmes mathématiques. Ceci constitue le premier type de transition.

D'autre part, les formules et les conjonctions d'énoncé sont fréquement décomposées en mathématiques. L'exemple de la relation entre coefficients du binôme a montré qu'un état, qui est à priori une relation d'identité à valider, l'est souvent par l'intermédiaire de formules résultant de la décomposition de ses deux membres. Il y a donc **décomposition et recomposition entre états** de nature différente pour prouver un résultat. Ceci constitue l'autre type de transition, utilisant des ***opérations de déduction***, par opposition aux transitions par ***opérations mathématiques***.

Lorsque le mathématicien construit une preuve à granularité variable, il est amené à construire des segments de preuve à plusieurs emplacements faiblement corrélés. Un exemple simple est celui d'un changement de variable nécessitant une **preuve annexe** pour déterminer les substitutions à effectuer. Le segment de preuve est alors en annexe d'une opération mathématique. Une preuve annexe de A à B peut aussi être utilisée en justification de A=B. La partie principale et la preuve annexe sont toutes deux intégrées dans une même structure de preuve à granularité variable.

Une preuve à granularité variable est donc un ensemble d'états qui sont des constructions du mathématicien. Ils sont reliés à l'aide d'opérations pas forcément complètement définies mais servant de justification. Ces opérations sont susceptibles d'être détaillées et de produire ainsi d'autres états à un niveau de détail inférieur. Chaque état est associé à un contexte qui donne les objets connus à ce niveau de détail. Il y a donc deux processus complémentaires fondamentaux dans la construction d'une preuve :
- un processus de construction d'états, qui s'appuye sur les données et les buts intermédiaires ;
- un processus de justification de transition, qui cherche à obtenir des transitions acceptables ;

Ce processus de justification de transition n'applique pas toujours les même critères d'acceptabilité. Il peut d'abord faire appel à des justifications intuitives et partielles, puis rechercher ultérieurement des justifications formelles. Il peut laisser en suspens certaines justifications, ou se contenter de conjectures parfois erronées. Il peut enfin plus classiquement déduire que telles transitions restent à justifier pour que la preuve soit valide et le problème soit résolu.

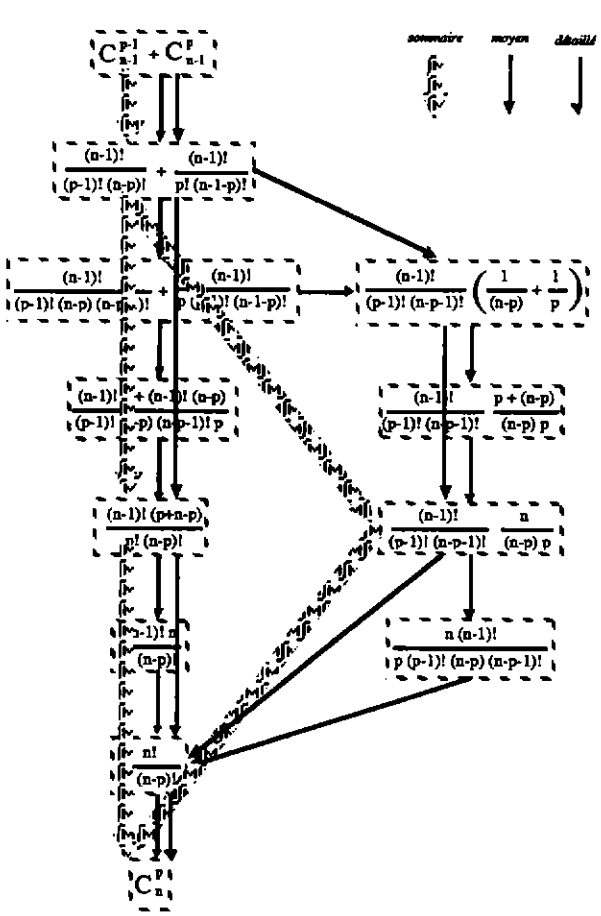
Ce modèle de preuve à granularité variable n'est pas complètement formalisé a priori, ce qui évite des choix péremptoires. Cela permet par exemple de pouvoir l'adapter aux opérations de déduction observées en mathématiques. Cela laisse ainsi la possibilité d'affiner sa définition en fonction des progrès de notre connaissance de l'activité mathématique.

L'intérêt d'une **étude expérimentale** de la construction d'une preuve par le mathématicien est d'identifier, recenser et formaliser les opérations employées. Ces opérations serviront en retour à valider la notion de preuve à granularité variable, dans la mesure où elles rendent compte de l'activité pratiquée par le mathématicien.

5.2. LA CONSTRUCTION DE PREUVES A GRANULARITE VARIABLE

5.2.1. Les segments de preuve à granularité variable

Nous considérons un **segment de preuve** comme en enchaînement linéaire d'états reliés par des transitions. L'état initial et l'état final sont toujours accessibles et leur donnée constitue la présentation minimale de la preuve : nous parlerons de *niveau zéro*. Deux états appartenant à un même enchaînement, ou même niveau, peuvent toujours être détaillés et devenir les extrémités d'une autre preuve située à un niveau inférieur. Ces deux états appartiennent alors simultanément à plusieurs niveaux. Un segment matérialise le déroulement d'une tâche conceptuelle que se fixe le mathématicien.



Le niveau de détail exposé pour la relation entre coefficients du binôme concerne le **niveau de profondeur** (rappel de la figure 1.19). Le mathématicien a ici deux paramètres principaux à fixer : le niveau de profondeur, et la voie de résolution. La voie de gauche présente quatre niveaux de profondeur selon le profil suivant :

Niveaux de profondeur

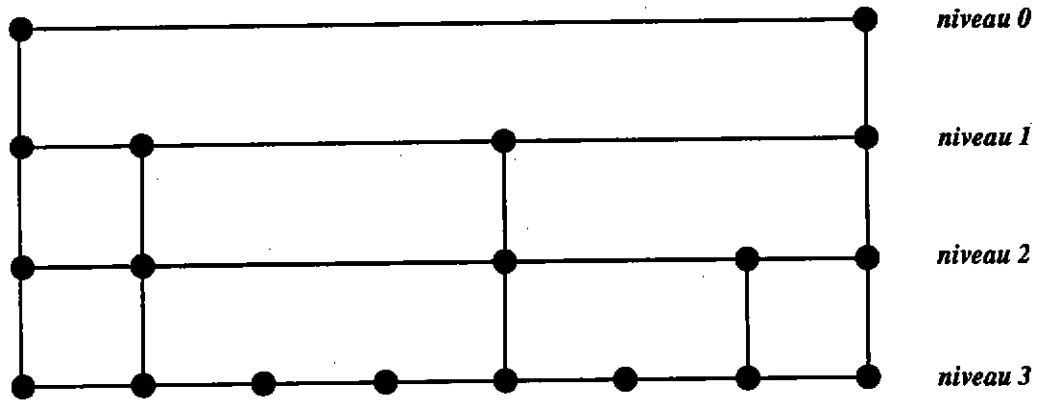


figure 1.21

Les liens verticaux représentent des états partagés par des niveaux différents. La succession des niveaux de profondeur s'effectue selon des critères de régularité étrangers à une décomposition régulière des états. De plus, la différence entre niveaux n'est pas forcément importante (entre 1 et 2). Enfin, la hiérarchie entre niveaux n'est pas forcément totale[1]. La notion de niveau de profondeur offre ainsi un cadre permettant des parcours différents d'un segment de preuve, selon des critères choisis par le mathématicien.

Nous avons vu par ailleurs que les deux voies représentaient l'alternative suivante de la résolution du problème :

- résoudre directement, c'est-à-dire mettre en œuvre des opérations permettant de ramener le problème à une construction ;
- simplifier le problème, c'est-à-dire le transformer en un autre problème (ou plusieurs) que l'on espère résoudre ;

La voie de droite - simplifier le problème - a ici été traitée au sein d'un même segment de preuve. Pourtant, la simplification d'un problème s'effectue souvent par une décomposition explicite en sous-problèmes. Il s'agirait dans notre cas de traiter séparément, de façon explicite, la sous-expression suivante:

$$\left(\frac{1}{(n-p)} + \frac{1}{p}\right)$$

Une preuve annexe peut ainsi être effectuée pour simplifier cette sous-expression. La recomposition du problème consiste à remplacer cette sous-expression par $\frac{n}{(n-p)\,p}$ dans la formule générale.

Nous dirons dans ce cas que cette preuve annexe s'est effectuée à un **niveau de décomposition** inférieur à celui de la preuve principale. Cela situe le segment de preuve du niveau de détail inférieur comme justification de l'opération mathématique qui a effectuée le remplacement.

Il est important de noter que le mathématicien n'est pas obligé d'introduire un niveau de décomposition. Le choix qu'il effectue est alors guidé par des relations de causalité et d'ergonomie dans la production de la preuve ou la présentation de la démonstration. Dans notre exemple, il peut décider d'effectuer ce calcul auxiliaire, mais ne pas le faire explicitement car il est trop court.

[1] On pourrait par ex. avoir plusieurs niveaux 2 non comparables en faisant varier l'état à rajouter au niveau 1.

5.2.2. Les niveaux de décomposition

L'exploitation de la relation d'équidistance de la parabole fournit un exemple typique de l'existence de plusieurs niveaux de détail emboités. Le segment principal part de cette relation : « la parabole est le lieu géométrique des points M équidistants d'un point fixe P, appelé foyer, et d'une droite D appelée directrice. ». Il transforme ensuite MA=MP en $\rho = p+\rho\cdot\cos\theta$ en passant implicitement, à un niveau de profondeur inférieur, par $p-\rho\cos(\pi-\theta) = \rho$.

La figure 1.22 présente la transition de MA=MP à $p-\rho\cos(\pi-\theta) = \rho$, que nous appelons niveau de référence R0. La structure d'une telle preuve annexe dépend des états qui sont mis en avant lors de sa construction. Elle serait différente par exemple si l'argument principal au niveau R0 était MA = PB-PM'.

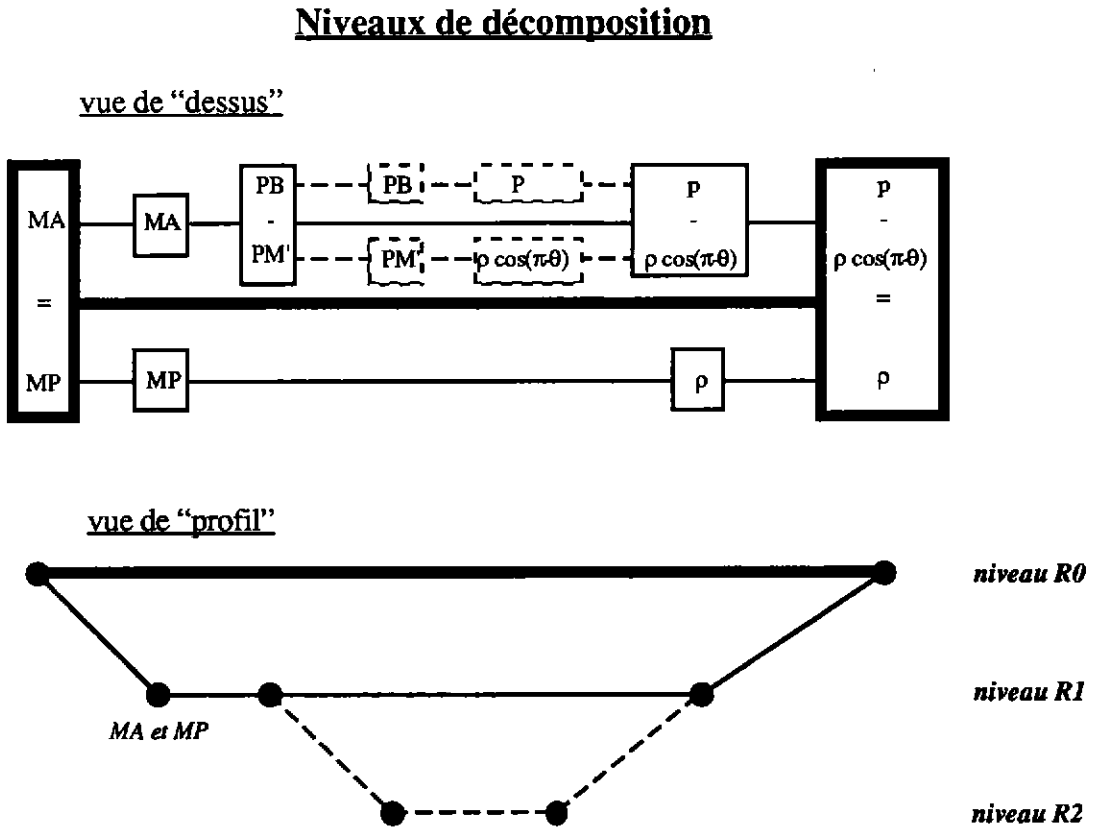


figure 1.22

Ce niveau R0 correspond à l'énoncé de la transition. Cette transition peut être acceptée comme telle, ou être décomposée au niveau R1, qui peut à son tour être décomposé en R2. Le niveau R1 est obtenu à partir du niveau R0 par isolement de deux sous-parties qui sont respectivement ici les deux membres. Il y a donc accès à la structure de l'état, puis isolement. La recomposition finale s'effectue par regroupement inverse après substitution de chacune des sous-parties isolées.

La transition inter-niveaux consiste donc essentiellement à gérer la structure interne de l'état. Cela introduit un parallélisme de description avec un facteur de branchement aussi élévé que nécessaire. Nous obtenons ainsi une description graphique parallèle et séquentielle d'une preuve.

Notons que la difficulté de la transition n'est pas directement dépendante d'un niveau. La transition de niveau R1 : MP = ρ, n'est qu'une utilisation de définition alors que la transition de niveau R2 : PM' = $\rho\cos(\pi-\theta)$, nécessite un raisonnement géométrique important.

A l'opposé des niveaux de décomposition, les niveaux de profondeurs présentent par définition des difficultés progressives en fonction du niveau. La motivation d'un niveau de décomposition étant essentiellement structurelle, elle ne peut présenter une difficulté aussi régulière. Le fait de gérer des

décompositions et leurs recompositions introduit néanmoins une difficulté spécifique aux niveaux de décomposition, qui nécessite à elle seule le recours à des étapes intermédiaires.

5.2.3. Le parcours effectif d'une preuve

Lorsqu'il trouve une preuve, ou pis encore lors qu'il l'aménage, le mathématicien utilise des parcours de preuve très variés. La notion de preuve à granularité variable offre une modélisation relativement indépendante d'un choix de description spécifique. Le mathématicien peut alors choisir un parcours en privilégiant, selon l'emplacement, un parcours à partir des données, des conclusions ou des résultats intermédiaires. Ces possibilités permettent une souplesse indispensable dans l'utilisation d'une preuve.

Nous présentons en Annexes quelques parcours d'un segment de preuve, par l'un des bouts ou les deux, [Solow 82, p 14-15]. L'auteur y présente plus précisément les habillages textuels correspondant à ces parcours.

En exploitant les niveaux de détail, le parcours d'une preuve à granularité variable dépend principalement des points d'entrée dans la structure et du sens choisis. Les habillages mathématiques comme les exercices pédagogiques rendent bien compte des possibilités de parcours :

	$MA = PB-PM' = p-\rho\cos(\pi-\theta)$,	①
or	$MA = MP = \rho$,	②
d'où	$\rho = p+\rho\cdot\cos\theta$.	③

Nous retrouvons ci-dessus le parcours choisi dans la démonstration de J. Quinet. La conclusion étant déjà une simplification d'une déduction stricte, elle est plus loin encore au niveau R0. Cette présentation aborde en trois temps les niveaux R1 et R0 selon la figure 1.23 suivante :

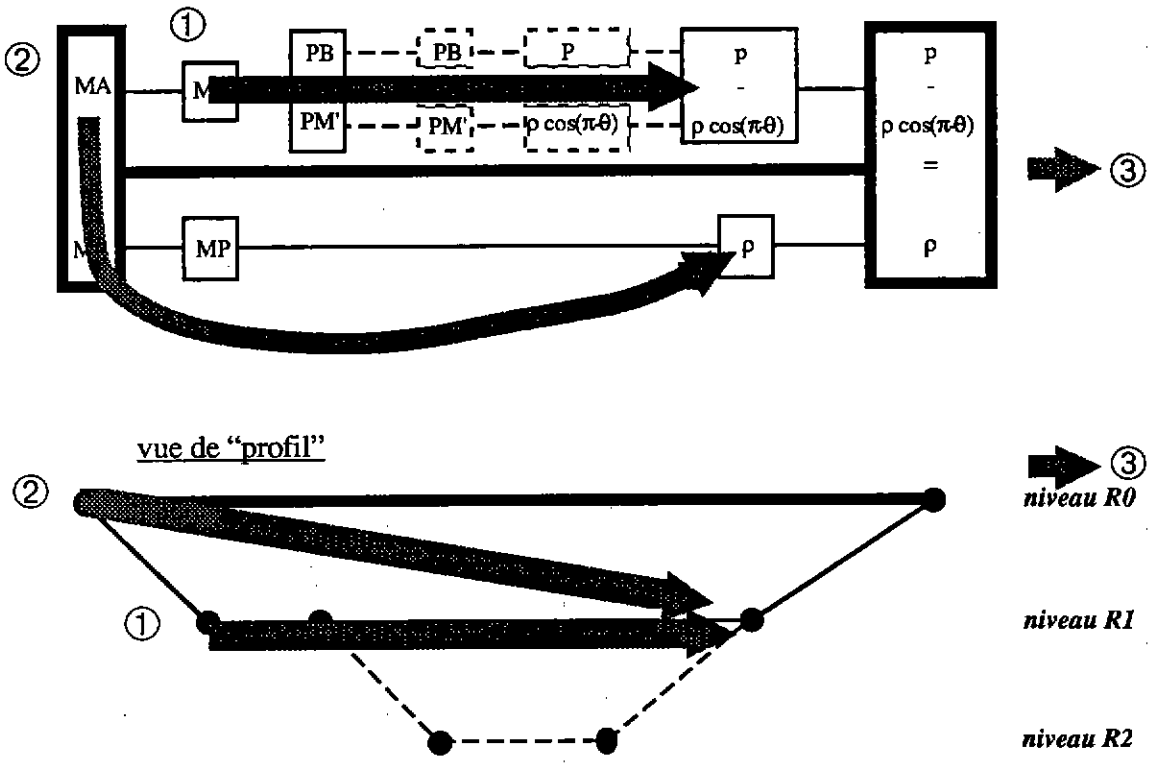


figure 1.23

Ce balayage part des données situées vers le bas à gauche pour aller vers le but cherché. Il rend implicite les opérations de décomposition. Des parcours plus simples peuvent donner lieu à des règles plus systématiques :

on a la relation
MA = MP
d'où $\rho = p+\rho\cdot\cos\theta$
en effet ...

Le parcours précédent effectue une présentation au niveau R0 avant de détailler à sa guise les niveaux inférieurs ;

montrer $\rho = p+\rho\cdot\cos\theta$ en utilisant la relation MA = MP.

Cette présentation adopte un sens inverse en partant de la conclusion afin d'aller vers les données ;

on a la relation :
MA = MP,
or la figure donne :
MA = PB-PM' = $p-\rho\cos(\pi-\theta)$,
d'autre part, on a par définition :
MP = ρ,
d'où $\rho = p+\rho\cdot\cos\theta$.

Le parcours précédent explore méthodiquement la preuve, en respectant les niveaux de décomposition. Il explicite le point d'entrée au niveau R0, décrit successivement les deux branches du niveau R1, puis retourne au niveau R0.

ρ = MP = MA = PB-PM' = $p-\rho\cos(\pi-\theta)$
d'où $\rho = p+\rho\cdot\cos\theta$

Ce dernier parcours balaye d'abord le niveau R1 à partir de l'inconnue ρ, avant de progresser au niveau R0. Ce parcours pourrait correspondre à un segment de preuve "par égalité", tel que son niveau zéro (les deux extrémités) forme un état dans un autre segment de preuve "par équivalence". Ici ce segment correspond au niveau R0, qui relie $\rho = p-\rho\cos(\pi-\theta)$ à $\rho = p+\rho\cdot\cos\theta$, etc.

Nous avons vu pourquoi cette autre possibilité de construction de la preuve à granularité variable n'a pas été retenue : elle minimise la donnée principale MA=MP, et suggère un parcours ne respectant pas une transcription géométrique-algébrique. Nous retrouvons ici **l'influence des données** sur la structure de la preuve à granularité variable, et **l'importance de la liberté de choix du mathématicien** dans sa construction.

5.3. LA RESOLUTION DES PROBLEMES MATHEMATIQUES

5.3.1. Problèmes et contextes dans une preuve

Après avoir exploré certains aspects des preuves mathématiques, nous abordons maintenant leurs liens avec les problèmes et les possibilités de modélisation de la résolution des problèmes mathématiques.

Une preuve peut être considérée comme la réponse à un problème et à son contexte d'objets introduits. Nous obtenons alors une modélisation théorique où tout est problème. A l'intérieur d'une preuve chaque couple d'états (voire chaque recouvrement connexe) peut être associé à un problème différent (qui consiste à les relier). Toutefois ce mécanisme de modélisation est très lourd et peu représentatif des présentations du mathématicien. Il faut réserver cet arsenal aux cas où le sous-problème est plus explicite.

problèmes et contextes dans une preuve

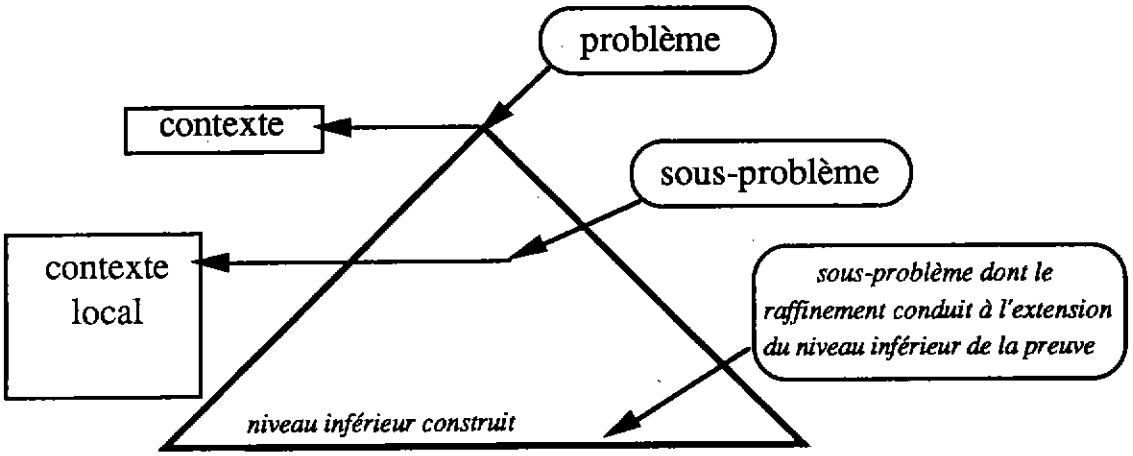


figure 1.24

La notion de sous-problème avec son contexte est néanmoins indispensable pour modéliser l'activité du mathématicien. Elle permet notamment d'adapter des connaissances générales à une situation spécifique. Nous définissons un **problème** par le quadruplet C.H.A.C. :

- Contexte ;
- Hypothèses ;
- Action ;
- Conclusions ;

Nous regroupons les hypothèses et le contenu du contexte sous l'appellation données. Le contexte (général, par opposition au contexte local d'un état de la preuve) regroupe des données stables qui sont utilisées dans la construction des hypothèses et des conclusions. Les hypothèses regroupe des données dont la validité est susceptible d'être remise en question pour démontrer les conclusions. L'action définit la nature du problème (prouver, trouver, explorer, etc.). C'est ainsi qu'en présence de propriétés sur des objets, on se retrouve face à trois types de problèmes de la forme "action tel que P(x)" : existence, dénombrement et caractérisation des relations entre objets vérifiant cette propriété.

Lorsque la formulation du problème est imprécise, préciser les conclusions fait partie du problème. Par exemple pour le problème "calculer S", nous avons une action "calculer" et une *construction partielle* "la surface S". La conclusion sera par exemple "la relation $S=\pi$ est vraie". L'action prend alors en argument une construction partielle, et la conclusion complète cette construction et émet un jugement.

Dans l'exemple de "l'existence d'un symétrique" (section 3), le problème posé par le système MAD est :

- Contexte :
  une théorie logique, un groupe G, une loi de composition interne + et un élément distingué 0 ;
- Hypothèses :
  $\forall l_1, \forall l_2, \forall l_3, (l_1+l_2)+l_3=l_1+(l_2+l_3)$ (1)
  $\forall l_1, l_1=l_1+0 \wedge 0+l_1=l_1$ (2)
  $\forall l_1, \exists l_2, l_1+l_2=0$ (3)
- Action :
  prouver
- Conclusions :
  $\forall l_1, \exists l_2, l_1+l_2=0 \wedge l_2+l_1=0$ (4)

La résolution du problème introduisait le sous-problème suivant, qui était lui-même emboité dans un sous-problème défini relativement au problème principal par introduction de x^d :

- Contexte (local) :
  le contexte général plus les éléments x, x^d et x^{dd}
- Hypothèses :
  les mêmes plus "x^d est l'inverse à droite de x" et "x^{dd} est l'inverse à droite de x^d"

- Action :
  prouver
- Conclusions :
  $x = x^d$.

Certains problèmes se résolvent intégralement par **transformation explicite de problème**. Le problème suivant est particulièrement exemplaire [Godement 66, p. 360] (cf Annexes) :

> Théorème 1. Soit K un anneau d'intégrité commutatif ; alors, pour tout entier n, l'anneau $K[X_1,...X_n]$ est intègre. En outre, quels que soient f, $g \in K[X_1,...X_n]$, on a
>
> $$d°(fg) = d°(f) + d°(g).$$

Sa résolution décompose littéralement la construction de $K[X_1,...X_n]$ jusqu'à obtenir deux éléments précisément choisis de K. Elle utilise des transformations de problème complexes qu'il serait intéressant de modéliser.

La transformation de problèmes est relativement bien étudiée car elle correspond à l'application de règles mathématiques que la logique a cherché à formaliser. Nous avons par exemple, pour démontrer A ⊢ B, la possibilité d'utiliser un raisonnement par réduction à l'absurde (A et non B ⊢ contradiction), un raisonnement indirect (A ⊢ non (non B)) ou un raisonnement par contraposition (non B ⊢ non A). Ces possibilités sont discutées par exemple dans [Cuppens 88b] ou [Solow 82].

Pour prolonger ces possibilités booléennes, il faut détailler la structure des opérandes, comme pour l'élimination de quantificateurs. Le mathématicien peut aussi être amené à combiner et à gérer plusieurs problèmes, comme pour le raisonnement par récurence.

En définitive, pour démontrer une égalité A=B le mathématicien a **le choix entre plusieurs schémas** de démonstration :

- par équivalence, comme une relation mathématique quelconque ;
- par un segment de preuve reliant les deux membres ;
- par double inégalité, ce qui est l'équivalent de deux problèmes ;
- par récurence, si la structure de l'égalité s'y prête ;
- par méta-propriété : un jeu d'écriture tel une symétrie permet de transformer A en B.

Le schéma de démonstration par récurence et le schéma par double inégalité contiennent par définition deux preuves de deux problèmes différents : par exemple $A \leq B$ et $B \leq A$. Pour les regrouper au sein d'une même preuve, une preuve à granularité variable comprend des opérations de déduction spécialisées dans les transformations de problème, et d'autres chargées de relier un problème à sa preuve. Il est possible à l'inverse de remplacer explicitement un segment de preuve par un sous-problème, lorsque ce sous-problème est résolu par transformation. Nous obtenons alors la grammaire suivante :

Problème-justifié	->	Problème-trivialement-résolu ou Problème-prouvé.
Problème-prouvé	->	Transformation-de-problème puis Problème-justifié ou Schéma-problème-preuve puis Preuve-justifiée.
Preuve-justifiée	->	suite de Transition-justifiée.
Transition-justifiée	->	Opération-mathématique ou Opération-de-déduction puis suite de Preuve-justifiée ou Schéma-preuve-problème puis Problème-justifié.

Un problème trivialement résolu est résolu par ailleurs, un théorème lui servant par exemple de justification. Pour aboutir à un problème mathématique prouvé, il y a finallement deux grands **modes complémentaires** de résolution d'un problème :

- résolution directe par construction de preuve ;
- résolution par transformation explicite du problème.

5.3.2. Les procédés de construction de preuve

Nous nous intéressons d'abord au processus de construction d'état avant de présenter des caractéristiques du processus de justification.

Chercher et construire une preuve détaillée nécessite le choix d'une structure de preuve et de son parcours. Le premier dilemne lors de la production d'une preuve est de permettre le **suivi des transformations** effectuées. Il y a pour cela deux procédés, utilisables simultanément :

① écrire cette transformation de façon à ce que l'emplacement, le point de départ, le résultat des règles et leur enchaînement éventuel soient évidents. Ce procédé se prête bien à l'utilisation de règles familières ;

② placer en commentaires de la transformation certaines des informations précédentes. Ce procédé se prête mieux à des règles d'application plus complexe, car il permet la desctription de développements éventuels.

L'inconvénient de la solution ① est que certaines étapes ne font intervenir qu'une petite partie de l'état et que la recopie du reste peut s'avérer encombrante. La rentabilité d'un palliatif dépend alors de la quantité de symboles inutiles et de la longueur des étapes où ils restent inactifs. Lorsqu'il n'y a pas de parallélisation possible d'opérations, une solution est de créer une preuve annexe qui concerne uniquement la partie à transformer.

Les inconvénients de la solution ② sont de briser la linéarité des déductions et de dupliquer certaines informations. Par exemple, décrire une règle et son instanciation revient à reproduire la partie de formule qui fait partie de la règle. Les mathématiques ont ainsi développé un usage permettant d'insérer des commentaires lapidaires pour expliciter leurs déductions par allusions, sans dupliquer inutilement ("par définition il vient", "en remplacant...", etc.).

Cette alternative entraîne par exemple, la présence ou l'absence d'une preuve annexe, et donc d'une structure de preuve plus complexe. Le parcours général peut alors être modifié par la mise en avant de la preuve annexe.

Un autre aspect de la construction de preuves consiste à pouvoir **mémoriser ou oublier certains aspects de la construction**. Nous avons vu, en effet, que certains points d'entrée intermédiaires à un segment de preuve sont heuristiquement importants. C'était le cas par exemple pour MA=MP ou pour la preuve de l'égalité de deux inverses dans un groupe (section 3). La présence d'un niveau de décomposition permet de matérialiser ce point d'entrée. Sa mise à plat permet "d'oublier" cette information.

Un cas identique se produit lors d'une stratégie que nous appelons *développement-identification*. Cette stratégie consiste à montrer une égalité connue en développant les deux membres (cf Annexes). Elle est utilisée par exemple pour montrer l'associativité d'une relation complexe. La preuve de A=B est alors réalisée en trois étapes :

- A=...=A1 (segment de preuve de A à A1) ;
- B=...=B1 ;
- preuve de A1=B1 (directe tel A1=...=B1, ou par identification de sous-blocs soigneusement choisis).

Cette construction est modélisée en reliant, au niveau R0, A=B à A1=B1. Les constructions A=...=A1 et B=...=B1 sont au niveau de décomposition R1, tandis que la preuve de A1=B1 est une justification en annexe de cet état.

preuve construite par développement-identification

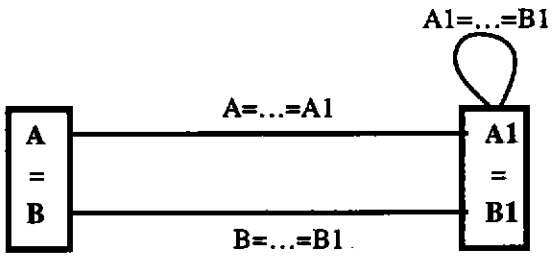


figure 1.25

Cette preuve à granularité variable permet un parcours restituant notamment la construction suivie par le mathématicien. Le fait que seul l'état A=B soit distingué à partir du problème privilégie ce parcours. Si l'énoncé précise "en utilisant le résultat intermédiaire A1=B1", les parcours peuvent être plus diversifiés.

Un parcours possible est d'ailleurs A=...=A1, A1=...=B1, B1=...=B. Ce parcours est équivalent au segment A=...=B, que le mathématicien aurait pu choisir s'il avait voulu masquer les aspects heuristiques de sa construction. Il aurait pour cela transformé la preuve en sa forme la plus simple, un seul segment sans A1=B1.

Une preuve à granularité variable permet ainsi de représenter quelques caractéristiques essentielles de la construction. Ces caractéristiques peuvent être oubliées si besoin par transformation de la structure de la preuve. Il est aussi possible de représenter d'autres caractéristiques telles le sens et le niveau du parcours suivi par des annotations.

Cette construction d'état serait néanmoins fastidieuse sans une certaine **souplesse du processus de justification.** Tout d'abord, les justifications autorisées en mathématiques permettent parfois de supprimer tout besoin de construction par transformation de problème. C'est le cas lorsqu'une transformation ramène un problème à un problème déjà résolu.

Plus fréquement les transformations de problème permettent de réduire les preuves à fournir. C'est le cas lorsque le mathématicien ramène "P(n)" à deux problèmes identiques d'ordre n/2, lorsqu'il n'étudie que le cas "x positif" pour une fonction paire, ou lorsqu'il conclut qu'il peut permuter a, b et c dans la solution car l'argumentation reste valide indépendamment de cette permutation.

Lorsqu'il faut néanmoins justifier la validité d'une preuve en construction, le mathématicien est expert en gestion des tâches. Nous avons vu qu'il sait repousser le calcul des bornes d'une intégrale transformée, qu'il omet les conditions de validité, qu'il sait laisser de coté des cas particuliers, etc. Il sait aussi, parfois, prendre les causalités à rebours en introduisant un ε tel que $P(\varepsilon)$, et en se permettant ultérieurement de restreindre ce choix en rajoutant $Q(\varepsilon)$.

Enfin, le mathématicien est passé maitre dans la simplification des constructions par l'amélioration du langage. Outre des choix de notation concis et simples à utiliser, le mathématicien étudie la continuité en 0 plutôt qu'en x_0 car cela simplifie les écritures, choisit a=1 sans perte de généralité, ou adopte une présentation graphique de la construction...

Le mathématicien dispose ainsi d'un arsenal de techniques destinées à faciliter la justification de la validité d'une preuve. Le mathématicien doit pouvoir les utiliser à bon escient pour faciliter sa construction de preuves. Il complète ensuite ultérieurement la justification des transitions lors d'une vérification et finition de la preuve.

5.3.3. L'exploitation des preuves construites

5.3.3.1. Aménager une preuve détaillée

Une fois construite une preuve détaillée, le mathématicien est amené à exploiter cette preuve. Nous distinguons principalement trois points :

- aménager et peaufiner une preuve détaillée ;
- construire un parcours sommaire à partir d'une preuve détaillée ;
- réutiliser une preuve pour d'autres problèmes.

L'aménagement d'une preuve débute durant sa construction. Quel qu'en soit le moment, des améliorations ponctuelles peuvent être apportées à une preuve. Nous avons vu par exemple que le mathématicien pouvait **"oublier" certaines caractéristiques de constuction** par des aménagements partiels. Par exemple, le mathématicien a eu recours à une stratégie de développement-identification faute de trouver facilement un chemin direct. Il répare ultérieurement cela par un aménagement séquentiel de la preuve. Certains aménagements sont toutefois pédagogiquement regrettables lorsqu'ils s'apparentent à un tour de prestidigitation.

Cet aménagement peut aussi concerner le **choix des étapes** d'une preuve. Les principes précédemment décrits, qui déterminent les étapes d'une preuve, peuvent aussi être utilisés pour les améliorer. Il est en effet plus facile de diagnostiquer a posteriori les améliorations possibles. Une étude systématique par thème est envisageable pour cela. Par exemple le thème "montrer au lecteur en quoi deux expressions sont comparables" se ramène souvent au principe ergonomique "éviter de dupliquer des expressions compliquées".

Cet aménagement peut aussi concerner le **contenu mathématique**. Dans les problèmes difficiles, la mathématicien n'a pas toujours, en effet, les moyens d'anticiper totalement une présentation finale "consacrée". Un cas typique est celui du découpage initial de l' "ε", afin d'aboutir à l' "ε" de la définition usuelle de la continuité. Cette rétropropagation de contraintes sur des termes dépendants est néanmoins plus utile lorsqu'elle s'applique pour limiter les conditions du choix antérieur de l' "ε".

L'aménagement peut porter enfin directement sur la **présentation textuelle** afin d'obtenir des formes de présentation plus condensées. Par exemple, dans le texte :

> on a la relation :
> $MA = MP$,
> or la figure donne :
> $MA = PB-PM' = p-\rho\cos(\pi-\theta)$,
> d'autre part, on a par définition :
> $MP = \rho$,
> d'où $\rho = p+\rho\cdot\cos\theta$.

MA = MP et MP = ρ peuvent être textuellement regroupés et leurs justifications "évidentes" supprimées. Moyennant quelques aménagements, nous pouvons ainsi retrouver la démonstration originelle de J. Quinet.

5.3.3.2. Construire un parcours sommaire

Construire un parcours sommaire à partir d'une preuve détaillée fait appel de façon plus poussée aux mécanismes d'aménagement précédents, selon les **intentions du mathématicien** dans son activité de démonstration. Elle introduit des critères d'importance et de difficulté. Les opérations importantes de la preuve concernent les principaux théorèmes ou propriétés utilisés, alors que le besoin et la façon de les utiliser viendra peut-être très naturellement.

Ce *critère d'importance* permet de déterminer de façon très grossière les résultats mathématiques externes utilisés pour bâtir cette preuve, tandis que le *critère de difficulté* est lié au contrôle de la recherche. La difficulté peut provenir d'une opération difficile à trouver comme d'une succession d'opérations simples à guider dans un espace de recherche combinatoire. Dans le premier cas l'opération est aussi présente dans le parcours sommaire, alors que des indications de guidage sont fournies dans le second cas.

L'argumentation et les faits mentionnés varient selon le parcours choisi. Un parcours de la preuve d'une égalité ensembliste peut indiquer seulement que la preuve s'effectue par inclusion mutuelle. Pour plus de détail, dans l'hypothèse où l'une des branches est difficile, la preuve en décrit les grandes lignes alors que l'autre branche est indiquée comme triviale.

Il semble actuellement difficile d'expliciter les lois qui gouvernent la production d'un parcours sommaire à partir d'un parcours détaillée d'une preuve. Un parcours sommaire repose essentiellement sur des arguments explicatifs dont l'intérêt consiste selon Pólya à :

> présenter juste le germe de la preuve, l'idée directrice sous sa forme la plus simple, et indiquer la nature des détails restants.

Il est possible de proposer des critères d'importance basés, par exemple, sur ce qu'il faudrait indiquer à une machine reproduisant un raisonnement humain ; de même un critère de difficulté peut être défini comme l'ordre de grandeur des possibilités à analyser, vu le guidage heuristique. Toutefois, le développement et l'affinement de ces critères ne se fera, ici encore, que conjointement à une étude expérimentale et dans le but de lui servir concrètement.

Une autre difficulté importante de la production d'un parcours sommaire est le passage à des représentations plus intuitives afin d'obtenir une justification synthétique évocatrice pour le mathématicien.

5.3.3.3. Réutiliser une preuve pour d'autres problèmes

Nous avons jusqu'à présent présenté la (ré-)utilisation d'une preuve d'un problème. La réutilisation d'une preuve concerne aussi d'autres problèmes. Le cas extrême est celui de résultats mathématiques généraux, qui sont réutilisés pour d'autres problèmes justement parce qu'il existe une preuve de leur validité. Nous dirons alors que leur preuve est externe au problème à traiter.

Par construction, l'utilisation de résultats généraux nécessite une spécialisation de leur **contexte de définition**. La preuve qui les justifie est donc plus générale que celle dont le mathématicien a besoin pour son problème. Nous présentons en Annexes un résultat obtenu dans des contextes de définition différents (il s'agit de la somme des n premiers nombres entiers).

Il y a alors plusieurs aspects de la réutilisation d'une preuve d'un problème. Il peut s'agir de :
- utiliser un résultat général, en exploitant ainsi implicitement sa preuve ;
- généraliser un problème et sa preuve afin de disposer d'un résultat plus général ;
- réutiliser une preuve d'un problème dans un problème proche ;

Utiliser un résultat général ne nous concerne pas ici. Le point important est plutôt quand **décider d'établir un résultat général**. Lors de la construction d'une preuve, le mathématicien peut être amené à s'interroger sur la pertinence de justifier un résultat par une preuve annexe (intégrée dans la preuve à granularité variable) ou s'il doit donner un statut plus général à ce résultat en construisant une preuve externe. Il crée en général une preuve externe lorsque ce résultat lui sera utile par ailleurs. Ce problème est comparable à celui de la définition de code local et global dans les logiciels.

Généraliser la preuve d'un problème est fort complexe, mais fréquent en mathématiques. Cela nécessite au minimum de garder la trace de l'usage des données à généraliser, afin de connaître leur influence sur le résultat obtenu et savoir quoi conserver. Par exemple, [Graham 89, p33] écrit en substance, pour la preuve de $S_n = \sum_{0 \le k \le n} k\,2^k$:

> Donc nous avons $S_n + (n+1)\,2^{n+1} = 2S_n + 2^{n+2} - 2$, et le calcul donne $S_n = \ldots$
>
> Une preuve similaire avec x à la place de 2 nous aurait donné l'équation :
> $S_n + (n+1)\,x^{n+1} = xS_n + (x - x^{n+2}) / (1 - x)$; d'où nous pouvons déduire que $S_n = \ldots$

Les résultats respectifs ont une allure très différente. Le mathématicien sait trouver ici l'état critique où la preuve diverge selon les données. Il sait aussi revenir sur une simplification explicitée à un niveau de détail inférieur : $(2 - 2^{n+2}) / (1 - 2) = 2^{n+2} - 2$. La provenance mathématique des termes est ainsi capitale puisque le résultat n'est pas directement réutilisable ici.

La réutilisation de S_n, par **morceaux de preuve paramétrés**, ne s'applique que pour des états syntaxiquement similaires. Au dela, les opérations utilisées pour les transitions diffèrent, et la structure de la preuve diverge selon les données. Cela implique de raisonner en terme de tâches mathématiques à effectuer pour obtenir plus de souplesse dans la réutilisation. Il s'agit alors de réutiliser non plus directement la preuve, mais les **objectifs du mathématicien dans le développement de cette preuve**. C'est une forme affaiblie, mais objectivable, de capture de l'idée directrice du mathématicien.

L'apprentissage symbolique automatique permet actuellement de généraliser des morceaux de preuves dans des cas favorables. Vouloir faire mieux nécessite de capturer la construction de la preuve et les objectifs correspondants du mathématicien. La structure de preuve à granularité variable est une étape prometteuse pour cela. Des transformations peuvent être étudiés et expérimentés sur les preuves à granularité variable car elles reproduisent mieux les raisonnements des mathématiciens.

La généralisation de la preuve d'un problème est un cas particulier de la réutilisation d'une preuve pour un problème proche. Les développements précédents s'appliquent aussi dans ce cadre. De plus une preuve présente l'avantage d'être assez insensible à certaines modifications de la formulation du problème. G.Pólya l'illustre par exemple pour un cas de preuve par raisonnement indirect et un cas de raisonnement par l'absurde [Pólya 57 p163-171]. Il transforme a posteriori chacun des deux problèmes de façon à permettre un raisonnement direct, et utilise pour cela l'essentiel de la preuve initiale.

Il existe fréquemment plusieurs argumentations possibles en mathématiques. [Graham 89, p33] donne aussi une argumentation pour S_n radicalement différente de la première (en disant que $k\ x^{k-1}$ est la forme dérivée de x^k). La principale limite à la généralisation d'une preuve est alors que **l'argumentation choisie n'est plus valable dans un cadre différent.**

Un exemple illustre bien cela : soit un ensemble A inclus dans B, alors $A \cap (A \cup B) = A$ car en remplacant $A \cup B$ par B (ce qui est vrai par l'inclusion) on obtient l'équivalence avec $A \cap B = A$ (qui est aussi vraie par l'inclusion). Cette preuve est inutilisable lorsque A n'est pas inclus dans B. Pourtant, ce résultat est vrai pour deux ensembles quelconques. L'argumentation utilisée localement est donc totalement à revoir.

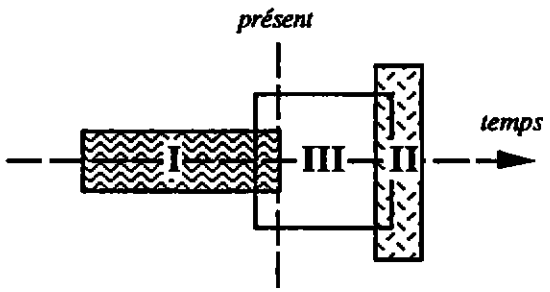
La réutilisation d'une preuve est alors conditionnée par :

- son contexte de définition ;
- une bonne anticipation du mathématicien dans le choix de son argumentation ;
- la faculté de déterminer les aspects de la preuve qui restent valables et ceux qu'il faut modifier ;
- la capacité de modifier une preuve, non seulement dans sa structure syntaxique, mais en allant jusqu'à remettre en cause son processus de développement.

Nous disposons maintenant d'un modèle de preuve suffisamment adapté et adaptatif pour représenter les preuves produites par le mathématicien. Ce modèle de preuve à granularité variable fournit un canevas basé sur des transitions utilisées par le mathématicien. Il est issu de l'étude de démonstrations mathématiques et ne se conçoit que dans la poursuite de cette expérimentation. Cette modélisation est notamment précisée et validée sur des exemples de construction de preuve en partie II.

PARTIE II

Les Mathématiques Assistées



1. L'ASSISTANCE AU MATHEMATICIEN

1.1. LE MATHEMATICIEN FACE A L'INFORMATIQUE

Le premier problème que doit affronter un mathématicien face à un système informatique est celui des formules ! Rien n'est plus dur en effet que de reconstituer la disposition bidimensionnelle de ces symboles hétéroclites. Si la décennie 80 a diffusé des systèmes spécialisés disposant d'un affichage agréable, le mathématicien se bat encore souvent avec un langage informatique peu pratique pour construire et manipuler ses formules. **L'informatique devient pourtant quasiment obligatoire,** ne serait-ce que pour la diffusion de travaux mathématiques. Le pas du mathématicien vers l'informatique n'est toutefois franchi que pour des besoins impérieux, souvent limités au travail de secrétariat final. C'est pourquoi actuellement, l'essentiel du travail du mathématicien reste dévolu au papier-crayon.

Pourtant, des facilités de calcul formel seraient bien utiles aux mathématiciens, mais la seule démarche pour les utiliser est souvent plus longue que la résolution manuelle. Nombre de calculs restent alors effectués manuellement, car seule une utilisation assidue justifie l'usage d'un système de calcul formel. Une telle utilisation ne se justifie d'ailleurs que pour des domaines spécialisés dans lesquels d'autres mathématiciens ont exhibé des méthodes analytiques générales. Comme d'aucuns disposent d'un FORmula TRANslator, les usagers des systèmes de calcul formel bénéficient donc d'un langage de programmation convenable pour traiter ces classes de problèmes. Ils obtiennent ainsi les résultats qu'ils recherchent, malgré une formulation ad hoc, une justification cachée et des explications inexistantes. Pour résumer la situation, **les mathématiciens sont démunis** lorsque l'essentiel du problème consiste à savoir l'aborder ou le résoudre, ou lorsqu'ils ont besoin d'une démonstration plus explicite qu'une trace d'exécution.

Beaucoup de mathématiciens ont ainsi classé les systèmes informatiques disponibles, les obstacles prenant le pas sur l'intérêt de leurs fonctionnalités. Pourtant l'informatique est d'une utilité permanente dès que des objets peuvent être réutilisés, à condition de pouvoir décrire comment les relier. Les mathématiques et plus particulièrement les manipulations de formules offrent une réutilisation idéale du fait de la cohérence mathématique. Les mathématiques réutilisent non seulement les méthodes et programmes comme en informatique, mais surtout les formules et autres données intermédiaires produites au cours de la preuve. C'est par ce biais que nous comptons **amener les mathématiciens à profiter de l'informatique** : nous proposons d'intégrer manipulations textuelles et mathématiques au sein d'un processus de construction de preuves et démonstrations.

Les champs d'application de l'informatique ont dépassé le domaine du calcul et de la gestion de données : les générations informatiques récentes manipulent des **données symboliques complexes.** Parallèlement, une véritable **communication** s'est établie avec l'utilisateur, lui permettant de s'abstraire des considérations informatiques qui le détournent de son problème.

Pour un usage mathématique, cela se traduit par des applications autres que calculatoires, comme la résolution de problèmes mathématiques, la manipulation de formules et le suivi d'une démonstration. Une grande disparité règne toutefois : elle résulte principalement d'une absence d'approche globale unifiante. De fait, les outils informatiques existants présentent souvent plusieurs inconvénients majeurs :

- Leur utilisation nécessite un **savoir-faire spécifique** important, pour un champ d'application mathématique très limité.
- Il n'y a pas de **compréhension mathématique** des tâches effectuées pouvant, par exemple, amener une justification. Les opérations effectuées ont un caractère plus informatique que mathématique (par ex. entrer les données d'un programme pour résoudre une équation).
- L'**utilisation des connaissances** d'ordre mathématique n'est pas explicite. Les outils sont cloisonnés, difficilement extensibles, et ne sont pas conçus pour communiquer avec d'autres outils.
- La boucle d'interaction ne bénéficie pas de la souplesse d'expression et de la lisibilité du **Langage Mathématique.**

1.2. INTRODUIRE UN ASSISTANT MATHEMATICIEN INTELLIGENT

Notre approche des Mathématiques Assistées par Ordinateur consiste à dire :

> Un usager mathématicien dispose d'un assistant capable d'effectuer des tâches élémentaires, fastidieuses ou mécaniques : **comment va-t-il l'employer** ?

La réponse est simple, il va d'abord jauger ses capacités, voir quelles sont les tâches qui relèvent de sa compétence et comment employer au mieux cette aide bénévole ! L'usager va alors expérimenter le travail en commun dans le but inavouable, mais O combien agréable, d'exploiter au maximum cet assistant pour s'intéresser à des choses plus passionnantes encore...

Dans l'état actuel, le désenchantement ne se fait guère attendre ! Le plaisir de disposer d'une telle aide s'estompe lorsque les contraintes deviennent trop présentes. L'usage d'un système fatigue, agace ou ennuie : le mathématicien parle avec passion de ses résultats, pas de l'assistant utilisé. De plus, l'assistant n'est utile que pour des tâches trop ponctuelles. Finalement le mathématicien a l'impression d'être au service de la machine.

Le mathématicien souhaiterait un assistant convivial et suffisamment intelligent pour faciliter son activité mathématique. Il veut un :

Assistant mathématicien intelligent (Ami)

Nous cherchons à réaliser ce rêve initial pour procurer à l'usager des satisfactions et des possibilités insoupçonnées sans Ami. Pour cela nos questions sont :

① Quelles sont les tâches que l'Ami est à même d'effectuer ?
② Comment les intégrer à l'activité du mathématicien sans en faire des tâches séparées ?
③ Comment vont-elle être commandées et contrôlées par le mathématicien ?
④ Quelle est l'influence d'un Ami sur la pratique de l'activité mathématique ?

Nous nous proposons de répondre d'abord aux deux premières questions.

1.3. NOTRE CHOIX DE CONSTRUCTION DE PREUVES ET DEMONSTRATIONS

Communiquer avec la machine est l'une des premières exigences de l'usager. Malheureusement, le mathématicien ne dispose actuellement que d'un dictionnaire de fonctions. Le travail du mathématicien se rapproche de celui de l'écrivain qui fait vivre les mots et les idées, les assemble, les guide vers des recoins inexplorés. Nous voulons lui fournir les moyens d'utiliser son langage et les concepts mathématiques qu'il manipule. Notre but n'est pas de nous substituer à lui, mais d'**assister le mathématicien dans ses tâches élémentaires.**

Pour demander plus que ce dont on dispose actuellement, pour avoir un véritable Ami, il faut, comme dans les autres domaines scientifiques abordés par les systèmes experts, **expliciter, rassembler, classifier et représenter les connaissances.** Mais cette étape est trop importante pour la laisser aux uniques soins des informaticiens, ou la garder dépendante d'une application. Elle doit résulter d'une action concertée entre mathématiciens et informaticiens, dans le but d'établir des bases de connaissances regroupant les divers aspects des Mathématiques et de leur langage. Ceci nous a conduit à étudier le Langage Mathématique et les multiples connaissances qu'il fait intervenir.

Le Langage Mathématique constitue le premier maillon de la spécification d'un problème. Il est souple, clair, concis, et correspond au mode de pensée du mathématicien. Les autres formalismes nécessitent un effort de traduction et de compréhension sans même faciliter l'activité mathématique. Nous nous proposons d'exploiter la fonction de communication du Langage Mathématique pour l'acquisition et la modélisation des connaissances. La fonction mathématique du langage nous servira à utiliser "judicieusement" ces connaissances.

Si la linguistique des mathématiques et les sciences cognitives sont prêtes à bénéficier d'un modèle informatique de l'activité mathématique, ces préoccupations n'intéressent guère les mathématiciens ! Et pourtant, nonobstant l'apport d'une telle étude à la connaissance des mathématiques, il est nécessaire de débuter par un objectif profitable aux deux communautés pour réaliser le projet. Le meilleur choix pour cette étude pratique de l'activité mathématique nous semble être :

l'assistance à la construction de preuves et démonstrations.

En effet, la présentation "naturelle" de preuves à l'aide de démonstrations intéresse les linguistes et les mathématiciens. Plus généralement, les procédés de construction de preuve (et l'historique de leur construction) concernent deux thèmes majeurs des sciences cognitives : connaissances et raisonnement. Avec cette tâche d'assistance, les diverses communautés sont amenées à collaborer pour définir les structures et les procédés de construction des preuves et démonstrations, ainsi que pour exploiter leur modélisation informatique. De plus cet objectif empêche une schématisation du langage manipulé, puisque le produit final doit présenter la qualité attendue pour une communication interpersonnelle.

Le transfert du cycle de conception du mathématicien vers la machine n'a pas pour seul but de produire et de vérifier une preuve. Son objectif principal est d'y rattacher les systèmes à vocation mathématique et paramathématique afin d'exploiter des preuves sous tous leurs aspects. Les données mathématiques étant fortement corrélées, leur réutilisation par un système d'assistance se fera d'autant mieux que **le cycle de conception sera étendu.**

Par exemple, produire la transformation d'une formule compliquée par l'application d'un théorème nécessite au minimum la saisie de la formule et des indications sur le théorème :

- Si l'objectif du mathématicien est uniquement textuel, il l'atteindra par des manipulations syntaxiques en effectuant indépendamment le travail déductif.
- S'il cherche de plus à construire la preuve, il n'aura qu'à suggérer d'appliquer le théorème pour se voir dispenser des détails de la réalisation et de sa validation : les contorsions textuelles sont alors superflues !
- Si la transformation est plus complexe, il faut détailler la preuve qui pourra être utilisée dans plusieurs présentations textuelles voire réutilisée pour des problème voisins... Le problème majeur consiste alors à disposer d'une description des données à un niveau conceptuel suffisant pour que les similarités existantes soient exprimables[1].

[1] Quelques exemples rassemblés partie III -1.3.2 lors de l'état des lieux des systèmes de traitement de texte montrent que ce niveau conceptuel atteint fréquemment la gestion des tâches de production de la preuve... Un Ami doit alors disposer d'informations à ce niveau de description, quitte à devoir les synthétiser par observation

1.4. LE LANGAGE ET LES CONNAISSANCES

Fournir une assistance opérationnelle bute sur les facteurs les plus fondamentaux du cycle de conception, à savoir le langage et les connaissances disponibles.

Les principaux facteurs d'assistance à la construction de preuves et démonstrations sont les connaissances qui permettent :

- la mise à disposition d'opérations naturelles pour le mathématicien ;
- une assistance ergonomique dédiée aux opérations mathématiques ;
- la production et l'utilisation de moyens de construction.

La production et l'utilisation de moyens de construction (plans, stratégie de guidage, choix des étapes de preuve, etc.) ont été examinées en partie I lors de l'étude des preuves et démonstrations. Nous nous intéressons dans cette partie aux deux premiers points qui concernent les aspects les plus élémentaires d'un système informatique : les opérations disponibles.

L'assistance qui nous intéresse ici concerne donc l'**usage du langage pour les manipulations élémentaires** (dans leurs composantes "Phrases" et "Formules"). Elle vise à la fois :

l'interprétation (analyse et représentation) et l'utilisation (raisonnements et génération) du langage .

Une partie substantielle des informations présentes dans le Langage Mathématique est implicitement destinée à être reconstituée par le mathématicien. Lever les ambiguïtés tire ainsi profit des importantes capacités d'analyse et de raisonnement humain, ainsi que d'un savoir-faire très diversifié.

L'ordinateur, déjà handicapé par son mode de fonctionnement, est voué à l'échec sans connaissances spécifiques à la compréhension du langage. Il devra, à partir de la séquence de bits entrée, restituer la syntaxe et la sémantique, c'est à dire reconstituer le message. Sous cet angle très général, sa tâche est trop ardue et doit être assouplie. Un choix raisonnable consiste à réduire la partie implicite par des déclarations, profiter de l'assistance du mathématicien, et autoriser les échecs ou les erreurs d'analyse.

Une activité mathématique assistée par ordinateur consiste alors en deux étapes globalement distinctes, présentant une rétroaction éventuelle :

① **édifier un modèle** pour représenter les problèmes et introduire les connaissances disponibles
② **utiliser ce modèle** dans les activités correspondant à la résolution de problème.

L'étape ① correspond aux informations mathématiques et paramathématiques qui concernent une théorie. L'objet de cette section est la nature de l'assistance proposée pour l'étape ② d'utilisation du modèle. L'état des lieux de la construction de preuves et démonstration étant quasiment inexistant, nous cherchons à concevoir les ambitions et les limites d'un système réaliste.

2. ÉTUDE ET MODELISATION DE PREUVES

2.1. LA CONSTRUCTION D'UNE PREUVE A PARTIR D'UNE DEMONSTRATION

Notre objectif est de produire sur un support informatique des démonstrations dont la preuve soit manipulable et correcte. La construction d'une preuve à partir d'une démonstration est un processus actif. Le système cognitif qui l'effectue ne se borne pas à noter les états et les transitions de la preuve, mais complète voire vérifie les informations produites. Le système cognitif qui interprète (correctement) une démonstration construit donc une preuve plus détaillée que celle qui était contenue dans la démonstration énoncée[2].

L'intérêt d'une démonstration est justement de ne fournir que l'information pertinente à l'élaboration de cette preuve. Cette information pertinente est de plus à reconstituer à partir de la présentation textuelle de la dite démonstration. La robustesse de ce processus de reconstitution vient de la souplesse d'adaptation de l'homme, de la "codification" du langage, et de la possibilité d'interagir pour éclaircir certaines étapes.

Malheureusement, le niveau compréhensible par un assistant informatique Ami est largement au dessous des démonstrations usuelles et correspond habituellement à des preuves très détaillées. De plus, les difficultés rajoutées par une présentation textuelle sont importantes. C'est pourquoi, nous proposons de travailler sur la représentation visuelle de la preuve ou d'adopter des présentations textuelles permettant de reconstituer facilement des preuves détaillées. Les deux cas de construction : mathématicien-mathématicien et mathématicien-Ami sont présentés figure 2.1.

Construction d'une preuve à partir d'une démonstration

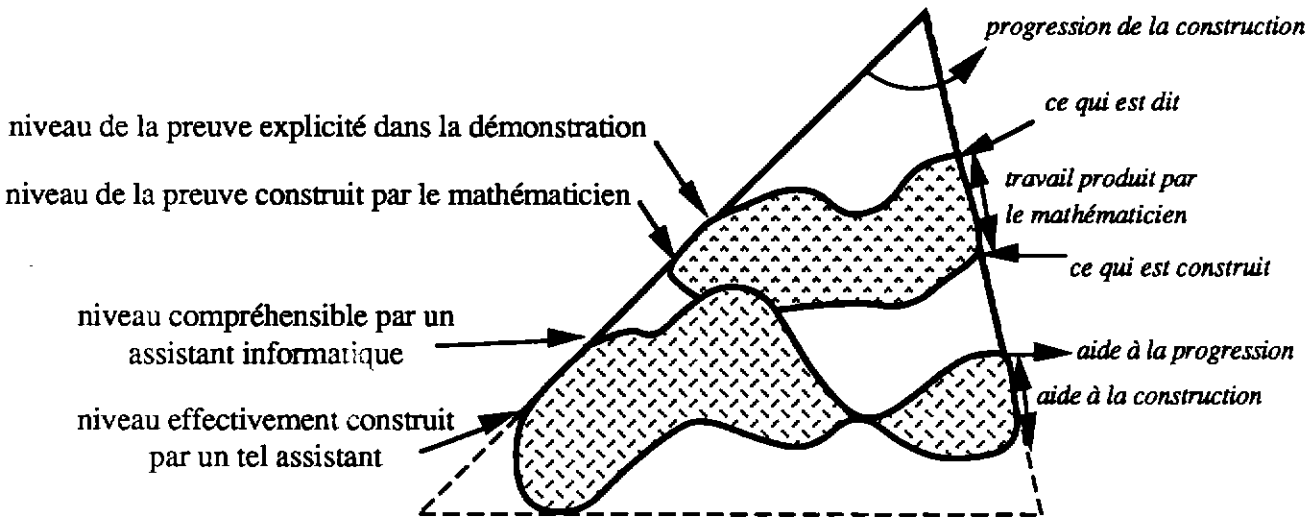


figure 2.1

Pour exploiter activement une preuve, le niveau du discours du mathématicien doit se contraindre à arriver jusqu'au niveau compréhensible par son assistant informatique. Il peut alors bénéficier à ce niveau de l'assistance dans ses manipulations voire l'automatisation de la construction de la preuve par des outils de calcul ou de raisonnement. Il obtient ainsi deux formes d'assistance :

- une **aide à la progression**, qui facilite l'opération de construction en cours et anticipe les suivantes ;
- une **aide à la construction**, qui déduit des informations supplémentaires nécessaires à la construction de la preuve.

---

[2] Ce procédé autorise en particulier l'ambiguïté intrinsèque dans une preuve, puisque certains passages non explicités peuvent être interprétés de plusieurs façons. Cet effet est voulu car son concepteur estime que ce passage existe mais que la ou les manières de le concrétiser ne sont pas intéressantes pour l'exposé global de la solution. Mal maitrisé, cet effet ouvre la voie aux preuves incorrectes... mais reste un outil indispensable pour la recherche et la mise au point de preuves valides.

Le mathématicien retrouve alors à ce niveau la possibilité de ne fournir que l'information pertinente à l'élaboration de la preuve. Vouloir bâtir la preuve dans ses moindres détails revient en effet à introduire des informations redondantes (appliquer tel théorème dans tel sens avec telles substitutions pour obtenir tel résultat grâce à telle simplification élémentaire...). La connaissances de ces opérations élémentaires permet de minimiser l'information à fournir tout en garantissant la construction effective.

Un processus interactif de formulation, recherche et présentation devient alors le moteur de la construction d'une preuve très détaillée. Un langage de saisie plus impératif et moins descriptif se substitue à celui des démonstrations usuelles, en se rapprochant d'un langage de manipulation de formules.

Mais en quoi ce détour jusqu'à une preuve très détaillée serait-t-il bénéfique au mathématicien ? C'est à notre avis le seul possible informatiquement. Notre objectif immédiat est de fournir les moyens de représenter des preuves à granularité variable et de leur associer une démonstration textuelle. Nous escomptons à terme savoir générer un parcours sommaire à partir d'un parcours détaillé. Nous escomptons aussi construire simultanément la démonstration en même temps que la preuve.

Il faut remarquer de plus qu'il n'existe pas actuellement de projet informatique visant des démonstrations dont la preuve soit manipulable et correcte. Les quelques travaux qui s'intéressent aux démonstrations n'en abordent que partiellement certains aspects textuels. Parallèlement ceux qui s'occupent de preuves s'intéressent plus aux aspects logiques ou à la trace d'algorithmes de calcul. Dans tous les cas, l'essentiel de l'activité observable du mathématicien est mis de côté au profit d'une tâche particulière. Il s'ensuit que le **cycle de conception** de la résolution de problèmes mathématiques jusqu'aux démonstrations n'est descriptible qu'en terme de liens très indirects.

L'avantage de notre approche est de chercher à intégrer - même si ce n'est qu'à bas niveau - les liens et causalités de ce cycle de conception, au lieu de construire séparément les parties désirées avec une aide minime. Cela signifie qu'il faut expliciter plus de choses avant d'aboutir aux parties émergées de l'édifice, mais que cela se fait dans une plus grande fluidité d'interaction. Les avantages de notre approche sont de :

- pouvoir faire plus de traitements sur les objets manipulés ;
- expliciter les connaissances et comprendre leurs mécanismes d'utilisation ;
- fournir une base commune au développement d'applications mathématiques ;
- permettre, à terme, le recueil de données et des expérimentations cognitives sur l'activité du mathématicien.

2.2. ÉTUDE DE LA CONSTRUCTION DE PREUVES ET DEMONSTRATIONS

Nous analysons maintenant quelques démonstrations mathématiques dans l'optique de leur construction par un système d'assistance. Nous abordons respectivement :

- une analyse détaillée de leurs caractéristiques mathématiques, permettant une modélisation fine de la preuve et de sa construction par le mathématicien ;
- les actions d'assistance possibles et les connaissances mathématiques nécessaires à cette construction ;
- les moyens d'assistance et le guidage proposé lors de construction de ces preuves mathématiques.

Nos exemples ont pour thème la sommation formelle Σ. Nous étudions d'abord la démonstration du calcul de la somme des n premiers nombres entiers[3], dont voici l'exemple historique :

Raisonnement suivi par Gauss enfant pour calculer la somme des n premiers entiers

$$\begin{array}{rccccccc} & 1 & + & 2 & + & \ldots & + & n \\ + & n & + & (n-1) & + & \ldots & + & 1 \\ \hline & (n+1) & + & (n+1) & + & \ldots & + & (n+1) \end{array}$$

D'où $2S = n\,(n+1)$

[3] Différentes démonstrations de ce résultat, issues de traités mathématiques, sont disponibles en Annexes.

Notre premier exemple reproduit le schéma attribué à Gauss enfant en utilisant des sommations formelles. Nous avons choisi tout au long de notre analyse une présentation spécifique des preuves, permettant de détailler clairement les différentes composantes. La présentation regroupe les productions mathématiques à gauche, avec à droite un commentaire sur les opérations effectuées. Les formules présentées à gauche correspondent aux états du parcours de la preuve. Une démonstration désigne dans ce qui suit cette forme de présentation textuelle.

Démonstration inspirée de Gauss enfant

Calculer la somme des n premiers entiers — *le problème à formaliser*

$$\sum_{j=0}^{n} j$$

$$= \sum_{j=0}^{n} (n-j)$$ — changement de variable j —> n-j

Or — *rupture d'enchaînement*

$$\sum_{j=0}^{n} j + \sum_{j=0}^{n} (n-j)$$

$$= \sum_{j=0}^{n} n$$ — regroupement de Σ

$$= n \sum_{j=0}^{n} 1$$ — mise en facteur de n

$$= n\,(n+1)$$ — définition du Σ

D'où — *production d'une "déduction logique"*

$$\sum_{j=0}^{n} j = \frac{n\,(n+1)}{2}$$ — *cqfd.*

Les preuves détaillées se prêtent bien à ce type de présentation visant à séparer les calculs d'un texte traduisant le raisonnement. Le besoin de présentations alternatives est effectif puisque nous avons une illustration datant de 1875 [Whitworth 1875]. Elle est reproduite en Annexes, accompagnée d'une autre présentation récente destinée à détailler des résultats de théorie des nombres[4].

Posons $A=\sum_{j=0}^{n} j$, $B=\sum_{j=0}^{n} (n-j)$ et $C= n(n+1)$.

Nous adoptons la preuve suivante pour la démonstration inspirée de Gauss enfant :

[4] Cette présentation a été trouvée indépendamment par Paul Depasse pour les besoins d'un article. La recrudescence de travaux en théorie des nombres pour les applications informatiques ouvre un champ d'application prometteur pour des systèmes d'assistance au mathématicien (construction et validation de preuves).

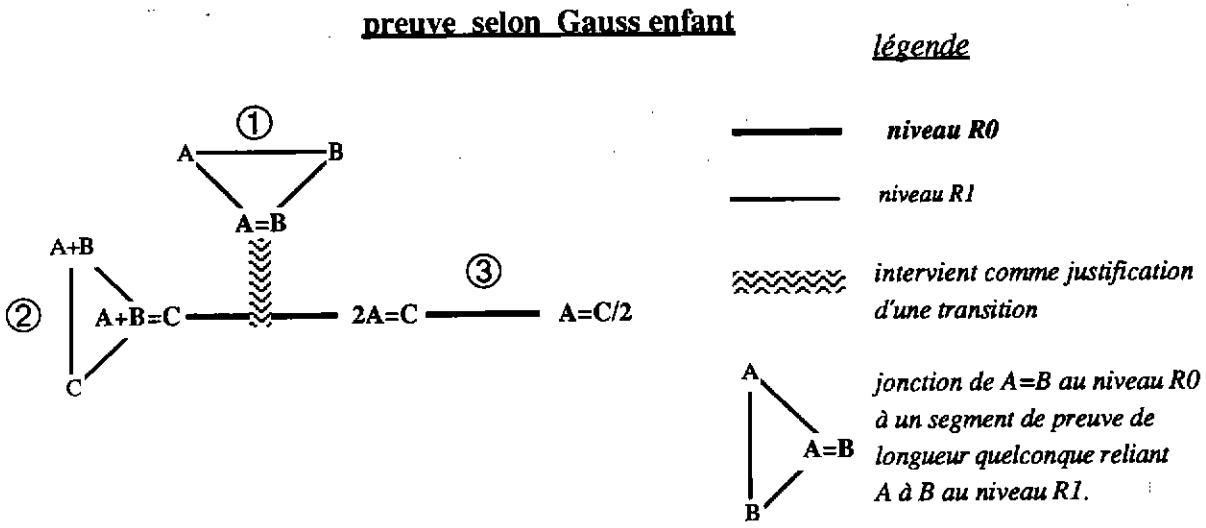


figure 2.2

L'ordonnancement de la déduction consiste à déterminer les causalités entre constituants. Le mathématicien construit d'abord le segment de preuve ①. Il récupère les extrémités A et B pour bâtir un autre segment disjoint ②. Il effectue alors une synthèse de ces résultats pour bâtir le segment ③. Ce segment constitue le niveau de référence R0 car il regroupe des fragments des autres segments situés au niveau inférieur R1.

L'**opération de déduction** s'effectue ici en trois temps. Elle est composée de :
- une jonction du niveau R1 vers le niveau R0 du segment ① ;
- une jonction du niveau R1 vers le niveau R0 du segment ② ;
- une synthèse qui utilise les deux états produits au niveau R0 (A=B et A+B=C) pour produire 2A=C (nous omettons ici la nuance A+A=C).

L'argumentation mathématique est très brièvement explicitée dans la démonstration par les marqueurs "Or" et "D'où", indiquant respectivement une rupture d'enchaînement et une synthèse aboutissant à la production d'une "déduction logique". La synthèse consiste à organiser deux sources d'information en effectuant éventuellement un changement de niveau. Cette notion de triangle de jonction du niveau R1 au niveau R0 est très utilisée en mathématiques mais n'avait pas été décrite à notre connaissance.

L'analyse de preuves peut être décomposée en trois thèmes :
- les actions externes à la preuve réalisées par le mathématicien ;
- l'argumentation mathématique utilisée pour les opérations de déduction ;
- les opérations mathématiques utilisées dans les transformations de formules ;

Le premier thème étant principalement traité en partie I, nous abordons respectivement les deux autres.

2.3. LES CAUSALITES QUI REGISSENT LES OPERATIONS DE DEDUCTION

Les causalités qui régissent les opérations de déduction méritent une attention particulière afin de déterminer celles qui sont accessibles à partir des données. Ces dernières peuvent en effet être prises en compte pour faciliter la construction.

Effectuer une opération de déduction est considéré comme élémentaire par le mathématicien. La difficulté principale consiste à savoir l'anticiper (aspect heuristique), une difficulté secondaire est de reconnaître l'opération employée. La surcharge d'utilisation des marqueurs langagiers "Or" et "D'où" cache une grande diversité d'opérations de déduction possibles. Nous avons par exemple :

A=B or A=C d'où B=C

A+B=C or A-B=D d'où 2A=C+D

Dans notre exemple A=B or A+B=C d'où 2A=C, le mathématicien pourrait aussi bien conclure 2B=C. Il ne le fait pas car certains termes sont considérés comme "inconnu, à déterminer" et d'autres comme "inconnu, à éliminer". De plus, les termes sont ordonnés selon leur intérêt : C est plus intéressant que B, ce qui fait que le mathématicien préfère une relation entre A et C à la relation entre A et B. Ces critères doivent être connus d'un système d'assistance pour **anticiper la synthèse** escomptée de l'opération de déduction.

Le facteur heuristique le plus difficile à prévoir est celui de la **présence d'une opération de déduction**. Pour bien le comprendre, il faut savoir pourquoi le mathématicien a besoin d'introduire une opération de déduction. Avant tout, rappelons que l'initiative appartient prioritairement au mathématicien. Prenons pour cela l'exemple suivant :

segment 2A à C

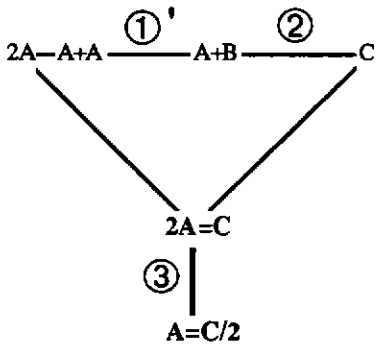


figure 2.3

Le mathématicien débute par 2A de façon à pouvoir enchaîner sans rupture en A+B. Cette solution est heuristiquement artificielle, puique la seule explication du point de démarrage 2A est d'anticiper sur la présence et la simplification de A+B. Lorsqu'il maitrise ainsi les difficultés heuristiques, le mathématicien peut se permettre une résolution directe : A=2A/2=(A+B)/2=C/2.

Nous en concluons qu'une opération de déduction évite une anticipation heuristique trop importante et permet des preuves plus condensées (segment de A à B au lieu de A+A à A+B). Cet exemple serait néanmoins heuristiquement approprié si C était un point de départ "logique" et que 2A en était l'aboutissement. La présence ou l'absence d'une opération de déduction traduit donc des causalités liées aux données du problème. Cela entraîne des preuve de structure totalement différentes.

Le problème est ici de trouver la forme exacte du terme initial A. Le mathématicien peut le faire directement par transformations à partir de A. Dans les cas plus complexes, il cherche une relation sur A qu'il résoud ultérieurement. La façon d'obtenir cette relation sur A détermine la structure de la preuve.

Le suivi et la production de cette preuve est envisageable par un assistant Ami moyennant des connaissances spécifiques aux preuves et aux opérations de déduction. La première action d'un Ami est de diagnostiquer que cette preuve permet potentiellement de trouver A (et donc de résoudre le problème), mais qu'elle ne débute pas par A : il s'agit forcément d'un segment annexe au niveau R1. L'emplacement de la synthèse à effectuer dépend alors du développement de ce segment. Lorsqu'il aboutit à un facteur constant C comme ici, ce facteur constant est simplifié ici sous peine de reporter la simplification ultérieurement. L'emplacement de la synthèse à effectuer dépend donc du développement de ce segment.

Cette présentation des opérations de déduction à l'aide de termes abstraits A, B et C ne rend pas suffisamment compte des **causalités de la construction de preuve**. Nous retrouvons cela dans l'exemple ci-dessous contenant le segment A à C-A.

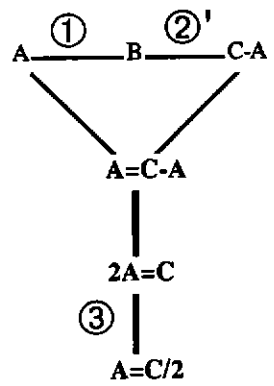


figure 2.4

Par rapport à l'exemple précédent le mathématicien découvre ici une relation permettant de déterminer A, au lieu d'aboutir à un facteur constant. Il s'arrête ici immédiatement pour effectuer une synthèse, au lieu de chercher à simplifier C. La simplification de C aura lieu ici en annexe de la transition 2A=C à A=C/2 du segment ③. Le terme abstrait C est donc simplifié ou non avant la synthèse, selon la preuve considérée. On retrouve ici que l'emplacement de la synthèse à effectuer dépend du développement de ce segment ①.

Une telle opération de déduction est courante car le procédé mathématique est très général. Il est décrit en heuristique de calcul de sommation dans une situation comparable [Graham 89, p32]. Il s'applique dans les séries pour calculer que $\sum 1/j^2 = \pi^2/6$ sachant de façon analytique que $\sum 1/(2j+1)^2 = \pi^2/8$. La relation est alors $A=A/4 + C$. Les opérations de déduction proposées sont donc très générales et s'appliquent à des domaines mathématiques différents. Rappelons notamment que les exemples de sommation sont mathématiquement très proches du calcul intégral.

La fréquence des opérations de déductions permet d'envisager une assistance spécifique en fonction de l'opération. Cette assistance peut avoir lieu sur plusieurs plans :

- anticipation : quasiment impossible sauf quand l'assistant Ami gère la résolution de problème.
- diagnostic : possible à partir du moment où l'assistant Ami connaît l'opération de déduction et attend une situation où le terme initial est reproduit (même si elle ne doit avoir lieu).
- synthèse : il s'agit d'appliquer l'opération de déduction reconnue, mais surtout de prolonger la suite de la preuve en commencant la résolution de la relation obtenue.

Il est donc possible d'anticiper certaines causalités à partir des données grâce à des connaissances spécifiques. Il s'agit d'un **groupement de connaissances autour d'une opération de déduction**. L'assistance aux opérations de déduction nécessite alors la maitrise de la résolution de relations (équations) entre inconnues. Les principes d'isolation, collection, etc. énoncés par A. Bundy [Bundy 83] sont alors adaptables à la production d'opérations de déduction et à la gestion de leur ordonnancement.

Nous obtenons finalement pour cette preuve la démonstration suivante :

Démonstration utilisant le pouvoir de formalisation du Σ

Calculer la somme des n premiers entiers — *le problème à formaliser*

$$A = \sum_{j=0}^{n} j$$

= changement de variable j —> n-j

$$\sum_{j=0}^{n} (n-j)$$

= décomposition des termes d'un Σ

$$\sum_{j=0}^{n} n - \sum_{j=0}^{n} j$$

D'où — *production d'une " déduction logique"*

$$2A = \sum_{j=0}^{n} n$$

= mise en facteur de n

$$n \sum_{j=0}^{n} 1$$

= définition du Σ

$$n\,(n+1)$$

Soit — *production d'une " déduction logique"*

$$A = \frac{n\,(n+1)}{2}$$

cqfd.

Le mathématicien débute de A à B comme Gauss, mais il a trouvé ici un segment de preuve qui relie B à C-A. Dans ce cas l'opération de déduction est prévue à l'apparition de C-A et peut donc être proposée par un assistant Ami qui exploite des connaissances spécialisées.

Dans le cas de Gauss, rien ne laisse prévoir l'opération et l'Ami doit attendre le second segment débutant par A+B pour diagnostiquer qu'une opération de déduction utilisant A=B est en cours. Les solutions plus astucieuses étant atypiques, leur assistance est plus difficile et doit se contenter de connaissances très générales.

2.4. UNE DEMONSTRATION PAR RECURRENCE

Nous introduisons une dernière démonstration du calcul de la somme des n premiers nombres entiers afin d'introduire les démonstrations par récurrence. Elles se prêtent plus facilement à une assistance informatique car leur mise en œuvre est plus contrainte. Par contre, elles présentent deux inconvénients principaux qui fait qu'elles ne sont pas employées systématiquement par les mathématiciens :

- d'un point de vue heuristique, elles nécessitent la connaissance a priori du résultat ;
- d'un point de vue élégance, elles sont souvent boudées lorsqu'il existe des arguments plus directs.

Démonstration utilisant la connaissance du résultat

Montrer par récurrence :

P(n) $\sum_{j=0}^{n} j = \frac{n\,(n+1)}{2}$ — *problème nommé et formalisé*

On a : — substitution de n par 0

$\sum_{j=0}^{0} j = \frac{0\,(0+1)}{2}$

$\Leftrightarrow$ simplification

$0 = 0$ — *cqfd partiel*

Supposons P(n) :

$\sum_{j=0}^{n+1} j$

$=$ extraction du dernier terme

$\left(\sum_{j=0}^{n} j\right) + n+1$

$=$ P(n) déplié

$\frac{n\,(n+1)}{2} + n+1$

$=$ mise en facteur de n+1

$\left(\frac{n}{2} + 1\right)(n+1)$

$=$ factorisation

$\frac{(n+2)}{2}(n+1)$

$=$ mise sous forme $\frac{N(N+1)}{2}$

$\frac{(n+1)\,(n+2)}{2}$ — identification de P(n+1)

Donc

$\forall n, \sum_{j=0}^{n} j = \frac{n\,(n+1)}{2}$ — *cqfd.*

La construction de cette démonstration apparaît plus facile informatiquement que pour les démonstrations précédentes. La récurrence présente ici l'avantage d'éliminer rapidement le Σ afin de laisser place à des calculs que la connaissance du résultat peut guider efficacement. La difficulté heuristique principale est la recherche d'une relation Σ qui fait apparaître l'hypothèse P(n).

Comme pour les opérations de déduction, la démonstration par récurrence nécessite un groupement de connaissances spécifiques. La mise en œuvre d'une récurrence sur les entiers consiste à :

- qualifier la relation en fonction d'une ou plusieurs variables ;
- déterminer une borne entière initiale ;
- déterminer pour chaque variable un mode de récurrence (par ex. supposer P(n-1) et P(n) pour montrer P(n+1)).

Un choix par menu peut être associé à une **mise en œuvre générale d'une transformation de problème par récurrence** afin de déterminer les paramètres précédents. Toutefois, l'avantage de ce type de démonstration est que ces paramètres varient peu et peuvent être déterminés par un assistant ami :

- l'analyse de la relation permet de déterminer les noms des variables entières et de proposer un symbole de relation par défaut.
- la borne initiale est la plus petite pour laquelle la relation ait un sens, ce qui revient à envisager ici si les fonctions sont définies pour la substitution de n par 0.

- le mode de récurrence est plus difficile à déterminer mais une valeur par défaut fiable peut être proposée telle que supposer P(n) pour montrer P(n+1). Le cas général permet de choisir parmi P(n) et P(n+1) pour la relation à montrer, et de déterminer le nombre d'antécédents à supposer (1, 2, 3, etc. ou tous).

La donnée de ces paramètres permet l'exploitation de la récurrence. Dans notre exemple, ils permettent d'aboutir au schéma de preuve suivant :

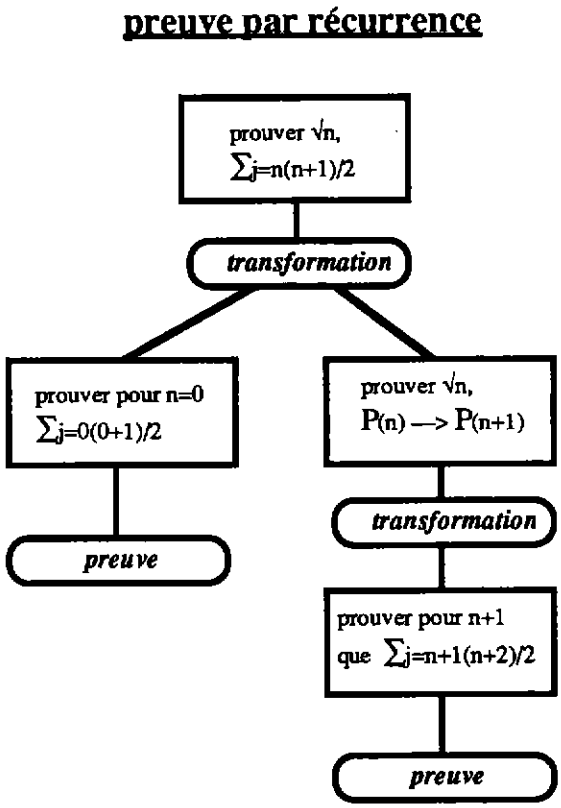


figure 2.5

Ce schéma de preuve comprend des problèmes, des transformations de problèmes et des preuves. Il n'est pas figé mais est construit par l'expertise spécifique aux démonstrations par récurrence. La nature des transformations à effectuer fait partie intégrante de cette expertise. Tous les problèmes sont produits à partir du problème initial et des paramètres de contrôle du mathématicien. Les preuves terminales sont construites indépendamment, respectivement ici par équivalence et par égalité.

Le mécanisme de production de ce schéma n'est pas une structure prédéfinie dans laquelle le mathématicien se contente de placer des éléments du problème. Il y a instanciation et opérations mathématiques sur les formules pour produire les divers problèmes. Le schéma global d'organisation des problèmes n'est pas figé. Il est lui aussi construit par morceaux, et comporterait par exemple deux problèmes instanciés pour n=0 et n=1 s'il avait fallu supposer P(n-1) et P(n) pour montrer P(n+1).

La ramification de droite n'est pas obligatoire et fait partie d'un mode de résolution usuel des problèmes de type A implique B. Dans ce cas, l'utilisation de l'hypothèse A est recherchée et sa présence est obligatoire dans la preuve. Nous retrouvons ici qu'il est indispensable en mathématiques de raisonner conjointement à partir des données de début de preuve, et des buts à atteindre dans son développement. Il s'agit d'une **stratégie fins-moyens** telle qu'elle a été définie par les pionniers de l'IA.

Nous présentons en Annexes une autre version de cette démonstration par récurrence, expliquée indépendamment dans [Solow 82]. L'explication fournie permet de retrouver le schéma de preuve par récurrence, d'apprécier la diversité des connaissances mises en œuvre, et de mettre en évidence les connaissances implicites nécessaires à la production et à l'interprétation d'une démonstration produite par un mathématicien, même quand elle se veut très détaillée. Elle illustre la nécessité des aides à la construction et à la progression déjà présentées figure 2.1.

2.5. LES OPERATIONS MATHEMATIQUES ELEMENTAIRES

Les exemples de démonstration que nous avons introduits utilisent en partie droite des opérations mathématiques élémentaires. Il serait souhaitable que le mathématicien n'ait qu'à indiquer l'information présente en partie droite pour effectuer une transition. Malgré leur qualificatif d'élémentaire, les quatres opérations requises dans nos deux premiers exemples nécessitent un savoir-faire auxiliaire important.

Le changement de variable j —> n-j nécessite pour sa mise en œuvre une action de remplacement dans la formule et dans les bornes accompagnée d'une action de résolution incluant des calculs annexes tels n-j=0 d'où j=n.

La mise en facteur de n et la définition du Σ utilisent des relations spécifiques au Σ. Ces relations utilisent des relations annexes comme "n=n·1" ou "le nombre de termes du Σ est la borne sup moins la borne inf plus 1", soit (n-0+1).

Le regroupement de Σ exploite la relation $\sum_{j=p}^{q} a_j + \sum_{j=p}^{q} b_j = \sum_{j=p}^{q} a_j + b_j$. Cette relation est ici instanciée aux données du problème, et exploitée avec un savoir-faire spécifique puisque "$a_j + b_j$" sont simplifiés. De plus, cette relation sert aussi en sens inverse dans notre deuxième exemple avec "$a_j + b_j$" égal à "n-j". L'utilisation d'une telle relation fait donc appel à des mécanismes complexes que nous étudions spécifiquement en partie III.

A cela vient s'ajouter une opération mathématique plus inusuelle concernant la formulation. Elle consiste à traduire "la somme des n premiers entiers" en $\sum_{j=0}^{n} j$.

L'exemple de la démonstration par récurrence a introduit, par ailleurs, des opérations supplémentaires et de nouvelles situations. Cela comprend par exemple l'utilisation de l'hypothèse de récurrence, qui est introduite dynamiquement, la reconnaissance de facteurs communs et la mise sous forme $\frac{N(N+1)}{2}$.

Nous avons donc globalement deux types d'opérations mathématiques élémentaires selon qu'elles sont :

- décrites par leur effet sur l'état initial et l'état final de la preuve (dites **relation à effet**) : changement de variable j —> n-j, substitution de n par 0, simplification, factorisation, mise sous forme $\frac{N(N+1)}{2}$, etc.
- décrites par une **relation à instancier** et à adapter au contexte de la preuve : regroupement de Σ, définition de Σ, extraction du dernier terme, etc.

Les relations à effet sont applicables à de nombreux états différents et ne peuvent être valablement décrites par une relation à instancier. Elles exploitent des caractéristiques mathématiques plus fines telles que des règles microscopiques comme 0.x=0, des capacités de perception permettant la reconnaissance de constantes et de facteurs communs, et un savoir-faire d'utilisation de règles pour la mise en forme et la simplification.

Les relations à effet permettent de mieux s'adapter à l'état du problème lorsqu'il n'existe pas de relation traitant le cas général. Même lorsqu'une telle relation générale existe, décrire un effet à rechercher permet de bénéficier implicitement de la relation la mieux appropriée, si tant est que l'Ami ait les connaissances pragmatiques nécessaires. Par exemple, pour le calcul de l'élément différentiel de "$y = \frac{a}{x^3}$", la relation de dérivation proposée n'est pas celle d'une fraction de deux fonctions "$y = \frac{f(x)}{g(x)}$". Par contre, pour celui de "f(x,y)=0", la relation de dérivation choisie sera "$y_x' = -\frac{f_x}{f_y}$".

Les opérations mathématiques décrites par une relation à instancier présentent des caractéristiques plus stables. Elles ont pour mérite d'être facilement accessibles, et de permettre à l'Ami de déterminer leur emplacement d'application privilégié. Elles peuvent être caractérisées par le schéma suivant :

- nom ;
- accès pour l'emplacement d'application dans la preuve ;
- valeur de la relation à instancier ;
- substitutions d'instanciation avec ajustements ;
- calculs et preuves de justification de l'application.

Le mécanisme d'instanciation est notamment très clair dans l'exemple archétype d'intégration par parties :

$$\int u'v = uv - \int uv'. \quad [5]$$

Cette relation est utilisée dans le problème Calculer la primitive de Log x. Le mathématicien écrit par exemple :

$$\int \text{Log } x \, dx = x \text{ Log } x - \int x \, \frac{1}{x} \, dx = x \text{ Log } x - x.$$

Les substitutions d'instanciation avec ajustements résultent d'une unification mathématique entre la relation et l'état de la preuve. L'ajustement est ici 1.Log x = Log x, et la substitution est u' —>1 et v —> Log x.

Les calculs et preuves de justification de l'application déterminent l'instanciation des variables manquantes du membre droit à savoir u et v'. La réutilisation de ce membre droit est alors possible.

Notons finalement qu'il est toujours possible de décrire une opération mathématique par son effet sur l'état initial et l'état final de la preuve. Les relations à instancier peuvent donc parfois être utilisées comme les relations à effet. Le nom des relations à instancier est d'ailleurs souvent un indicateur de l'effet obtenu (intégration par parties).

Nous avons ainsi défini un modèle de preuve et de démonstration détaillé. Pour l'implanter sur un support informatique, le problème principal sur lequel nous butons est a priori inattendu : il s'agit de **modéliser des opérations et des tâches mathématiques élémentaires**. Les connaissances pragmatiques pour cela sont très peu formalisées, et leur acquisition nécessite à elle seule un arsenal de techniques sophistiquées. Il apparaît de plus que des formalismes permettant l'expression de ces connaissances restent à définir en restant proche d'une description en Langage Mathématique (notamment en ce qui concerne le vocabulaire).

Nous abordons dans ce qui suit l'extension à des exemples de démonstrations plus complexes, indépendamment des aspects spécifiquement informatiques. Nous continuons à explorer sur ces exemples la modélisation ergonomique de l'activité, c'est-à-dire la modélisation des tâches et des actions du couple mathématicien-système dans l'activité mathématique commune.

Dorénavant, **notre problème principal est d'ordre ergonomique**. La faisabilité de ces opérations ne pose pas de problème et le détail de leur mise en œuvre peut rester masqué. Ce qui nous intéresse est de savoir comment des manipulations ergonomiques peuvent être utilisées. Il serait vain en effet de proposer une approche d'assistance aux tâches élémentaires si le mathématicien ne pouvait bâtir facilement des preuves complexes.

[5] Remarquons que cette relation est une présentation mnémonique sans élément différentiel ni expression de la variable liée. Nous n'abordons pas ici l'adaptation syntaxique que cela suppose.

3. ASSISTANCE LORS DE LA CONSTRUCTION DE PREUVES

3.1. LA DEMONSTRATION DE LA FORMULE DU BINOME

La démonstration par récurrence de la formule du binôme a été analysée dans sa version intuitive en partie I. Il est possible de la modéliser dans l'optique d'une construction assistée. La partie significative de sa construction est la gestion du développement principal de la récurrence qui manipule des **formules complexes**. Des opérations s'effectuent alors simultanément sur plusieurs termes de la formule, et la transition d'une preuve ne peut se résumer à une opération mathématique élémentaire. C'est pourquoi la justification des transitions s'effectue en terme de tâches ou d'opérations marquantes.

La preuve qui correspond à cette démonstration utilise alors, selon notre modèle de preuve à granularité variable, des **niveaux de décomposition**. Pour construire cette preuve, il est néanmoins souhaitable de disposer de la formule générale car les opérations effectuées sont coordonnées par le mathématicien en fonction de ce niveau global. Les niveaux de décomposition inférieurs doivent donc rester implicite ou apparaître temporairement dans une preuve annexe le temps de leur développement.

Nous abordons d'abord le début du développement principal. Le début de la formule complexe provient de la distribution des facteurs constants (a+b). Notons que cette opération va à l'encontre de la tradition établie dans les algorithmes de calcul préprogrammés, qui tendent à éliminer systèmatiquement les facteurs constants. De façon analogue, appliquer la règle x^{n+1} —> x x^n est inusuel car elle peut donner lieu à un bouclage dans une application récursive incontrôlée.

Automatiser de telles opérations nécessiterait alors un mécanisme de guidage très précis relatif à une expertise spécifique aux raisonnements par récurrence. Elle pourrait par exemple trouver comment déplier P(n) et suggérer la distribution de (a+b) comme une opération intéressante.

Un système qui sache retrouver en sens inverse $(a+b)^{n+1}$ à partir de la somme des deux Σ serait d'une utilité importante pour les manipulations de formules. Il permettrait d'automatiser certaines parties de la construction. Dans le sens direct, il effectuerait les deux premières opérations à partir du but de la récurrence. Un tel système devrait disposer d'un mécanisme général de raisonnement fin-moyens, et pouvoir exploiter des relations mathématiques sans décrire des informations heuristiques trop importantes.

Démonstration par récurrence de la formule du binôme

Montrer par récurrence :

P(n) $(a+b)^n = \sum_{j=0}^{n} C_n^j a^j b^{n-j}$ *problème nommé et formalisé*

On a : substitution de n par 0

$(a+b)^0 = \sum_{j=0}^{0} C_0^j a^j b^{0-j}$

$\Leftrightarrow$ simplification des deux membres

$1 = C_0^0 a^0 b^{0-0}$

$\Leftrightarrow$ simplification supplémentaire

$1 = 1 \times 1 \times 1$ *cqfd partiel*

Supposons P(n) :

$(a+b)^{n+1}$

$= x^{n+1} = x\, x^n$

$(a+b)\,(a+b)^n$

= P(n) déplié

$(a+b) \sum_{j=0}^{n} C_n^j a^j b^{n-j}$

= distribution de (a+b)

$\sum_{j=0}^{n} C_n^j a^{j+1} b^{n-j} + \sum_{j=0}^{n} C_n^j a^j b^{n-j+1}$

= extraction des termes extrêmes

$\sum_{j=0}^{n-1} C_n^j a^{j+1} b^{n-j} + C_n^n a^{n+1} b^0 + C_n^0 a^0 b^{n+1} + \sum_{j=1}^{n} C_n^j a^j b^{n-j+1}$

= homogénéisation des bornes

$\sum_{j=1}^{n} C_n^{j-1} a^j b^{n-j+1} + C_n^n a^{n+1} b^0 + C_n^0 a^0 b^{n+1} + \sum_{j=1}^{n} C_n^j a^j b^{n-j+1}$

= regroupement de Σ en utilisant $C_n^p = C_{n-1}^{p-1} + C_{n-1}^p$

$\sum_{j=1}^{n} C_{n+1}^j a^j b^{n-j+1} + C_{n+1}^{n+1} a^{n+1} b^0 + C_{n+1}^0 a^0 b^{n+1}$

= regroupement des termes

$\sum_{j=0}^{n+1} C_{n+1}^j a^j b^{n+1-j}$

Nous proposons d'effectuer les transformations complexes morceau par morceau en découpant la formule en zones d'intérêt. Cela donne un découpage visuel tel que :

$$\sum_{j=0}^{n} C_n^j a^{j+1} b^{n-j} \quad + \quad \sum_{j=0}^{n} C_n^j a^j b^{n-j+1}$$

① | ②

$$\sum_{j=0}^{n-1} C_n^j a^{j+1} b^{n-j} + C_n^n a^{n+1} b^0 + C_n^0 a^0 b^{n+1} + \sum_{j=1}^{n} C_n^j a^j b^{n-j+1}$$

① | ② | ③ | ④

$$\sum_{j=1}^{n} C_n^{j-1} a^j b^{n-j+1} + C_n^n a^{n+1} b^0 + C_n^0 a^0 b^{n+1} + \sum_{j=1}^{n} C_n^j a^j b^{n-j+1}$$

① | ② | ③ | ④

$$\sum_{j=1}^{n} C_{n+1}^j a^j b^{n-j+1} + C_{n+1}^{n+1} a^{n+1} b^0 + C_{n+1}^0 a^0 b^{n+1}$$

① | ② | ③ |

$$\sum_{j=0}^{n+1} C_{n+1}^j a^j b^{n+1-j}$$

Donc $(a+b)^{n+1} = \sum_{j=0}^{n+1} C_{n+1}^j a^j b^{n+1-j}$ *cqfd.*

La production des formules complexes nécessite des manipulations plus nombreuses, ne serait-ce que pour désigner la zone d'intérêt actuel de la formule. La première transformation donne :

① sélection de la sommation de gauche et application de l'opération d'extraction du dernier terme.
② sélection de la sommation de droite et application de l'opération d'extraction du premier terme.

Ces deux opérations interviennent en parallèle dans un ordre indifférent, qui est a priori gauche-droite. La transformation suivante fait intervenir quatre zones d'intérêt, dont seule la première est utilisée pour faire un changement de variable j —> j-1.

3.2. LA PRODUCTION D'UNE TRANSFORMATION DE FORMULE COMPLEXE

La transformation la plus complexe est la suivante :

$$\sum_{j=1}^{n} C_n^{j-1} a^j b^{n-j+1} + C_n^n a^{n+1} b^0 + C_n^0 a^0 b^{n+1} + \sum_{j=1}^{n} C_n^j a^j b^{n-j+1}$$

① | ② | ③ | ④

$$\sum_{j=1}^{n} C_{n+1}^j a^j b^{n-j+1} + C_{n+1}^{n+1} a^{n+1} b^0 + C_{n+1}^0 a^0 b^{n+1}$$

Sans assistance particulière, le mathématicien sélectionne deux zones : ① et ④. Il utilise l'opération de regroupement des Σ pour obtenir :

$$\sum_{j=1}^{n} \left(C_n^{j-1} + C_n^j\right) a^j b^{n-j+1}.$$

Il utilise alors explicitement la relation entre les coefficients du binôme :

$$C_n^{j-1} + C_n^j = C_{n+1}^j.$$

L'exploitation de capacités de raisonnement permettrait de commander le regroupement de Σ et la relation en indiquant seulement d'utiliser la relation précédemment démontrée (cf partie I) :

$$C_n^p = C_{n-1}^{p-1} + C_{n-1}^p$$

Il est en effet envisageable de trouver une expertise générale permettant de déterminer que cette relation est applicable et comment le faire. L'analyse fin-moyens que cela suppose permettrait de trouver qu'il faut utiliser le regroupement de Σ. Nous supposons ici que ce regroupement a été indiqué et que le problème est réduit à :

$$\text{utiliser } C_n^p = C_{n-1}^{p-1} + C_{n-1}^p \text{ pour transformer } \sum_{j=1}^{n} \left(C_n^{j-1} + C_n^j\right) a^j b^{n-j+1}.$$

Ce problème constitue l'un des problèmes essentiels à résoudre pour permettre une assistance ergonomique, bien qu'il soit considéré comme une tâche élémentaire par le mathématicien. La formulation générale de ce problème est :

utiliser <relation à instancier> pour transformer <formule>

La solution attendue est ici, en reprenant le schéma général d'une relation à instancier :

- nom :
  relation entre coefficients du binôme ;
- accès pour l'emplacement d'application dans la preuve :
  le premier facteur du terme général du Σ ;
- valeur de la relation à instancier :
  $C_{n-1}^{p-1} + C_{n-1}^p = C_n^p$;
- substitutions d'instanciation avec ajustements :
  n-1 —> n et p —> j ;
- calculs et preuves de justification de l'application :
  simplification de l'instanciation par équivalence mathématique de p-1—>j-1 et p—>j.

Lors de ce développement des zones ① et ④, le mathématicien construit donc une preuve annexe sur plusieurs niveaux de détail (le résultat visible, le regroupement puis le binôme, la preuve de la relation du binôme si elle n'est pas connue...). Nous pouvons par ce biais capturer les opérations élémentaires employées afin de vérifier, réutiliser ou présenter différemment cette preuve. Il est par exemple possible de présenter cette preuve comme nous l'avons fait, en plaçant simplement en partie droite de la démonstration un commentaire désignant l'emploi de la relation entre coefficients du binôme.

Les opérations dans les zones ② et ③ consistent à utiliser des relations inusuelles comme : $\forall n, C_n^n = 1$ pour montrer l'équivalence entre les termes. Ces relations proviennent d'une **analyse de la définition** des coefficients du binôme et de sa particularisation à des relations particulières entre les variables. Elles ne sont donc pas fournies par le mathématicien, mais elles sont déduites par un assistant Ami à partir de la définition pour des situations typiques. Cela nécessite des capacités de calcul pour simplifier automatiquement $\frac{n!}{n!(n-n)!}$, et une connaissance des variables pour généraliser $\forall n$.

La mise en œuvre de la transformation de ces zones se fait difficilement à partir de ces relations inusuelles qui sont peut-être non disponibles. Le plus ergonomique consiste à effectuer une sélection du coefficient à changer, suivi d'un remplacement syntaxique de n —> n+1 et d'une demande de vérification mathématique du résultat. Cette façon de procéder consiste à construire rapidement l'état final indépendamment de tout critère mathématique, puis à demander à un assistant Ami de vérifier le résultat $C_n^n = C_{n+1}^{n+1}$. Cela évite de devoir gérer une transformation trop lourde à réaliser de façon mathématiquement valide.

Nous avons ici un exemple de relation à instancier : $\forall n, C_n^n = 1$, qui est plus facilement décrite par l'effet à obtenir. Cela représente donc une autre forme d'assistance qui correspond au problème :

trouver <relation à instancier> pour réaliser <effet>

Une autre voie ergonomique pour produire cette transformation de formule complexe consistait à utiliser une stratégie de preuve à partir des deux bouts pour se rejoindre lors de la transformations complexe ci-

dessus. Le mathématicien peut obtenir la formule inférieure et il n'a qu'à détailler les associations à effectuer : les zones ① et ④ se regroupent (...) tandis que les zones ② et ③ se correspondent entre les deux formules. Ceci est une autre façon de vérifier que $C_n^n = C_{n+1}^{n+1}$ en produisant l'effet à obtenir à l'aide de buts de la résolution du problème mathématique général.

Les capacités de raisonnement sont ainsi utiles pour **faciliter les manipulations de formules** en utilisant des informations minimales. Ce type d'exploitation dans des situations très diverses ne peut a priori bénéficier de connaissances heuristiques trop spécialisées. Il est urgent de chercher à satisfaire des tâches élémentaires avec des informations restreintes aux informations mathématiques "usuelles". Cela nous semble un objectif prioritaire dans l'extension des outils mathématiques.

3.3. LA CONSTRUCTION PAR ISOLEMENT-REGROUPEMENT

Nous avons entrevu dans la construction de preuve précédente l'intérêt de travailler depuis une formule complexe en isolant et en transformant des termes : certains termes sont isolés et développés en des preuves séparées, puis regroupés pour former l'état suivant. Ce procédé de construction est suffisamment fréquent pour que nous en fassions une opération à part entière afin d'aider le mathématicien à gérer les paramètres à introduire. Cette opération, que nous nommerons *isolement-regroupement*, peut donner lieu à des développements qui forment la partie principale de la preuve, bien qu'elle se résume au niveau supérieur en deux états !

opération d'isolement-regroupement

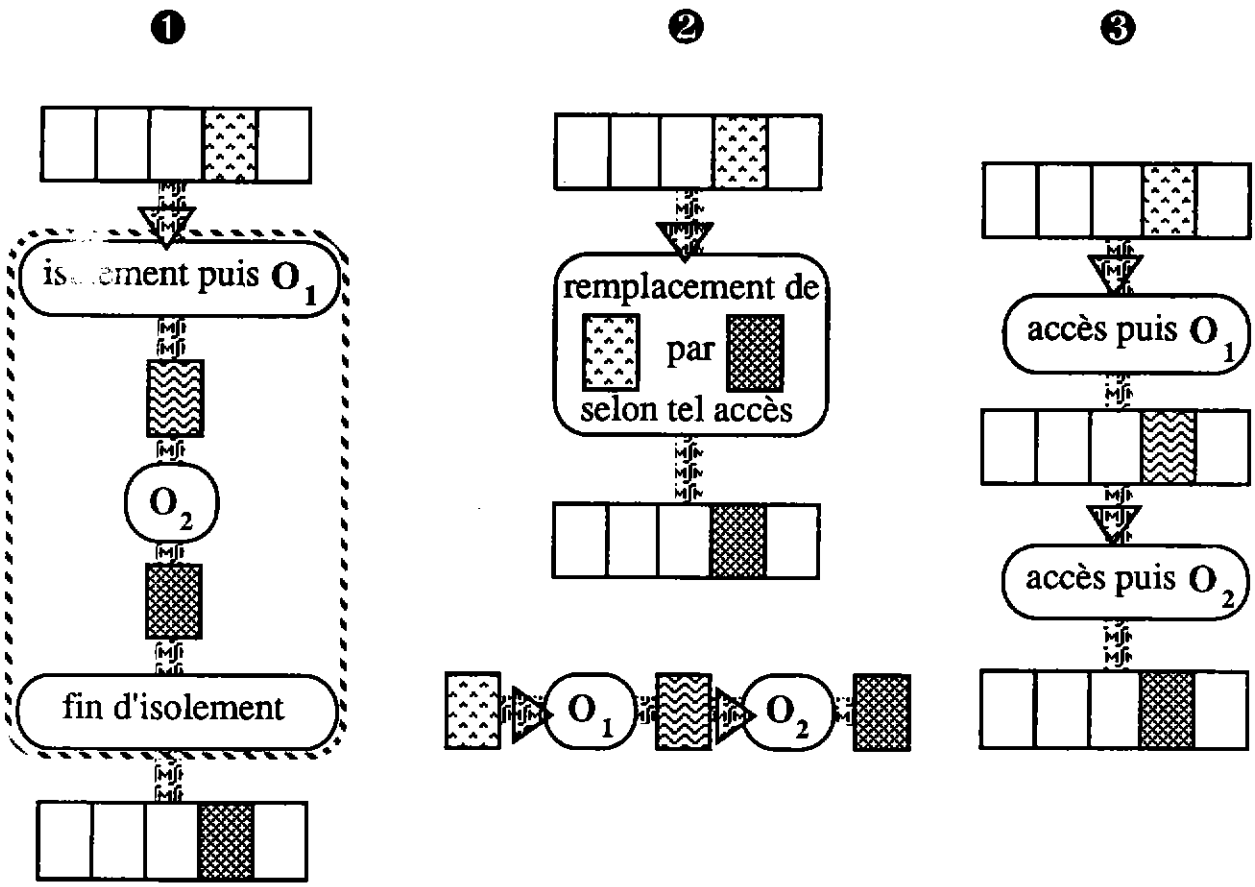


figure 2.6

Considérons le cas le plus simple d'un isolement unique dont le résultat du calcul est substitué. Le segment de preuve construit a, dans notre exemple précédent, deux opérations : l'opération O_1 est le

regroupement des deux Σ en un seul, l'opération O_2 est l'utilisation de la relation entre coefficients du binôme.

Le schéma❶ décrit l'opération d'isolement-regroupement par une vue du dessus de la preuve à granularité variable. La formule est découpée en zones et schématisée ici par cinq zones rectangulaires. Les rectangles arrondis représentent les actions du mathématicien. Le cadre pointillé indique que cette opération n'est qu'une étape de la preuve au niveau supérieur, mais qu'elle se décompose en une autre preuve à un niveau de détail plus bas, grâce notamment aux opérations O_1 et O_2.

Le schéma❷ représente une présentation de la preuve produite au niveau R0 avec en dessous une présentation du niveau R1. Le fait que ces deux niveaux appartiennent à une même preuve à granularité variable et non à deux preuves séparées est un avantage important car :

- ces deux niveaux sont prêts à se correspondre (pas de résultat trop général à instancier, homogénéité de notation acquise, etc.) ;
- les liens de causalité entre les niveaux et le parcours choisi par le mathématicien peuvent être capturés ;
- la possibilité de décomposer les niveaux permet de fournir des explications ou de tracer la provenance des termes dans une formule.

Le schéma❸ représente l'écueil que le mathématicien peut chercher à éviter : lors de la construction, être systématiquement obligé de décrire le même accès, et lors de la présentation, devoir présenter l'ensemble de la formule complexe alors que seule la zone transformer est pertinente.

La mise en œuvre de l'opération d'isolement-regroupement s'accompagne de nombreuses variantes en découplant à la fois :

- la définition des zones d'intérêt dans le formule et leur gestion dans l'interaction avec le système d'assistance (zone de travail séparée, etc.) ;
- le traitement séquentiel ou parallèle de la construction dans ces zones : lorsqu'il y a plusieurs zones à modifier, des actions en parallèle font parfois perdre le fil de la pensée ;

l'isolement-regroupement réduit les paramètres à introduire

Cette opération d'isolement-regroupement permet donc l'utilisation des liens de causalité entre les actions du mathématicien. Au lieu de copier directement le terme à isoler pour le développer indépendamment et présenter artificiellement le "bon" résultat prouvé, le mathématicien déclare son intention de le développer : plus besoin de le copier ailleurs ! Une fois développé (ainsi que d'autres éventuellement) lorsque le mathématicien décrète la fin d'isolement, son intention est claire : l'emplacement et la nature des substitutions sont connues ! Elle est elle-même définie en terme d'opérations plus élémentaires (sélection, copie...) avec des parties variables (nombre des isolements en parallèle et nature de leurs développements).

3.4. LES FACILITES DE COMMANDE

Nous considérons ici la construction de preuve comme l'activité principale du mathématicien. La conduite de cette activité - et son assistance - sont alors du ressort de la pragmatique. Nous sommes alors amenés à introduire une quatrième sorte de pragmatique du langage pour prendre en compte l'Ami, et compléter la section 2.3.1 partie I (pragmatiques du processeur, langagière et mathématique).

La **pragmatique de l'assistant** concerne l'activité "naturelle" transférée dans un autre contexte d'utilisation : elle s'efforce, avec des observables et des moyens de compréhension limités, de faciliter la tâche du mathématicien. Bref :

la pragmatique de l'assistant consiste à savoir quoi faire à tel instant précis.

L'assistance à la construction d'une preuve peut s'effectuer de deux façons différentes :

- en effectuant la résolution du problème mathématique ;
- en facilitant la construction pas à pas.

Ces façons d'assister sont bien sûr complémentaires. Nous nous intéressons à la seconde qui est indispensable à notre approche. Une opération mathématique comprend en général beaucoup trop d'informations pour que le mathématicien puisse se permettre de les donner toutes. L'assistance consiste donc à donner au mathématicien les moyens d'utiliser facilement ses opérations habituelles. La problématique de cette assistance est alors essentiellement **ergonomique**.

La commande d'un système informatique demande de nombreux paramètres même lorsque les opérations disponibles sont judicieusement choisies. L'alternative est alors :

- **spécifier tous les paramètres** et obtenir une opération illisible : c'est bien sûr ce qu'il est indispensable d'obtenir pour la machine ;
- **permettre des commandes allusives** ne fournissant que les paramètres essentiels : la machine doit alors effectuer de nombreuses inférences afin de reconstituer l'opération désirée par l'homme. C'est le rôle de la pragmatique de l'assistant.

Un problème est bien défini lorsque son état initial, état final et l'espace des transitions possibles est connu. Dans les exemples mathématiques précédents, seul l'état initial est connu en général et l'espace des transitions possibles est démesuré par rapport à celui des transitions souhaitées. Il y a alors deux grandes façons d'indiquer une commande à un assistant qui maitrise potentiellement l'espace des transitions :

- en précisant la séquence d'opérations (ou plus généralement un élément clé) afin de trouver un état final adapté ;
- en précisant l'état final (ou plus généralement l'effet à obtenir) afin de trouver la séquence d'opérations réalisant la transition.

Lors d'une relation à instancier, cela se traduit par les deux problèmes précédemment identifiés :

- utiliser <relation à instancier> pour transformer <formule> ;
- trouver <relation à instancier> pour réaliser <effet>.

Pour résumer, le mathématicien peut **dire ce qu'il cherche à obtenir, ou dire comment l'obtenir.**

Les fonctionnalités attendues d'un système sont alors de :

- disposer des opérations générales utilisables lors de la preuve ;
- accéder facilement à ces opérations ;
- trouver comment les utiliser dans la preuve.

Trouver comment utiliser les opérations est le rôle d'un Ami ; accéder facilement aux opérations et indiquer ses intentions par dialogue concerne les facilités de commande que nous décrivons ici.

Une première facilité d'accès aux opérations est celle d'une présélection des opérations en fonction de l'état initial. Elles peuvent alors être accédées par thème comme les opérations concernant le Σ, par leur nom ou par leur effet. Un premier principe de présentation pour permettre leur lisibilité est :

Une opération est décrite en langage de Formules ou de Phrases.

Par exemple une relation à instancier comme l'intégration par parties est décrite par son nom ou par sa formule correspondante. Les relations à effet sont décrites par leur nom comme : changement de variable $j \longrightarrow n-j$.

Les facilités de commande autorisées par le langage permettent de décrire des opérations sous des formes plus conceptuelles comme :

"n est impair" qui est traduit sous forme symbolique par "$\exists p,\ n=2p+1$".

Cette facilité de traitement du langage permet par exemple de construire la formule de "la sommation des n premiers entiers" à partir de cette description.

Ces facilités de commande offrent au mathématicien **plusieurs façons de procéder**. Nous l'illustrons dans le cas d'une relation à instancier pour débuter un morceau de preuve. Considérons par exemple la définition de la surface d'une courbe :

$$S(\rho) = \int \rho^2 \, d\theta \quad \text{.....(1)}$$

L'utilisation de cette relation à instancier peut se faire de trois façons :

- aller chercher directement la formule de la relation, ce qui est commode si elle est facilement visualisée ;
- invoquer la relation par un lien, tel son nom ou sa référence "(1)" ;
- suggérer son usage par complétion après un début comme (2) qui a le mérite d'indiquer de plus l'instanciation souhaitée (et de réduire le dialogue nécessaire dans les cas précédents) :

$$S\left(\left(\frac{\rho}{\rho_0}\right)^{\gamma}\right)= \quad \text{.....(2)}$$

Quelle que soit la façon choisie, le résultat produit sera[6] :

$$S\left(\left(\frac{\rho}{\rho_0}\right)^{\gamma}\right)= \int \left(\frac{\rho}{\rho_0}\right)^{2\gamma} d\theta \quad \text{.....(3)}$$

De manière générale, le lien entre une relation et son utilisation ne se réduit pas forcément à une instanciation : il peut comporter une **preuve annexe**. C'est le cas par exemple de la relation (4) issue d'un article sur la complexité [Duris 84] :

$$p \leq \frac{k\,T(n)}{g} \quad \text{.....(4)}$$

Cette relation étant donnée, l'auteur veut borner :

$$C_{(n)}^{\frac{4kp}{f}} \quad \text{.....(5)}$$

La preuve annexe à construire consiste à multiplier les deux membres de (4) par $\frac{4k}{f}$, puis à utiliser l'élévation à la puissance pour obtenir à gauche $C_{(n)}^{\frac{4kp}{f}}$. Le sens de l'inégalité est fonction de la connaissance du signe de $\frac{4k}{f}$: cette preuve peut même être scindée en deux cas si ce signe est inconnu. Cette preuve utilise donc des conditions sur les constituants ($f \neq 0$ et $kf \geq 0$) qui doivent être vérifiées ou mises en hypothèse. Le mathématicien obtient finalement la formule (6) complète accompagnée de la justification de l'inégalité et ses conditions de validité :

$$C_{(n)}^{\frac{4kp}{f}} \leq C_{(n)}^{\frac{4k^2T(n)}{gf}} \quad \text{.....(6)}$$

Cette preuve ne peut être produite que par la connaissance simultanée de la relation (4) et de l'état (5). Sans assistance, le mathématicien construit cette preuve annexe. La résolution de problème est ici guidée par une heuristique générale : lorsque le mathématicien dispose d'une relation entre variables, il cherche à l'utiliser dans ses constructions ultérieures. Indépendamment de cette résolution de problème, les facilités de commande pour l'assistance à cette construction sont :

- utiliser <relation à instancier> pour transformer <formule> :
  Le mathématicien peut aller chercher la relation parmi les résultats intermédiaires ou les données du problème. Il peut aussi y accéder par la variable "p" en demandant les relations qu'elle vérifie.
- trouver <relation à instancier> pour réaliser <effet> :
  Construire l'état final revient ici à effectuer l'essentiel du travail fastidieux de l'assistant (construire un état complexe). Paradoxalement ici, cela ne facilite pas la tâche de l'assistant qui doit analyser cet état final. Il est plus ergonomique d'indiquer à l'assistant l'effet qui est recherché : obtenir une inégalité.

[6] Notons que les problèmes abordés ici sont valables aussi pour l'utilisation des mathématiques dans d'autres sciences. Ainsi, la nature de cette formule fait penser à un problème physique.

Il est alors envisageable de proposer un **mode d'action direct** à partir de la construction des formules. Dans ce mode, il suffirait par exemple de taper "≤" pour indiquer l'effet recherché :

$$C_{(n)}^{\frac{4kp}{f}} \leq$$

De multiples opérations peuvent être décrites de cette façon, à condition d'associer un sens dirigé par l'effet de ces actions, par exemple :

$$\sum_{j=0}^{n} (n-j)$$

Le mathématicien déplace le signe "-" à gauche avant le Σ. L'effet désiré est alors d'obtenir le signe "-" à l'extérieur de l'opérateur Σ alors qu'il était auparavant à l'intérieur. L'opération qui réalise cela est la décomposition des termes d'un Σ afin d'obtenir :

$$\sum_{j=0}^{n} n - \sum_{j=0}^{n} j$$

De même, le mathématicien déplace le "n" à l'extérieur afin d'obtenir sa mise en facteur :

$$\sum_{j=0}^{n} n = n \sum_{j=0}^{n} 1$$

Au delà des facilités de commande, nous n'abordons pas explicitement les **initiatives de l'Ami**. Elles sont souvent dépendantes de capacités de résolution de problème d'un domaine spécifique. Ici encore, les aides les plus réalistes sont parfois très limitées : il peut s'agir, par exemple, de proposer une définition instanciée dès que son utilisation est pressentie. Cela peut être aussi, simplement, de proposer des choix judicieux de voisinage en "x_0", comme $\{x_0 + \frac{1}{n}\}$ ou $\{x_0 + f(\varepsilon)\}$, dès qu'un raisonnement intuitif sur la continuité est envisagé. De telles aides sont d'ailleurs indispensables pour le confort d'usage et la crédibilité d'un système.

Obtenir un système ergonomique ne nécessite donc pas forcément des connaissances très sophistiquées en terme de résolution de problèmes. L'ergonomie requiert principalement des **connaissances plus diversifiées** : l'essentiel réside alors dans les mécanismes d'exploitation qui sont alors rendus possibles. L'intégration de ces mécanismes crée alors une synergie dans l'exploitation de ces connaissances.

3.5. PRAGMATIQUE DE LA PRESENTATION

Démonstration:

Montrons l'associativité du produit :

$a*b(x,\omega) = \int a(y,\omega)\, b(-y+x,-y.\omega)\, \sigma(y,-y+x)\, dy$ (1)

$[(a*b)*c](x,\omega) = \int (a*b)(z,\omega)\, c(-z+x,-z.\omega)\, \sigma(z,-z+x)\, dz$ (2)

$[(a*b)*c](x,\omega) = \int\int a(y,\omega)\, b(-y+z,-y.\omega)\, \sigma(y,-y+z)\, c(-z+x,-z.\omega)\, \sigma(z,-z+x)\, dy\, dz$ (...) (3)

L'exemple ci-dessus, extrait d'un article d'un mathématicien, illustre l'intérêt d'une stratégie de présentation et de nommage. La difficulté essentielle est ici la gestion de la présentation. Le rappel de la tâche et de la définition (1) présente un intérêt cognitif certain : il visualise la définition à utiliser, fixe le choix des symboles et donne des indications sur la stratégie de nommage à employer. Notons enfin la simplification classique de la zone d'intérêt entre (2) et (3) après l'application de la définition (pas de parenthésage, regroupement des intégrales et des mesures, mais ordre des termes inchangé).

La pragmatique de présentation commence dès l'utilisation de relations dans une preuve. Elle présente les définitions et les relations à instancier selon une forme utilisable facilement dans la preuve. Elle adapte notamment le choix des symboles et des notations.

La transition entre (1) et (2) est une application élémentaire de la définition. Au lieu de respecter l'ordre d'apparition, une substitution "automatique" apparierait :

a —> a1*a2, b—> b	à moins que ce ne soit	a —> x113*x114, b—> x112

L'assistance consiste donc aussi à **proposer des symboles pertinents**, afin d'éviter à l'usager de le faire. Il faut pour cela modéliser et lui fournir des concepts pour agir globalement sur ces choix, tout en l'autorisant localement à revenir sur le choix de certaines notations.

La preuve se résume au niveau le plus abstrait à "(a*b)*c = a*(b*c)". Le détail des niveaux inférieurs doit alors rester compatible avec cette présentation, et respecter des critères d'homogénéité des notations même lorsque la stratégie de construction consiste à développer séparément les deux extrémités avant de les identifier. La stratégie de nommage est donc définie par des connaissances pragmatiques qui prennent en compte la formule générale de l'associativité.

Un point essentiel de l'assistance est donc la prise en compte des notations dans les formules manipulées et les opérations proposées. Un des inconvénient des systèmes actuels, est la pauvreté syntaxique et typographique des systèmes de calcul, qui contraste avec la diversité des systèmes de traitement de texte. L'assistance consiste donc aussi à produire une **transition acceptable** d'après les critères de qualité humains.

La présentation nécessite une expertise de mise en forme qui gère une formule à des niveaux différents. Cela est bien mis en évidence dans la démonstration de la formule du binôme :

$$\sum_{j=1}^{n} C_n^{j-1} a^j b^{n-j+1} + C_n^n a^{n+1} b^0 + C_n^0 a^0 b^{n+1} + \sum_{j=1}^{n} C_n^j a^j b^{n-j+1}$$

Le choix et la définition des symboles s'effectue au niveau de la preuve. Tous les états de la démonstration sont présentés avec les mêmes choix. Notons aussi que cela s'applique à des symboles mathématiquement indépendants comme les deux symboles de variable liés "j" de la formule.

A l'intérieur de la formule l'emplacement des quatre termes de la structure prégnante est régi par une convention. Cela peut être le degré pour les polynômes, ou plus simplement l'origine génétique des termes comme c'est le cas ici.

Pour chacun des quatre termes, à un niveau de structure plus fin de la formule, le mathématicien retrouve une **forme normale de présentation** : $C_n^p a^j b^k$. Cette forme normale a aussi une réalité mathématique pusiqu'elle est utilisée pour manipuler des termes par leur forme générale. Cette forme normale prime sur les simplifications mathématiques, puisque $C_n^0 a^0 b^{n+1}$ est tout simplement b^{n+1}.

Afin de travailler efficacement, le mathématicien doit pouvoir gérer de façon fine la présentation à ces différents niveaux. Par exemple, il peut définir une forme normale comme ci-dessus et décider qu'elle est valable pour tout le déroulement de la preuve, empêchant ainsi des simplifications parasites.

Il y a donc globalement pour la présentation :

- un résultat interne au système issu d'une opération ;
- des actions de mise sous forme normale interne qui contrôlent les simplifications ;
- un choix mathématique de nommage et de notation pour la présentation ;
- des contraintes informatiques agissant sur la présentation effective.

La présentation peut aussi bénéficier de **fonctionnalités dynamiques** spécifiques à un traitement informatique. Par exemple, la preuve peut être présentée graphiquement selon plusieurs niveaux de détail. Un style typographique (gras, souligné, couleur) peut être choisi pour chaque objet en fonction de son rôle mathématique. Cette dernière fonction permet en particulier la visualisation des symboles de variable, de l'origine d'un terme particulier, ou des occurences d'un symbole.

Pour ne détailler qu'un point, la **visualisation de l'origine d'un terme** peut s'avérer utile pour la compréhension d'une formule complexe. Elle permet au mathématicien de le situer dans une représentation plus intuitive. Ce suivi d'un terme doit même être automatisé lors d'applications physiques où la qualification du terme compte parfois plus que son contenu mathématique. Un terme identifié comme la force de frottement solide sur la poulie C pourra, par exemple, être négligé.

Ces fonctionnalités dynamiques sont difficilement descriptibles a priori. L'expérience de l'amélioration des interfaces montre que le vécu d'une interface est un critère important d'évaluation. De plus, les besoins émergent de la pratique et **les possibilités intéressantes ne sont pas forcément anticipées.**

4. FONCTIONNALITES D'UN SYSTEME D'ASSISTANCE

4.1. LA CONSTRUCTION DU TEXTE DE DEMONSTRATION

4.1.1. Production du contenu mathématique

Nous abordons maintenant quelques pistes permettant d'approcher la complexité de la production de démonstrations mathématiques telles qu'elles sont pratiquées. Considérons par exemple la session de production de l'étape de démonstration suivante, issue de [Quinet 62] (visible section 4.2.1.1 partie I) :

$$x^3 + y^3 = 3xy$$

Pour l'exprimer en coordonnées polaires, posons

$$x = \rho \cos \theta \text{ et } y = \rho \sin \theta$$

d'où

$$\rho^3 \cos^3 \theta + \rho^3 \sin^3 \theta = 3 \rho^2 \sin \theta \cos \theta$$

Cette session part d'une formule et indique quelle transition effectuer pour aboutir à une autre formule. Cette démonstration partielle est associée à un élément de preuve qui part de l'état : "$x^3 + y^3 = 3xy$", et qui arrive à l'état final grâce à la transition :

"changement de coordonnées cartésiennes —> polaires ; $x = \rho \cos \theta$ et $y = \rho \sin \theta$"

Cette session de quelques lignes va nous permettre de cerner quelques caractéristiques souhaitables d'un système d'assistance efficace. Observons l'usage d'un tel système :

- Une fois l'état initial introduit, quel est le moyen le plus simple de produire et pouvoir éventuellement vérifier la cohérence de cette étape de démonstration ?
  - Sans conteste, le mieux est de disposer de l'opération de changement de coordonnées cartésiennes —> polaires, et de l'appliquer.
- Quelle est la façon la plus simple de l'appliquer ?
  - Ne rien avoir d'autre à préciser ! Pour cela, le système doit avoir les moyens d'identifier et dénombrer l'ensemble des variables, de choisir et d'ordonner les variables à transformer x et y. Il doit enfin pouvoir générer les nouvelles variables ρ et θ ainsi que les relations de transformation.
- Mais si le système ne propose pas les bons choix de variables ?
  - L'usager agit alors uniquement sur les quatres variables d'entrée-sortie de l'opération (que nous appelons des *paramètres* de cette opération) pour les modifier à sa guise.
- Et si cette macro-opération de changement de coordonnées n'existe pas ?
  - L'usager en est réduit à indiquer intégralement la substitution, ce qui n'est d'ailleurs pas toujours évident en dimension 3... L'utilité du système d'assistance est alors considérablement diminuée sans ces opérations usuelles.
- Quel est le résultat de cette substitution ?
  - La forme brute est "$(\rho \cos \theta)^3 + (\rho \sin \theta)^3 = 3 \rho \cos \theta \rho \sin \theta$", ce qui est peu adapté aux besoins d'un mathématicien même en supprimant les parenthèses inutiles...
- Que se passe-t-il usuellement dans les systèmes existants ?
  - Les termes sont mis sous une forme normale qui regroupe (et simplifie) en quatre champs : les valeurs numériques, les constantes, les variables et les termes fonctionnels. A l'intérieur de ces champs, un ordre lexicographique arbitrairement édicté est utilisé. Les systèmes plus sophistiqués cherchent à appliquer un jeu de règles de simplification qui permettent d'obtenir :

$$\rho^3 \cos^3 \theta + \rho^3 \sin^3 \theta = 3 \rho^2 \cos \theta \sin \theta$$

▶ Et si le mathématicien désire malgré tout la présentation "3 $\rho^2 \sin\theta \cos\theta$" choisie dans la session ?

✍ Un système convivial doit permettre de définir d'autres règles de présentation, en indiquant par exemple que sinus doit précéder cosinus. Ce choix fait apparaître une lacune importante des systèmes actuels. Le membre gauche est lui aussi affecté par l'ordre de présentation et la somme permutée en "$\rho^3 \sin^3\theta + \rho^3 \cos^3\theta$".

▶ Quel est l'inconvénient de ces règles de présentation agissant après une opération ?

✍ Le suivi de l'action de l'opération devient plus difficile, et la clarté de l'explication diminue. L'organisation spatiale des éléments pertinents de la transition précédente doit rester proche du schéma initial. Ici, lorsqu'il substitue l'occurence de "x" dans le terme le plus à gauche, le mathématicien cherche le résultat dans le membre le plus à gauche. Les règles de simplification liées à la présentation doivent respecter ces contraintes de lisibilité qui mettent en évidence le lien avec l'état précédent.

▶ Comment évaluer la pertinence des simplifications liées à la présentation ?

✍ Il faut utiliser les notions de **structure prégnante et de zones d'intérêt** introduites en partie I. Les zones d'intérêt correspondent ici aux zones "proches" des objets de la formule intervenant dans l'opération (caractérisation ascendante des termes contenant x et y). La structure prégnante correspond à l'organisation spatiale de la formule (caractérisation descendante). Ainsi, les trois zones d'intérêt pour l'opération utilisée dans cette session concernent x et y, ce sont :

$\boxed{x^3} + \boxed{y^3} = \boxed{3xy}$	zones d'intérêt
A $\boxed{+}$ B $\boxed{=}$ C	structure prégnante

▶ Comment utiliser ce modèle ?

✍ Les zones d'intérêt agissent comme des limites au sein desquelles ont lieu des simplifications de présentation. La structure prégnante permet des repères spatiaux qu'il faut préserver. C'est pourquoi, la permutation des termes du membre de gauche est repoussée ici à l'état suivant ("ρ=..."). Les simplifications de présentation ne sont pas contraintes hors des zones d'intérêt et permettent des "rattrapages". Ce modèle permet de plus d'estimer la difficulté d'une séquence d'opérations. Il permet alors de choisir entre plusieurs séquences et aide à déterminer les états à présenter.

▶ Quel rôle jouent ces variations de voisinage dans l'usage d'un système ?

✍ Ils cherchent à déterminer le meilleur point de vue sur les constituants présents et leur origine. La lisibilité d'une preuve à granularité variable dépend d'un choix judicieux des symboles utilisés, de la disposition spatiale employée, et des états présentés. Nous avons cherché à déterminer les moyens de contrôler les paramètres de cette présentation. Cette connaissance de visualisation est essentielle pour la production d'une démonstration associée.

▶ Quel rôle pratique est amené à jouer un tel modèle ?

✍ Il permet d'obtenir une présentation acceptable en structurant les simplifications. Cela donne par exemple pour la démonstration par récurrence de la formule du binôme :

$$(a+b)^0 = \sum_{j=0}^{0} C_0^j \, a^j \, b^{0-j}$$

$\Leftrightarrow$ simplification des deux membres

$$1 = C_0^0 \, a^0 \, b^{0-0}$$

$\Leftrightarrow$ simplification supplémentaire

$$1 = 1 \times 1 \times 1$$

cqfd partiel

4.1.2. Définition de la structure interne d'une démonstration

La structure de preuve à granularité variable est déjà un premier pas important vers une démonstration, puisqu'elle fait intervenir des opérations de haut niveau d'abstraction qui correspondent à une démarche et à des états "naturels". La construction d'une démonstration va alors consister en deux étapes :

- une **explication** consistant à réutiliser l'information de la preuve ;
- une **présentation** "textuelle" de cette explication.

Nous parlons de la structure interne de la démonstration pour qualifier ces informations qui décrivent le texte visible. Notre objectif est de voir comment il serait possible de produire une démonstration qui satisfasse l'usager.

L'une des caractéristiques de notre approche est justement qu'elle tient à **laisser au mathématicien le contrôle des paramètres** pertinents de sa tâche. En ce sens, elle ne cherche pas à vérifier la totalité du produit, mais à assister sa construction et vérifier la cohérence des liens entre informations accessibles par le système. Ainsi, l'appréciation des critères de génération de démonstration et du "rendu" de cette démonstration est sous contrôle du mathématicien.

Les principales informations nécessaires à la production d'une démonstration sont déjà présentes dans la preuve. C'est pourquoi, il est utile d'introduire la structure interne d'une démonstration comme une **annotation de cette preuve**. Avec ce procédé, la façon de construire la preuve prépare la construction de la démonstration.

Nous proposons pour la structure interne d'une démonstration une séquence d'états indépendants de toute rigueur mathématique. Ces états sont accompagnés d'explications guidant la compréhension, qui visent principalement à élucider les transitions effectuées. C'est pourquoi la composition d'un **état interne d'une démonstration**, regroupe trois points :

① les caractéristiques de la transition produisant cet état ;

② la partie principale de l'état, c.-à-d. un état de la preuve annotée ;

③ les définitions des objets et abréviations introduits ;

Dans le cas d'un enchaînement très détaillé comme dans une succession d'égalités, les points ① et ③ sont omis, ce qui réutilise littéralement la preuve. Les trois champs sont toutefois présents dans le cas général, comme celui de l'exemple ci-dessous :

Calculons la surface de la boucle du folium de Descartes donnée par l'équation : $x^3 + y^3 = 3xy$ (1)
Une première étape consiste à la transformer en coordonnées polaires, ce qui donne : $\rho = \frac{3 \sin\theta \cos\theta}{\sin^3\theta + \cos^3\theta}$ (2) en posant $\begin{bmatrix} x = \rho \cos\theta \\ y = \rho \sin\theta \end{bmatrix}$

Le premier encadrement représente l'introduction du problème à l'aide des deux premiers points d'un état de démonstration. Le second encadrement est complet : la première phrase correspond au point ① ; l'équation référencée constitue le point ② tandis que la fin correspond au point ③.

Une variante de ces états internes de démonstration consiste à utiliser un segment de preuve en l'annotant par des informations provenant d'une transition du segment (à un niveau de détail quelconque). Ainsi le segment :

$$\int \mathrm{Log}\, x\, dx = x\, \mathrm{Log}\, x - \int x \frac{1}{x} dx = x\, \mathrm{Log}\, x - x.$$

peut être suivi d'un commentaire le justifiant de plusieurs façons :

- par intégration par parties ;
- en écrivant Log x = 1 Log x ;
- en posant u' = 1, v = Log x, on a u = x et v' = 1/x ; etc.

4.1.3. Assistance à la construction d'une démonstration

Maintenant que nous avons introduit une structure interne permettant de représenter les démonstrations, imaginons comment la construire. Les systèmes actuels existant se contentent au mieux d'insérer des parties à calculer dans un texte mathématique. Nous disposons dans notre système de la preuve en construction, c'est-à-dire a priori d'informations indispensables à la production d'une démonstration. La démonstration est alors un **document structuré** produit à partir d'une preuve, et partageant avec elle certaines données.

L'assistance à la construction d'une démonstration se résume alors principalement à la mise à disposition et à la combinaison ergonomique de schémas de présentation liés aux opérations mathématiques. Le mathématicien accède principalement à ces informations à partir de la preuve qu'il a construite.

Lorsqu'une démonstration explique une opération, les points ① et ③ proviennent de **connaissances de présentation** rattachées à cette opération mathématique. Ainsi, l'opération mathématique "changement de coordonnées cartésiennes —> polaires ; $x = \rho \cos \theta$ et $y = \rho \sin \theta$" est partitionnée en plusieurs champs structurés. Chacun est identifié et peut être inséré dans un schéma de présentation utilisant des sous-schémas. Des schémas proposés pour cette opération sont :

" Le passage en coordonnées polaires donne : " .....(a)

" Par un changement de coordonnées cartésiennes en coordonnées polaires, on a :"(b)

" En posant $x = \rho \cos \theta$ et $y = \rho \sin \theta$, on obtient : ".....(c)

Lorsque les transitions sont simples ou peu détaillées, elles peuvent de plus être omises ou réduites à des mot-clés (on a, d'où...). Ici, les schémas (a) et (b) sont simplement des présentations textuelles de l'opération utilisée. Pour des transitions plus complexes ou celles que l'on cherche à détailler, la transition est décomposée à l'aide d'opérations plus simples comme dans le schéma (c).

Le mathématicien utilise en (c) le schéma plus général du changement de variable. Il omet l'information plus spécifique concernant le changement de coordonnées car cela est "évident" pour un autre mathématicien (concision). Le choix d'une présentation pour la transition dépend des inférences mises en œuvre dans l'application de cette transition. Ainsi, si le choix des variables à transformer n'est pas évident, il doit apparaître explicitement dans la démonstration (désambiguation). Les schémas (a) et (b) seront ainsi rejetés. De tels aspects sont soumis à l'unique appréciation du mathématicien.

Le mathématicien peut aussi désirer employer des **schémas inédits** pour bâtir sa démonstration. Dans l'exemple du changement de coordonnées développé précédemment, l'auteur met en valeur son intention. Il faut préserver cette liberté de création tout en conservant le partage de structure entre preuve et démonstration.

Techniquement, outre la gestion d'une structure complexe, cela nécessite une interface sophistiquée permettant la création dynamique de nouvelles structures. Les techniques informatiques sont disponibles, mais les produits n'en sont pas encore là. L'usager pourrait construire un schéma spécifique en récupérant des sous-schémas prédéfinis. Le schéma désiré est construit ici à partir des 4 champs encadrés :

Pour l'exprimer en [coordonnées polaires], [posons]

[$x = \rho \cos \theta$ et $y = \rho \sin \theta$]

[d'où]

Un tel schéma serait **assemblé** en considérant que l'opération de changement de coordonnées est un cas particulier du changement de variables. Ce dernier est décomposé en une tâche d'introduction, une tâche de visualisation du changement de variables, et une tâche de transition. Chacun de ces points apparaît ici sur une ligne différente. Chaque tâche propose plusieurs champs que l'usager peut affiner.

Chaque champ est la présentation d'une structure "profonde". [posons] est ainsi une présentation de l'action "poser" sous forme impérative. Ces présentations varient selon les accords de la phrase, des critères grammaticaux (actif, passif)... De façon similaire, le changement de variable central peut être décomposé en des champs plus élémentaires jusqu'aux composants mathématiques des formules !

Ici, la première ligne provient d'une extension de la tâche d'introduction à partir du champ [posons], avec insertion d'un champ issu du changement de coordonnées. Les parties non encadrées sont considérées comme non significatives pour le système et ne sont pas réutilisée. Toutefois, des actions transversales de cohérence grammaticale ou de mise en page peuvent être effectuées sur l'ensemble de cette démonstration partielle.

L'intérêt de disposer d'une représentation structurée d'une démonstration est faire correspondre le texte avec l'endroit précis de la preuve à granularité variable correspondante. Cela permet en particulier au lecteur d'accéder à une présentation rigoureuse et complète de la preuve d'un passage textuel. L'aboutissement ultime de cette **dualité** entre preuve et démonstration serait de générer intégralement une démonstration comme une explication de la preuve.

4.2. FONCTIONNALITES DE PERCEPTION

4.2.1. Un modèle de traitement perceptif

Vu l'accommodation réciproque [illegible]athématiciens à leur langage, optimiser la communication intervient dès le plan perceptif. Nous présen[illegible]s ainsi sommairement le mécanisme interne de tout **processus de traitement perceptif par un agent**. Il se décompose en :

① détection ;
② identification ;
③ réaction adaptée.

Ce mécanisme est applicable à plusieurs niveaux de structure pour les langages de Croquis, Figures, Formules et Phrases. Ainsi, dans le cas d'un langage de Phrases, il s'applique aux niveaux des mots (voire des signes), des énoncés, des liens entre énoncés, de l'organisation d'un théorème (contexte, hypothèse, conclusion) ou de celle du dicours complet incluant sa démonstration. Quel que soit le niveau d'intérêt du mathématicien, la phase de détection ① consiste à alerter le traitement cognitif du fait qu'un élément supposé intéressant intervient, l'identification ② analyse les données à considérer afin d'en valider l'intérêt et leur attribuer un sens qui déclenche une (ré-)action ③ adaptée.

Pour assister l'usager dans ses manipulations en langage mathématique usuel, le système doit disposer de capacités d'**association** et de **reconnaissance**. L'*association* consiste à faire le lien et à comparer des formes proches d'un même contexte. La *reconnaissance* consiste à **identifier** un objet comme une instance conceptuelle d'un autre objet. Ces capacités s'exercent principalement à un niveau sémantique dans l'usage mathématique. Ce niveau de compréhension est d'ailleurs essentiel, puisqu'il correspond à l'intérêt que porte le mathématicien aux dits objets.

Illustrons la généralité du modèle que nous proposons, sachant que l'agent peut être humain ou non :

- pour la **surveillance** de système : la réaction③ est une alarme, l'identification② de sa provenance peut être sommaire, et la capacité de détection① est déterminante ;
- la **reconnaissance de formes** cherche à détecter① un contour possible, et tente de l'identifier② comme objet afin de l'intégrer dans la scène à construire③ ;
- pour la **réparation** de disfonctionnement dans un système industriel ou biologique : la détection① peut être donnée par une alarme (symptome), le savoir-faire consiste alors à identifier② la cause (diagnostic) puis à proposer une réparation adaptée③ (ordonnance) ;
- le **déverminage** est l'équivalent de la réparation pour un système informatique. Selon le problème, chacune des trois phases peut être difficile et interagir avec les autres : la détection① varie du symptome le plus évident à l'effet le plus sournois (quand ce qui est pensé est différent de la spécification exprimée). Une erreur étant détectée, son origine doit alors être identifiée② dans le programme afin d'être corrigée③ ;
- pour la **résolution de problème** (par un moteur d'inférence ou par la pensée dirigée d'un mathématicien) : la détection① de l'état du problème déclenche une recherche heuristique sur la direction à suivre, l'identification② des règles applicables détermine alors le choix d'une règle à appliquer③.

4.2.2. Application à la comparaison de formules

Afin d'illustrer ces capacités d'association et de reconnaissance, considérons l'exemple suivant extrait d'une article d'automatique [Slotine 86]. Son auteur introduit d'abord un modèle de système dynamique qu'il décrit par la formule (1) :

$$x^{(n)}(t) = f(X; t) + b(X; t)\, u(t) + d(t) \quad \ldots\ldots (1)$$

Puis, il l'utilise dans des exemples, dont le premier est :

$$\ddot{x} = f + u \quad \ldots\ldots (2)$$

Comment un système peut-il être capable d'exploiter le modèle précédemment introduit sachant qu'on lui donne la formule (2) comme un exemple issu de (1) ? Faut-il pour cela que l'usager décrive exhaustivement les liens d'instanciation entre les divers constituants ?

Imaginons quel serait, en terme de manipulations élémentaires, le moyen le plus simple de procéder dans un système utilisant pour cela des menus :

① utiliser la commande "instancier la formule (1)" ; le système affiche alors la liste des constituants de (1) ;

② indiquer que b est l'identité ;

③ indiquer que d est nulle ;

④ indiquer que n vaut 2 ;

⑤ la validation de cette instanciation doit alors procéder à des simplifications automatiques, mais il faut au minimum indiquer de noter les fonctions sans leurs arguments pour aboutir à la formule (2).

Une telle façon de procéder s'avèrerait vite pesante. Nous préférons indiquer implicitement ces transformations à partir de la saisie de la formule (2). Cette façon de procéder correspond mieux à l'inclination du mathématicien (facteurs humains) tout en lui permettant d'indiquer de façon concise les instanciations et les nouveaux critères de notation.

La tâche du système consiste alors à utiliser ses capacités de reconnaissance et d'association pour réduire les différences entre (1) et (2). Le premier acte consiste à s'affranchir des caractéristique syntaxiques en remarquant que les procédés de notation choisis diffèrent. Une fois neutralisée cette différence, le problème consiste à comparer (1') avec la formule (2) :

$$x^{(n)} = f + b\, u + d \quad \ldots\ldots (1')$$

$$\ddot{x} = f + u \quad \ldots\ldots (2)$$

Le système doit alors considérer les structures des formules, en liaison avec les symboles utilisés pour les notations. L'analyse des correspondances et des différences donne :

- pas d'inversion de terme mais disparition de "b" et "d" ;
- comparaison de "$x^{(n)}$" et "$\ddot{x}$".

Ce procédé s'apparente à une **"reconnaissance des formes" symbolique**. La notion de terme et les connaissances élémentaires de déduction mathématique permettent de reconstituer la cause de la disparition de "b" et de "d". En particulier ici, un terme disparaît lorsqu'il devient égal à l'élément neutre de la loi correspondante. Les règles utilisées ici sont notamment : "1 x = x" ou "0 + x = x".

Les transformations plus complexes nécessitent d'autres règles. Par exemple ici, il pourrait exister des couples (b,d) tels que "b u + d = u". En l'absence d'une telle connaissance explicite, le système est conçu pour trouver des solutions "triviales". La garantie de trouver un résultat, ou de trouver "le bon" n'est pas recherchée dans un contexte d'assistance où l'utilisateur peut toujours aider à aboutir au résultat désiré.

4.2.3. Pragmatique et recueil d'informations implicites

La **connaissance des notations** et de leur usage en analyse comme en génération est déterminante dans ces mécanismes d'association et de reconnaissance. Il a fallu reconnaître que "f(**X**; t)" et "f" correspondent à deux façons de présenter un même objet. Il a fallu de même, savoir et reconnaître que "$x^{(n)}$" et "$\ddot{x}$" sont deux formes différentes d'une même notation. Cette faculté de gestion des notations est déterminante. Outre les aspects propres aux notations, elle fait intervenir des inférences sur les objets mathématiques sous-jacents. Par exemple, soit un vecteur "$\mathbf{A} \in R^{n}$", son instanciation par la phrase :

$$\text{" Posons } \mathbf{A} = \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} \text{"}$$

permet de préciser les composantes de ce vecteur, et surtout d'inférer que "n = 4".

Cette façon de procéder permet d'effectuer simultanément des choix implicites en inhibant les règles de cohérence des choix de notation par défaut. Les composantes choisies ici par défaut auraient été de la forme "a_i" (dès que $i \geq 2$). Le choix d'un symbole et de ses annotations graphiques permet en effet de suggérer la nature mathématique de l'objet associé ainsi que son usage dans le problème et ses liens avec les autres objets. Ainsi, effectuer un changement de variable pour remplacer "3 a - x_5" propose par défaut "y_5" si "y" est un symbole non encore employé.

Le rôle des fonctionnalités de perception est de **comprendre le langage mathématique** en assurant le lien entre une notation et la définition mathématique des objets correspondants. Par exemple, dans ses commentaires de la formule (1) précédente, l'auteur introduit la notation :

$$\mathbf{X} = [x\ \dot{x} \ldots\ x^{(n-1)}]^T$$

Un système convivial intelligent doit pouvoir reconnaître et interpréter cette notation à base de points de suspensions... Il doit identifier la suite considérée, sa longueur et son terme générique. De plus, lorsque l'auteur introduit les expressions :

$$\text{" } \mathbf{X}_d = \text{" puis " } \tilde{\mathbf{X}} = \mathbf{X} - \mathbf{X}_d \text{ "}$$

le système doit pouvoir inférer leur valeur symbolique et la notation associée, afin de compléter par exemple le membre droit de " $\mathbf{X}_d$ = ". Ces opérations requièrent des capacités de **création d'objets** mathématiques à partir de leur notation, tout cela grâce à des **critères d'association** au niveau des notations et des symboles employés.

Un grand nombre de particularités de notation peuvent être ainsi étudiées. Leurs caractéristiques principales concernent des facteurs humains d'association, de lisibilité et de concision. Les notations de matrices triangulaires, diagonales, par blocs, ou dont les lignes ou colonnes ont une forme particulière, fournissent à elles seules un sujet d'étude complet. Ces notations matricielles employent en particulier des notations à base de **points de suspensions**, qui sont des plus utiles dans l'usage des mathématiques.

La compréhension des points de suspensions est, en effet, l'un des points clés d'un système basé sur la pratique du mathématicien. Qui donc se priverait de la facilité d'écrire :

$$\mathbf{X} = [x\ \dot{x} \ldots\ x^{(n-1)}]^T \text{ voire } e^x = 1 + x + \frac{x^2}{2!} + \ldots$$

Le problème général est de reconnaître un terme générique à partir d'instances. Outre les facultés de **généralisation** de "0 1 2 ..." en "n", l'association et la reconnaissance exploitent pour leurs déductions tout un jeu d'équivalences triviales pour l'introduction de "facteurs cachés" comme :

$$x^0 = 0! = 1! = +1 = 1 \quad \text{ou} \quad -x^1 = \frac{x}{-1} = -(x) = (-1)\,x = -x.$$

Ces **fonctionnalités de perception** mettent en œuvre des mécanismes spécifiques d'association et de reconnaissance. En cas d'échec de ces derniers, le mathématicien dispose toujours d'une façon plus explicite de procéder. Le niveau minimal requis pour la pratique confortable du langage mathématique et son assistance est toutefois important en terme de fonctionnalités informatiques.

4.3. LE GUIDAGE DES TACHES

Notre approche est bien sûr complémentaire des techniques classiques de démonstration de théorèmes ou de calcul. En effet, dans des situations bien maitrisées, les tâches résultant de la planification locale peuvent être assistées dans leur réalisation :

- exécution d'une tâche de calcul ;
- réalisation d'une tâche de plus haut niveau conceptuel (changement de variable dans une intégrale avec calcul des nouvelles bornes...) ;
- recherche automatique de solution d'une tâche ;
- essai de tactiques de recherche d'une solution.

Dans une perspective d'assistance, la difficulté principale est toutefois **la définition et le guidage de tâches par l'usager**. Par exemple, lors de la manipulation d'une formule une tâche à effectuer a dû être décomposée en plusieurs opérations alors que son intention sémantique était claire. Cette tâche doit alors pouvoir être directement associée à une opération implantée dans le système.

La définition de cette nouvelle opération peut alors être introduite de plusieurs manières, notamment :

- grâce à un apprentissage par observation ;
- par spécification de l'action par l'usager (simultanée ou ultérieure) ;
- par spécification amenant à une extension de l'implantation du système (à éviter).

L'usager a donc besoin d'un système ouvert. Il est par exemple judicieux de pouvoir insérer des commentaires sur une difficulté rencontrée lors de la construction d'une preuve et de sa démonstration. L'enjeu essentiel est alors de pouvoir **définir facilement** de nouveaux plans à l'aide d'un langage simple mais expressif, et de bénéficier ainsi de l'usage expérimental du système par le mathématicien.

$S_n = \sum_{1\le j<k\le n} \frac{1}{k-j}$

The normal way to evaluate a double sum is to sum first on j or first on k, so let's explore both options.

A

$$
\begin{aligned}
S_n &= \sum_{1\le k\le n} \sum_{1\le j<k} \frac{1}{k-j} && \text{1}\\
&= \sum_{1\le k\le n} \sum_{1\le k-j<k} \frac{1}{j} && \text{2 summing first on j}\\
&= \sum_{1\le k\le n} \sum_{0<j\le k-1} \frac{1}{j} && \text{3 replacing j by } k-j;\ \text{simplifying the bounds on j}\\
&= \sum_{1\le k\le n} H_{k-1} && \text{4 by (2.13), the definition of } H_{k-1}\\
&= \sum_{1\le k+1\le n} H_k && \text{5 replacing k by } k+1;\ \text{simplifying the bounds on k}\\
&= \sum_{0\le k<n} H_k . &&
\end{aligned}
$$

macro-opération

Alas! We don't know how to get a sum of harmonic numbers into closed form. 6 *Get out the whip.*

B

1 If we try summing first the other way, we get

$$
\begin{aligned}
S_n &= \sum_{1\le j\le n} \sum_{j<k\le n} \frac{1}{k-j} && \text{2 summing first on k}\\
&= \sum_{1\le j\le n} \sum_{j<k+j\le n} \frac{1}{k} && \text{3 replacing k by } k+j;\ \text{simplifying the bounds on k}\\
&= \sum_{1\le j\le n} \sum_{0<k\le n-j} \frac{1}{k} && \text{4 by (2.13), the definition of } H_{n-j}\\
&= \sum_{1\le j\le n} H_{n-j} && \text{5 replacing j by } n-j;\ \text{simplifying the bounds on j}\\
&= \sum_{1\le n-j\le n} H_j &&\\
&= \sum_{0\le j<n} H_j . &&
\end{aligned}
$$

We're back at the same impasse. 6

But there's *another* way to proceed, if we replace k by $k+j$ *before* deciding to reduce S_n to a sum of sums:

C

$$
\begin{aligned}
S_n &= \sum_{1\le j<k\le n} \frac{1}{k-j} && \text{recopying the given sum}\\
&= \sum_{1\le j<k+j\le n} \frac{1}{k} && \text{replacing k by } k+j \quad \text{B3}
\end{aligned}
$$

$$
\begin{aligned}
&= \sum_{1\le k\le n} \sum_{1\le j\le n-k} \frac{1}{k} && \text{summing first on j} \quad \text{A2}\\
&= \sum_{1\le k\le n} \frac{n-k}{k} && \text{the sum on j is trivial}\\
&= \sum_{1\le k\le n} \frac{n}{k} - \sum_{1\le k\le n} 1 && \text{by the associative law}\\
&= n\Big(\sum_{1\le k\le n} \frac{1}{k}\Big) - n && \text{by gosh}\\
&= nH_n - n . && \text{by (2.13), the definition of } H_n \quad \text{A4}
\end{aligned}
$$

It was smart to say $k \le n$ instead of $k \le n-1$ in this derivation. Simple bounds save energy.

Aha! We've found S_n. Combining this with the false starts we made gives us a further identity as a bonus:

$$\sum_{0\le k\le n} H_k = nH_n - n . \tag{2.36}$$

abstraction du passage difficile de C

We can understand the trick that worked here in two ways, one algebraic and one geometric. (1) Algebraically, if we have a double sum whose terms involve $k+f(j)$, where f is an arbitrary function, this example indicates that it's a good idea to try replacing k by $k-f(j)$ and summing on j. (2) Geometrically, we can look at this particular sum S_n as follows, in the case $n = 4$:

représentation intuitive heuristiquement intéressante

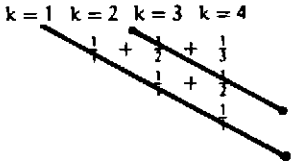


figure 2.7

Our first attempts, summing first on j (by columns) or on k (by rows), gave us $H_1 + H_2 + H_3 = H_3 + H_2 + H_1$. The winning idea was essentially to sum by diagonals, getting $\frac{3}{1} + \frac{2}{2} + \frac{1}{3}$.

2.5 GENERAL METHODS

Now let's consolidate what we've learned, by looking at a single example from several different angles. On the next few pages we're going to try to find a closed form for the sum of the first n squares, which we'll call $\square_n$:

$$\square_n = \sum_{0\le k\le n} k^2, \qquad \text{for } n \ge 0. \tag{2.37}$$

We'll see that there are at least seven different ways to solve this problem, and in the process we'll learn useful strategies for attacking sums in general.

Le guidage de tâches par l'usager est un point essentiel du raisonnement de résolution de problème. L'exemple figure 2.7 [Graham 89, p40] montre que seule une description par tâches permet le raisonnement de l'auteur dans les tentatives A, B et C.

Les opérations mathématiques élémentaires appliquées sont indiquées à droite. Nous avons regroupé sur la figure une première macro-opération de remplacement et simplification. Toutefois cela ne suffit pas pour expliquer le guidage de l'auteur lors de la tentative A, guidage qui est réutilisé pour la tentative B.

Pour être réutilisé, le guidage de la tentative A doit être exprimé comme :

1) choisir j plutôt que k ;
2) sommer sur j ;
3) se ramener à 1/j ;
4) appliquer la définition de $H_{f(k)}$;
5) se ramener à H_k ;
6) aviser.

A ce niveau de description conceptuel, le guidage de la tentative B se déduit par permutation de k et j. La tentative B consiste alors en une application directe du guidage de la tentative A. La condition à respecter pour l'exécution est de savoir appliquer ce guidage en terme d'opérations sur les formules.

Pour ce guidage, la macro-opération de remplacement et simplification est décrite à un niveau encore plus conceptuel. Elle se décrit par le but du remplacement : se ramener à 1/j. Cela implique que les opérations de l'usager ne soient plus décrites simplement par des opérations élémentaires, mais par des opérations conceptuellement plus sophistiquées décrites par leur but.

Les tentatives A et B ont un caractère exploratoire important pour le mathématicien. La tentative C récupère une sous-tâche de B avant de relancer le début de A.

Cet exemple illustre l'intérêt de la réutilisation temporaire, en cours de démonstration, d'un guidage effectué par le mathématicien (de A vers B). La figure 2.7 indique aussi après la solution C quels sont les points importants retenus pour une réutilisation plus générale : seul le passage difficile du guidage est retenu ; de plus, il est généralisé afin de s'appliquer à une classe de formules non restreinte.

Cet exemple illustre par ailleurs que la généralisation et la réutilisation de preuves ne peuvent usuellement s'effectuer à partir d'opérations mathématiques élémentaires comme cela est trop souvent proposé.

Notre approche d'assistance à la construction de preuves propose de définir expérimentalement de telles tâches à partir d'opérations mathématiques élémentaires. Le mathématicien pourrait alors décrire la construction de sa preuve directement par les tâches à effectuer. Une telle description associée à la preuve permettrait la mise en œuvre de raisonnements sophistiqués (induction, analogie, etc.) en vue de sa réutilisation.

4.4. LA PRODUCTION D'UNE PREUVE COMPLEXE

Notre objectif pratique d'assistance à la construction de preuves et démonstrations a permis de formuler des besoins, propositions et fonctionnalités pour cela. Nous le concluons maintenant sur un exemple de faisabilité d'une preuve complexe [Graham 89, p 174-177].

Les états de la preuve à granularité variable y sont décrits figure 2.8. Notons en particulier au niveau R2 une réutilisation dynamique de la preuve de B par transformation de problème pour obtenir A.

La présentation de cette preuve dans la démonstration introduit des égalités non prises en compte comme :

$$S = mA - (m-n)B = \ldots \quad (1)$$

Dans notre approche, les actions de nommer et désigner sont considérées comme externes à la preuve, quoique indispensables aux manipulations. Par exemple, la forme (1) ci-dessus contient un rappel de l'état précédent au niveau R1, rappel indispensable au bon suivi de la démonstration. Usuellement, le premier état d'un segment est décrit par son rôle dans le problème (la surface à calculer...) et désigné à un symbole (S=). La forme (1) est donc un résumé présentant l'état actuel du segment de preuve à ce niveau.

Problem 2: From the literature of sorting.

Our next sum appeared way back in ancient times (the early 1970s) before people were fluent with binomial coefficients. A paper that introduced an improved merging technique [165] concludes with the following remarks: "It can be shown that the expected number of saved transfers ... is given by the expression

$$T = \sum_{r=0}^{n} r\frac{{}_{m-r-1}C_{n-r-1}}{{}_{m}C_{n}}$$

le problème

Here m and n are as defined above, and ${}_mC_n$ is the symbol for the number of combinations of m objects taken n at a time. ... The author is grateful to the referee for reducing a more complex equation for expected transfers saved to the form given here."

We'll see that this is definitely not a final answer to the author's problem. It's not even a midterm answer.

First we should translate the sum into something we can work with; the ghastly notation ${}_{m-r-1}C_{n-r-1}$ is enough to stop anybody, save the enthusiastic referee (please). In our language we'd write

$$T = \sum_{k=0}^{n} k\binom{m-k-1}{n-k-1}\Big/\binom{m}{n}, \qquad \text{integers } m > n \geq 0. \quad \text{I}$$

The binomial coefficient in the denominator doesn't involve the index of summation, so we can remove it and work with the new sum

$$S = \sum_{k=0}^{n} k\binom{m-k-1}{n-k-1} \quad 1$$

What next? The index of summation appears in the upper index of the binomial coefficient but not in the lower index. So if the other k weren't there, we could massage the sum and apply summation on the upper index (5.10). With the extra k, though, we can't. If we could somehow absorb that k into the binomial coefficient, using one of our absorption identities, we could then sum on the upper index. Unfortunately those identities don't work here. But if the k were instead $m-k$, we could use absorption identity (5.6):

$$(m-k)\binom{n-k-1}{n-k-1} = (m-n)\binom{m-k}{n-k}.$$

So here's the key: We'll rewrite k as $m-(m-k)$ and split the sum S into two sums:

$$\sum_{k=0}^{n} k\binom{m-k-1}{n-k-1} = \sum_{k=0}^{n} (m-(m-k))\binom{m-k-1}{n-k-1} \quad 2$$

$$= \sum_{k=0}^{n} m\binom{m-k-1}{n-k-1} - \sum_{k=0}^{n} (m-k)\binom{m-k-1}{n-k-1} \quad 3$$

$$= m\sum_{k=0}^{n}\binom{m-k-1}{n-k-1} - \sum_{k=0}^{n}(m-n)\binom{m-k}{n-k} \quad 4$$

$$= mA - (m-n)B, \quad 5$$

where

$$A = \sum_{k=0}^{n}\binom{m-k-1}{n-k-1}, \qquad B = \sum_{k=0}^{n}\binom{m-k}{n-k}.$$

The sums A and B that remain are none other than our old friends in which the upper index varies while the lower index stays fixed. Let's do B first, because it looks simpler. A little bit of massaging is enough to make the summand match the left side of (5.10):

$$B = \sum_{0\leq k\leq n}\binom{m-k}{n-k} \quad a$$

$$= \sum_{0\leq m-k\leq n}\binom{m-(m-k)}{n-n} \quad b$$

$$= \sum_{m-n\leq k\leq m}\binom{k}{m-n} \quad c$$

$$= \sum_{0\leq k\leq m}\binom{k}{m-n}. \quad d$$

In the last step we've included the terms with $0 \leq k < m-n$ in the sum; they're all zero, because the upper index is less than the lower. Now we sum on the upper index, using (5.10), and get

$$B = \sum_{0\leq k\leq m}\binom{k}{m-n} = \binom{m+1}{m-n+1}. \quad e$$

The other sum A is the same, but with m replaced by $m-1$. Hence we have a closed form for the given sum S, which can be further simplified:

$$S = mA - (m-n)B = m\binom{m}{m-n} - (m-n)\binom{m+1}{m-n+1} \quad 6$$

$$= \left(m - (m-n)\frac{m+1}{m-n+1}\right)\binom{m}{m-n} \quad 7$$

$$= \left(\frac{n}{m-n+1}\right)\binom{m}{m-n} \quad 8$$

And this gives us a closed form for the original sum:

$$T = S\Big/\binom{m}{n}$$

$$\text{II} = \frac{n}{m-n+1}\binom{m}{m-n}\Big/\binom{m}{n}$$

$$\text{III} = \frac{n}{m-n+1}$$

Even the referee can't simplify this.

Again we use a small case to check the answer. When $m=4$ and $n=2$, we have

$$T = 0\cdot\binom{3}{1}/\binom{4}{2} + 1\cdot\binom{2}{1}/\binom{4}{2} + 2\cdot\binom{1}{1}/\binom{4}{2} = 0 + \tfrac{2}{6} + \tfrac{2}{6} = \tfrac{2}{3},$$

which agrees with our formula $2/(4-2+1)$.

niveau R0 I
niveau R1 1
niveau R2 a
} états de la preuve à granularité variable

: tâches vers opérations

Table 174 The top ten binomial coefficient identities.

$\binom{n}{k} = \frac{n!}{k!(n-k)!}$,	integers $n \geq k \geq 0$.	*factorial expansion*
$\binom{n}{k} = \binom{n}{n-k}$,	integer $n \geq 0$, integer k.	*symmetry*
$\binom{r}{k} = \frac{r}{k}\binom{r-1}{k-1}$, 5.6	integer $k \neq 0$.	*absorption/extraction*
$\binom{r}{k} = \binom{r-1}{k} + \binom{r-1}{k-1}$,	integer k.	*addition/induction*
$\binom{r}{k} = (-1)^k\binom{k-r-1}{k}$,	integer k.	*upper negation*
$\binom{r}{m}\binom{m}{k} = \binom{r}{k}\binom{r-k}{m-k}$,	integers m, k.	*trinomial revision*
$\sum_k \binom{r}{k}x^k y^{r-k} = (x+y)^r$,	integer $r \geq 0$, or $\lvert x/y\rvert < 1$.	*binomial theorem*
$\sum_{k\leq n}\binom{r+k}{k} = \binom{r+n+1}{n}$,	integer n.	*parallel summation*
$\sum_{0\leq k\leq n}\binom{k}{m} = \binom{n+1}{m+1}$, 5.10	integers $m, n \geq 0$.	*upper summation*
$\sum_k \binom{r}{k}\binom{s}{n-k} = \binom{r+s}{n}$,	integer n.	*Vandermonde convolution*

figure 2.8

Le texte de ce problème 2 présente deux particularités intéressantes :
- il aborde le problème en insérant sa résolution entre une tâche de formulation initiale et une tâche de vérification du résultat ;
- il verbalise le raisonnement du mathématicien lors de la résolution.

L'analyse du texte de la démonstration montre des passages :
- de perception ;
- d'action (tâches mathématiques à exécuter) ;
- de gestion des hypothèses et du centre d'intérêt (Et ensuite ?, Si nous pouvions, Donc, etc.).

Le raisonnement initial comprend deux niveaux de planification dont quelques tâches sont tracées figure 2.8. Elles illustrent notamment comment certaines opérations élémentaires du mathématicien peuvent être décrites de façon plus conceptuelle. Elle montre par exemple la genèse et le développement de la tâche :
- absorber ce k dans le coefficient du binôme, utilisant une de nos identités d'absorbtion ;
- utiliser l'identité d'absorbtion (5.6) ;
- appliquer $r\binom{r-1}{k-1} = k\binom{r}{k}$ avec r—>m-k et k —>m-n à tel emplacement.

Le raisonnement utilisé par le mathématicien est **dirigé par l'analyse des données et l'exploitation des identités disponibles** sur les coefficients du binôme, identités qui sont regroupées en tableau. Ce type de raisonnement est très net ici, et nous proposons d'en modéliser quelques mécanismes partie III en le considérant comme une :

expertise d'analyse et d'utilisation de formules

Une même expertise de perception s'applique ici sur les identités et sur les formules du problème. Elle inclut des propositions comme "l'indice de la sommation apparaît dans l'argument supérieur du coefficient du binôme mais pas dans l'argument inférieur". La seule identité directement applicable en fonction de ces caractéristiques est alors (5.10).

Le mathématicien utilise alors un raisonnement fin-moyens pour tenter de l'appliquer. Il utilise pour cela une expertise d'utilisation de formules qui peut :
- relancer un raisonnement sur des problèmes voisins, comme c'est le cas ici ;
- effectuer un aménagement par addition-multiplication-instanciation comme dans l'application de l'identité (5.6).

Il est envisageable de modéliser cette expertise d'analyse et d'utilisation de formules, valable pour des manipulations de formules dans des domaine mathématiques variés. Elle pourrait servir notamment pour assister les tâches élémentaires du mathématicien à partir des **formules exprimées sous leur forme externe par la mathématicien.**

5. PROBLEMATIQUE DE L'ASSISTANCE AU TRAVAIL

5.1. UNE ASSISTANCE ADAPTEE A L'ACTIVITE

5.1.1. L'étude expérimentale de l'activité mathématique

> Après une période où l'accent était mis sur les résultats et performances des programmes, les travaux récents en Intelligence Artificielle mettent d'avantage l'accent sur la modélisation des comportements humains, sur l'étude des connaissances et des modes de raisonnement utilisés par l'homme ; cela semble en effet une voie intéressante, sinon nécessaire, pour :
>
> - d'une part faciliter, voire permettre, le dialogue entre un système et un utilisateur humain dans le cadre d'activités "assistées" par ordinateur, activités professionnelles ou de formation.
>
> - d'autre part pouvoir bénéficier du savoir des experts pour la réalisation de systèmes basés sur les connaissances.

Monique Baron [Baron 85]

L'analyse de l'activité mathématique a permis d'entrevoir la complexité du rôle du mathématicien et comprendre pourquoi aucun système informatique ne peut prétendre actuellement utiliser ces méthodes humaines sur une échelle réaliste. Elle recadre les approches informatiques basées essentiellement sur une automatisation du raisonnement en les situant au service de l'**assistance au mathématicien.**

Nous proposons une **étude expérimentale** de l'activité mathématique sur trois plans :
- étude du mathématicien et de son activité telle quelle est pratiquée hors informatique ;
- conception et développement d'un système d'assistance ;
- utilisation du système par les mathématiciens.

Le but d'une étude expérimentale est d'aller au delà de la problématique actuelle au niveau du détail et de la précision des connaissances, au niveau de leur quantité, et aussi au niveau de leur dynamique d'utilisation. Elle cherche à faire un progrès qualitatif dans les choses exprimables, et donc susceptibles d'être comprises.

La partie I a montré l'intérêt d'analyses complémentaires pour étudier l'activité du mathématicien indépendamment de tout système informatique. L'expérimentation escomptée au niveau de la conception d'un système adopte une approche de type maquettage et prototypage. Notre approche est axée sur les moyens d'expression et sur l'interaction d'utilisation, avant de chercher à s'intéresser aux raisonnements plus complexes. Son objectif est d'améliorer le système, et la connaissance de l'activité mathématique, via les besoins exprimés lors de l'utilisation du système par les mathématiciens.

Une étude expérimentale permet de :
- **capturer les situations stéréotypées** où des actions sont conditionnées en fonction des objets présents, mais qu'on ne peut exprimer ni répertorier faute d'un langage (et d'un besoin) permettant d'exprimer la situation.
- **favoriser l'émergence d'invariants** que va capturer le mathématicien s'il en a la possibilité, afin de faciliter un aspect répétitif de sa pratique courante. Nous comptons sur l'émergence de concepts utiles pour la réutilisation de données à partir des besoins que les mathématiciens formulent pour résoudre leurs problèmes.
- **effectuer un recueil de données** indispensable pour mieux comprendre les mécanismes mathématiques : décrire comment on conflue vers un théorème à appliquer depuis son "voisinage" ou quelle construction est à effectuer pour appliquer le théorème de Thalès, identifier et tester des heuristiques dans des zones apparentées à du calcul formel, mieux connaître les opérations résultant d'interventions intuitives, etc.

5.1.2. Un environnement de travail naturel

Cette approche expérimentale des mathématiques n'est envisageable que si le système proposé au mathématicien est suffisamment proche de son activité mathématique usuelle. Notre approche consiste à reproduire et à enrichir l'**environnement "naturel" du mathématicien** (figure 2.9).

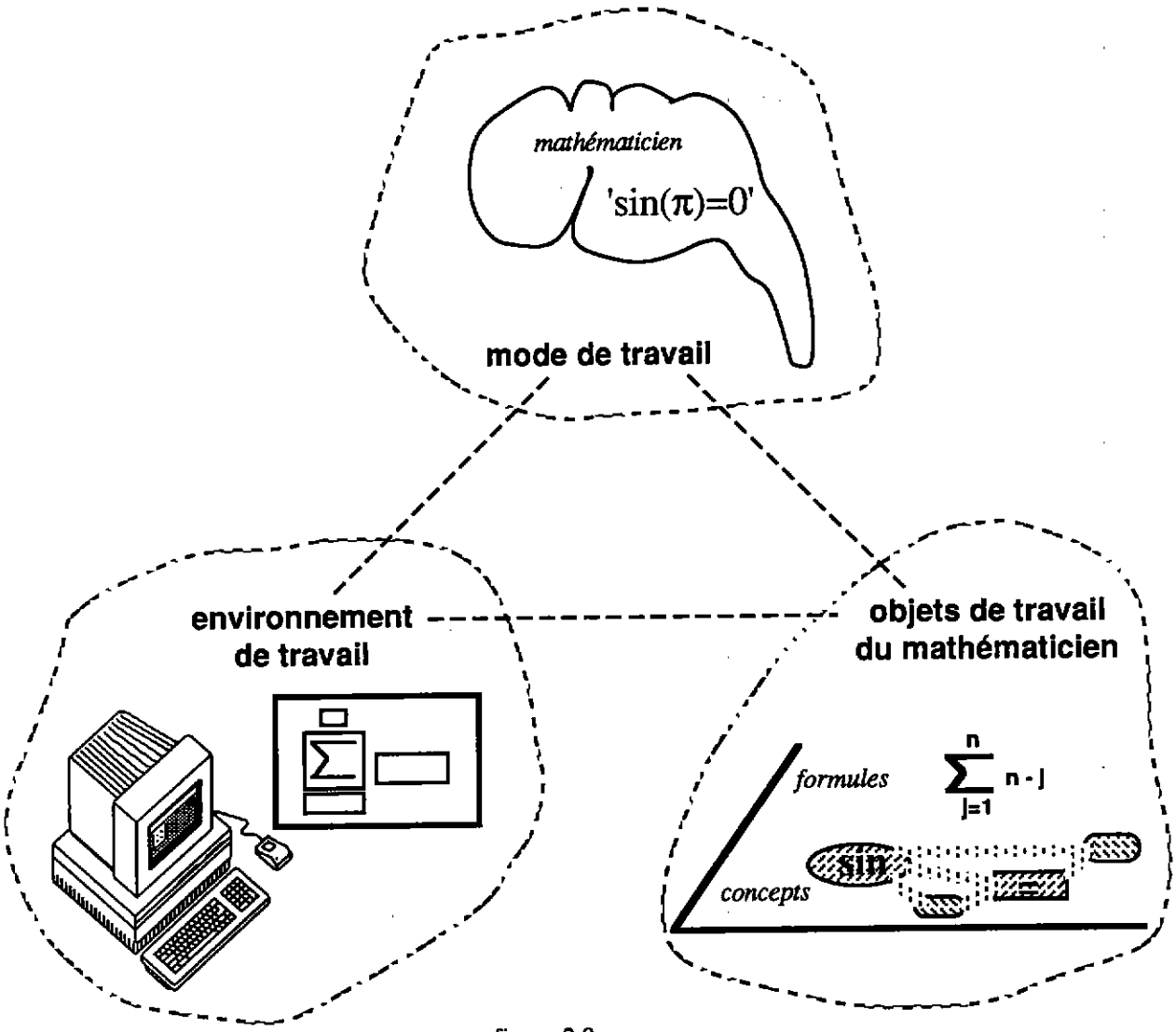


figure 2.9

Les objectifs de manipulation retenus se décomposent en trois niveaux :

- le **mode de travail**, qui se veut "naturel" dans la pratique des manipulations et s'appuie sur des connaissances (para-)mathématiques ;
- l'**environnement de travail**, en particulier l'interface de manipulation ;
- les **objets de travail**, qui sont les symboles, formules, définitions et concepts mathématiques du domaine.

Pour fixer les connaissances manipulées, nous avons choisi de rester près de la ligne de démarcation du Langage Mathématique afin de bénéficier des connaissances explicites. Avec l'adjonction de connaissances de traitement, on dispose à ce niveau d'un jeu de **paramètres de décision** suffisamment complet pour que les paramètres complémentaires demandés au mathématicien soient modérés. Le système est alors en mesure d'effectuer des tâches élémentaires.

Pour définir un langage de commande adéquat, **la formulation ne doit pas perturber la tâche mathématique principale du mathématicien**. Cela signifie que la formulation doit :

- s'exprimer avec des concepts familiers au mathématicien ;
- faire oublier l'intermédiaire informatique ;
- être d'autant plus facile que sa fréquence d'utilisation est grande.

5.1.3. Les besoins de définition d'un modèle d'assistance

La **simulation-communication** est un terme employé par J.L. Lebahar pour désigner l'interaction entre l'architecte et son problème [Quintrand 85]. Ce processus de communication produit de nouvelles informations pour l'homme et la machine, en stimulant notamment la créativité du concepteur. Le bon fonctionnement du processus de simulation-communication est à la base du projet TECTON dont le but est de réaliser un système de CAO intelligent qui intègre une partie du savoir architectural [Hanrot 88]. Leurs auteurs développent ainsi les besoins (p388) :

- Que la communication entre les deux acteurs (homme et machine) soit effective, notamment pour la compréhension aisée des codes utilisés.
- Que l'effet de la communication, transfert d'information, soit productif de nouvelles informations chez les deux acteurs. A chaque étape du processus, il y a non seulement "héritage" de valeurs de l'étape précédente, mais aussi production d'un nouvel état des problèmes par transformation de l'information.

Pour satisfaire la première condition, le système doit offrir à l'utilisateur des moyens de dialogue exploitant le langage de l'architecte.
Pour satisfaire la seconde le système doit disposer de fonctions de manipulation et de restitution de l'information intégrant une part du "savoir du concepteur" pouvant aider ou orienter le processus de conception.
En d'autres termes le système doit être "intelligent". Ses fonctionnalités peuvent être mises en œuvre aujourd'hui grâce aux techniques de programmation de l'intelligence artificielle.

Nous adoptons cette démarche pour l'**assistance au mathématicien**. Notons en particulier que cette approche qui utilise des techniques "intelligentes" à des fins d'assistance, se démarque d'une approche "système-expert" qui tend à résoudre des problèmes mathématiques. Pour un usage effectif, cette dernière est limitée à des problèmes très ciblés présentant un intérêt ponctuel. Voyons, dès lors, les principes d'assistance qui permettent un processus de simulation-communication.

Le rôle de l'homme dans le processus de Conception Assistée par Ordinateur est souvent au service de la logique du système. Pourtant l'assistance à la conception n'est efficace que lorsqu'elle est au service des processus créatifs. Le facteur déterminant de la simulation calculatoire est l'opération de **modélisation**. « On ne sait calculer que ce que l'on sait modéliser » déclarent les spécialistes en simulation. Mais les modèles portent le plus souvent sur des critères de simplicité calculatoire. Ils restent alors étrangers aux opérations humaines "naturelles". Les critiques des modèles disponibles sont :

- les modèles proposés sont étrangers au (ou limitent le) modèle mental du concepteur qui ne peut employer efficacement son savoir-faire qui lui confère sa supériorité ;
- les opérations désirées pour la tâche de l'homme ne correspondent pas à des opérations réalisables sur le modèle ;
- beaucoup de paramètres sont liés au fonctionnement du système mettant en œuvre le modèle plutôt qu'au problème lui-même ou à sa présentation ;
- le contrôle du modèle s'effectue par des manipulations très différentes de celles réalisables sur l'objet qui est modélisé ;
- les objets modélisés sont indissociés de la structure qui leur sert de représentation ;
- un même objet conceptuel est représenté par plusieurs structures de données non corrélées ;
- l'altération d'un des paramètres du modèle oblige à recommencer tous les calculs.

5.1.4. Un modèle d'assistance adapté à l'activité

Vouloir représenter une activité de conception comme un problème unique qui consiste à trouver une solution relève d'une abstraction trompeuse. Elle consiste plutôt à **développer un processus de conception**, souvent de façon incrémentale, dont le produit n'est qu'un aboutissement temporaire. Un ensemble de produits auxiliaires (problème repensé, preuve, démonstration, méthode, concepts, lemmes...) sont issus de ce processus et doivent être exploités.

La **flexibilité de mise en œuvre** est la clé de toute validation. C'est donc l'objectif prioritaire d'un système d'assistance. Il faut pouvoir créer rapidement et aisément un modèle, automatiser sa mise en œuvre informatique, visualiser ses résultats, oublier les contraintes informatiques, agir directement sur le modèle, l'étendre et l'affiner, en changer radicalement lorsqu'il se révèle insuffisant...

Ces contraintes entraînent que :

l'adéquation du modèle à l'activité "naturelle" est primordiale

Cela met à la disposition de l'usager des modèles adaptés à l'exploitation de ses connaissances et de ses talents créatifs. L'intégration d'un savoir-faire élémentaire pour tout le cycle de conception permet de traiter et réutiliser l'information, ce qui minimise l'apport demandé à l'usager. Le processus de simulation-communication repose alors essentiellement sur des techniques d'interaction. Nous nous proposons de nous inspirer pour cela des paradigmes d'Engagement direct et de Visualisation conceptuelle.

Nous essayons d'appliquer à deux niveaux cette recherche d'un **modèle adapté à l'activité humaine "naturelle"** :

- au niveau de l'usager et de sa perception d'un système mathématique : nous pensons qu'un tel modèle est le mieux adapté à l'obtention d'une solution tout en répondant aux autres besoins concernant le cycle de vie d'un "produit" mathématique ;
- au niveau du processus de conception d'un tel système, où le facteur primordial n'est plus l'utilisation de techniques informatiques, mais leur disponibilité au sein d'un modèle permettant de les exploiter au mieux, au service de l'usager.

5.2. LES PRINCIPES DE MISE EN ŒUVRE D'UN MODELE D'ASSISTANCE

5.2.1. Le paradigme d'Engagement direct pour l'interaction

Progressivement, un Génie cognitif émerge et se structure, notamment autour de l'ergonomie cognitive des interfaces [INRIA 87, Scapin 88, Falzon 89, Visser 90]. Pour évaluer un système, Bernard Senach introduit la distinction suivante [Senach 90] :

> **L'utilité** porte sur des propriétés telles que la capacité fonctionnelle, les performances du système et la qualité de l'assistance technique proposée au client.
>
> **L'utilisabilité** concerne la qualité de l'interaction homme-machine, c.-à-d. la facilité d'apprentissage et d'utilisation. (et aussi la qualité de la documentation)

Nous nous intéressons ici à l'utilisabilité d'un système. Les techniques d'**interaction homme-machine** sont d'un apport optimal pour l'exploitation des compétences humaines. Elles permettent de subvenir aux besoins cognitifs (liés au problème) et ergonomiques (liés à un système). Une interaction réussie les sublime en identifiant l'univers de description à l'univers d'arbitrage afin de proposer un *monde virtuel* dans lequel l'usager raisonne et agit "effectivement". Un premier concept dans cette voie fut celui de ***manipulation directe*** introduit par Bob Shneiderman [Shneiderman 83]. Il note que les idées centrales des systèmes qui "enchantent" leurs usagers sont :

- la visibilité des objets auxquels l'usager s'intéresse ;
- des actions rapides, réversibles et incrémentales ;
- le remplacement d'une syntaxe complexe d'un langage de commande par la manipulation directe des objets intéressants.

Dans sa présentation et son analyse du Génie Cognitif, Donald A. Norman présente une **théorie de l'action** [Norman 87]. Afin d'évaluer l'interaction, il propose le sentiment d'***engagement direct*** de l'usager pour qualifier la perception d'un système dont l'interaction est particulièrement soignée. Il cite l'exemple de certains jeux vidéo ou de simulateurs d'avions. Pour l'analyser et identifier les barrières à franchir, il décompose la séquence d'actions effectuant une tâche réelle en sept composantes symétriques, selon le schéma adapté figure 2.10[1]. L'état ❶ initial correspond à un but qui définit l'état à atteindre, comme terminer le rapport ou trouver la preuve. Ce problème posé est alors développé en un cycle d'exécution-évaluation.

---

[1] Ce schéma correspond aux actions du point de vue de l'usager. Un schéma similaire est aussi pertinent vis à vis d'un Ami qui cherche à assister les tâches élémentaires en cours d'exécution.

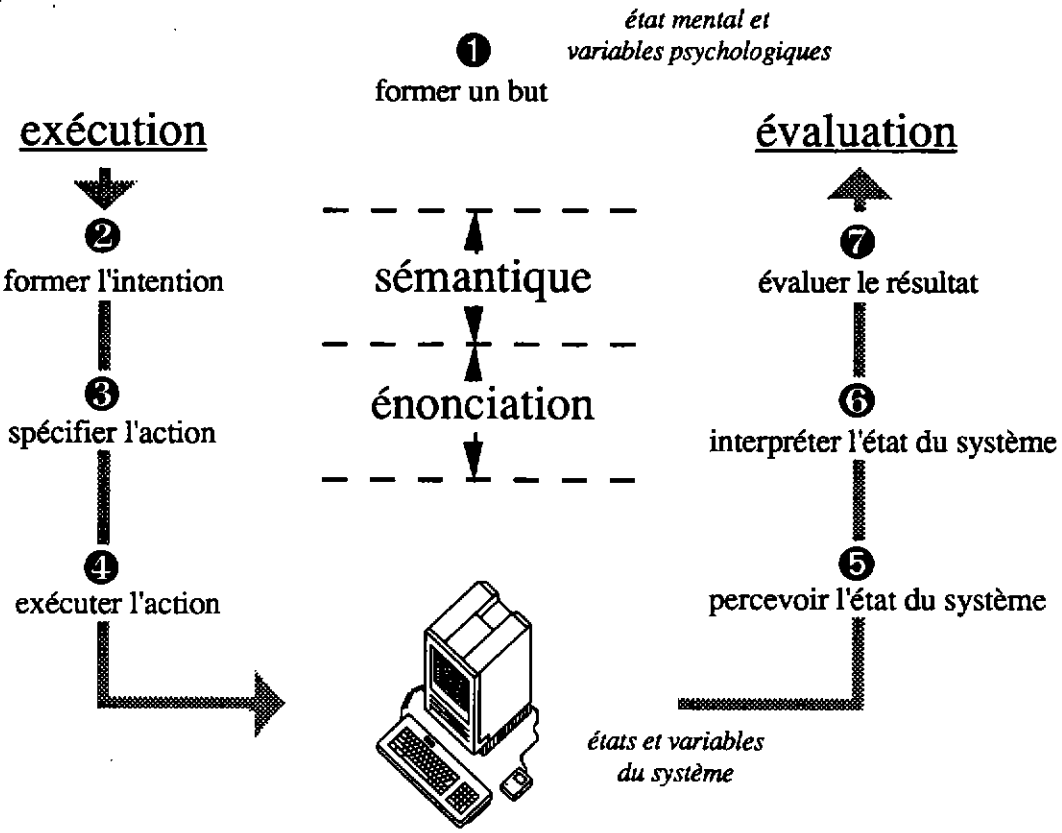


figure 2.10

Lorsque la machine s'interpose entre l'usager et sa tâche, elle oblige l'usager à raisonner avec les outils de la machine intermédiaire. D.A. Norman distingue la *barrière sémantique* qui sépare le langage dans lequel les intentions et prévisions mentales sont exprimées, et les langages des actions autorisées et des états du système. Il distingue ensuite la *barrière d'énonciation* qui reflète l'inadéquation entre les actions (autorisées) désirées et leur réalisation effective. La première barrière sépare les actions à effectuer sur la tâche, des actions qui veulent agir sur le système à partir du modèle qu'en a l'usager. La seconde sépare le sens des actions désirées sur le système de la forme qu'elles devront épouser. Enfin, une *barrière inter-référentielle* marque la distinction entre les objets manipulés en entrée, et ceux sur lesquels la sortie est perçue.

Ce modèle permet de structurer une **analyse de l'interaction**, qui met en avant les limites des systèmes existants. La barrière sémantique est parfois localement effacée lors d'actions ciblées par le système, comme démontrer, factoriser ou imprimer. Néanmoins elle demeure souvent imposante, car les systèmes ne raisonnent pas en terme de concepts et de pratique du mathématicien, et se contrôlent par des variables liées à la représentation informatique. De même, la barrière d'énonciation limite l'exécution par des langages peu adaptés et appauvrit l'évaluation qui se contente d'être interprétable. Enfin même lorsque la barrière inter-référentielle est atténuée par des objets semblables, ils permettent rarement une rétro-action sur les causes qui les ont générés, comme nous allons l'étudier pour le paradigme de visualisation conceptuelle.

5.2.2. Un module Ami comme assistant

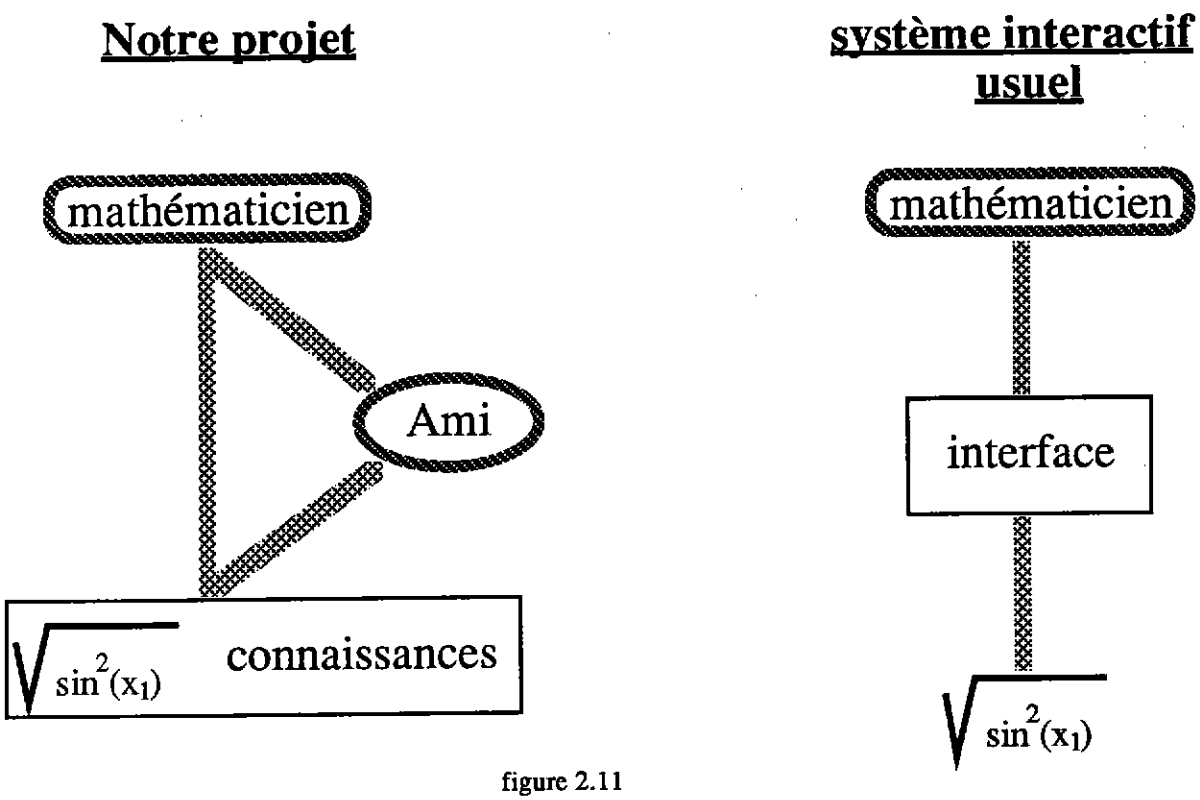


figure 2.11

Nous postulons figure 2.11 que le mathématicien doit avoir le sentiment de disposer et de **manipuler directement** ses objets mathématiques usuels. Il doit de plus percevoir l'Ami comme un assistant comprenant un langage "naturel" et capable d'effectuer des tâches élémentaires de façon ergonomique. L'Ami n'est pas alors un intermédiaire, mais un **assistant collaborateur** à la tâche partagée avec le mathématicien ! Pour l'usager, l'engagement direct place alors le problème mathématique au premier plan, laissant en annexe la façon d'utiliser le système et l'assistant.

Cette problématique de l'usager déchiré entre **"quoi"** communiquer et **"comment le faire"** est aussi valable pour le système. Toutefois, alors que la plupart des travaux axent leurs solutions sur l'idée de dialogue, nous postulons que l'efficacité d'un dialogue dépend fondamentalement de l'(inter-)action sur les objets de dialogue. Comme le note Bob Shneiderman, la visibilité des objets auxquels l'usager s'intéresse est l'une des conditions majeures de la manipulation directe. La qualité et la richesse de la visualisation concrétise alors un engagement direct réussi. Cette qualité relève d'une réflexion sur la présentation qui cherche à exprimer graphiquement des propriétés concernant des objets[2].

5.2.3. La visualisation de propriétés conceptuelles

Nous distinguons la visualisation de l'infographie (production d'images), en considérant la visualisation comme un processus actif visant à présenter au mieux des informations pertinentes à une tâche. La visualisation exploite un support d'information visuel qui sert d'interface, alors que les objets à présenter ne disposent pas forcément d'une telle représentation bidimensionnelle ou spatiale. Ces objets sont alors accessibles à l'aide de symboles relevant de la linguistique ou de la sémiologie. La visualisation d'un objet dispose alors de diverses utilisations de plusieurs langages textuels et graphiques. Cette visualisation est

---

[2] Cette réflexion rejoint celle des environnements de programmation face à l'évolution de la technologie visuelle [Ambler 89]. Elle peut bénéficier notamment des études sur la visualisation d'algorithmes et d'objets informatiques [Brown 88, Liem 89]. Elle peut s'appuyer sur les critères d'analyse d'une présentation graphique de la théorie de l'image de Jacques Bertin (1967). Elle requiert toutefois une extension de cette dernière à l'aspect dynamique comme aux aspects polysémiques et pansémiques (pouvoir évocateur d'une figure).

conceptuelle, car au delà de la forme du message, elle **trace et identifie les concepts** qui interviennent sur les objets présentés. Ces concepts deviennent alors accessibles et modifiables.

L'intérêt d'une ***visualisation conceptuelle*** est d'utiliser la cohérence entre des structures et bénéficier des hiérarchies de niveaux d'abstraction. La désignation d'un constituant d'une visualisation doit permettre d'accéder aux concepts ayant participé à la génération du constituant. Une formule par exemple, est appréhendée par le mathématicien comme un assemblage de concepts mathématiques. Ces concepts sont eux-même exprimés à l'aide de concepts de syntaxe abstraite relatifs aux notations, puis concrétisés par des symboles et composants lexicaux. Le tout est enfin adapté à une visualisation donnée, éventuellement avec des césures, des holophrastes ou des décorations de sélection.

Cette modélisation conceptuelle est essentielle pour l'assistance. Par exemple [Alem 88] propose un modèle biologique raffiné en un modèle mathématique qualitatif puis formel. En génération de scènes, [Djedi 89] part de phrases en Langage Naturel, les transforme en propriétés énoncées symboliquement, puis numériquement, pour aboutir à l'image via des valeurs numériques décrivant totalement la scène. Si la machine n'apporte au concepteur qu'un support au niveau de la représentation numérique - c'est le niveau actuel des modeleurs chargés de cette tâche - rien que le nombre de paramètres à introduire est suffisant pour rendre ce travail fastidieux. Une modélisation conceptuelle introduit des **critères de description pertinents**, et la productivité gagne un facteur d'échelle. Le Langage Naturel peut d'ailleurs, à lui seul, supporter plusieurs niveaux de description...

Lorsque le mathématicien choisit de visualiser une formule, il doit pouvoir agir directement sur les composants à tous ces niveaux. La visualisation conceptuelle bénéficie alors des répercussions que produisent un niveau supérieur, comme un changement de notation. Cette notion de manipulation directe étant liée à la visibilité, cela signifie que l'usager peut rendre visible les composants de ces niveaux d'abstraction supérieurs en agissant depuis la visualisation actuelle. L'**engagement direct** n'utilise pas exclusivement ce mode, puisqu'une requête adressée à l'Ami assistant peut effectuer l'action sur ce concept, en passant ou non par sa visualisation intermédiaire. Le mathématicien peut alors ici effectuer ce changement de notation en le demandant directement à l'assistant, ou en lui demandant les actions de changement de notation possibles.

Mais le mathématicien n'est pas obligé de visualiser une formule sous sa forme graphique habituelle. Il peut la vouloir par exemple dans un langage de commandes de saisie, dans un langage de programmation, ou décrite avec des mots et phrases du langage mathématique. De même, il peut n'en vouloir que certains aspects, comme visualiser les variables libres ou les symboles indicés. Un cas intéressant est celui de la visualisation par niveau conceptuel, comme celle du niveau d'assemblage des concepts mathématiques. Cet état brut peut être visualisé en tant qu'arbre ou par une description des concepts de l'assemblage. Dans les deux cas, il fait appel à des **schémas de présentation génériques** (arbre ou structure représentant un concept) dont les caractéristiques sont adaptables.

Cet exemple de visualisation d'une formule nous a permis de dégager deux types de points de vue sur une même scène. Le point de vue de **"modélisation conceptuelle"** choisit les concepts qui interviennent pour concrétiser l'objet mathématique abstrait considéré (ici : une formule). Ce point de vue relatif à l'expression gère alors les niveaux d'abstraction jusqu'à la forme concrète cible. Il est complété par un point de vue **d'observation**. Le point de vue d'observation permet d'agir sur les caractéristiques de ces niveaux afin d'adapter la présentation, pour mettre par exemple en valeur certaines propriétés de la formule.

Par analogie, une scène 3-D peut être modélisée par une expression langagière, ou une visualisation fil de fer, volumique... L'observation peut alors changer d'angle, déformer, observer le déroulement de la construction, choisir une coupe 2-D, voire rendre certains objets transparents ou colorés en fonction de leur température...

La **mise en œuvre** de la visualisation conceptuelle fait nécessairement appel à des techniques informatiques sophistiquées. L'enjeu consiste alors à savoir composer et présenter des informations dans un but relevant d'une tâche donnée. La connaissance des tâches élémentaires et des concepts mathématiques, additionnée de schémas de présentation spécialisés permet de l'envisager. Cette visualisation relève de lois disponibles a priori (issues de principes ergonomiques, d'usages informatiques et mathématiques consacrés ou ad hoc...). Elle est éventuellement complétée d'une action interactive de l'usager.

La visualisation conceptuelle cherche à favoriser les mécanismes de **"pensée visuelle"** et permettre l'association, la suggestion et l'évocation en stimulant l'usage de processus non verbaux.. Elle ne cherche donc pas systématiquement à transmettre une information non ambiguë ! Souvent lors d'un raisonnement

imaginatif, des raisonnements non verbaux sont à l'origine de la résolution du problème. Des notes et croquis très informels peuvent résumer et stimuler l'intuition. Plus tard, des informations incomplètes ou grossières sont utilisées (elles sont d'ailleurs ambiguës voire erronées à un niveau de détail trop fin). Elles peuvent être couplées à des concepts de portée générale. Finalement des informations précises mais de forme ambiguë sont visualisées, et l'ambiguïté peut être supprimée par changement du point de vue. Ces types de visualisation doivent disposer d'outils différents spécialisés dans leur traitement.

5.3. LA CONCEPTION D'UN SYSTEME D'ASSISTANCE

5.3.1. L'approche Intercom

Nous présentons l'approche Intercom permettant de **définir un modèle d'assistance dérivé de l'activité naturelle**. L'approche Intercom est venue en réaction à l'existence de systèmes déductifs puissants, mais finalement peu exploitables car leur principe de résolution de problème est différent et empêche le guidage humain.

Le postulat initial d'Intercom[3] est :

L'interaction est l'une des clés d'un travail efficace !

Pour cela, nous avons présenté la visualisation conceptuelle comme une réponse au besoin d'engagement direct de l'usager lors d'un processus de simulation-communication basé sur les connaissances et le langage de l'activité naturelle. Le problème de la conception d'un tel système d'assistance consiste alors à dériver les **spécifications du système** à partir du **savoir du domaine d'intérêt**, afin d'utiliser ultérieurement ce savoir à l'aide d'un système conçu pour faciliter l'exploitation de ce domaine. Ce problème est illustré figure 2.12 et développé pour la démarche adoptée dans notre cas.

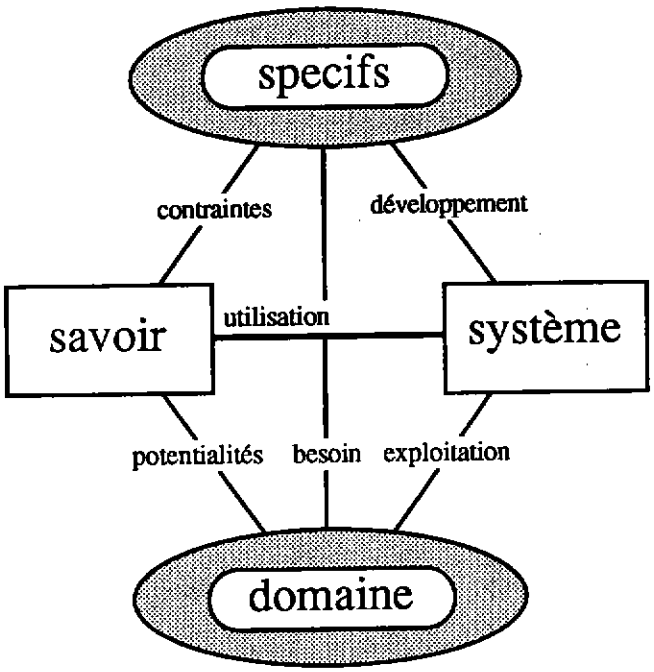


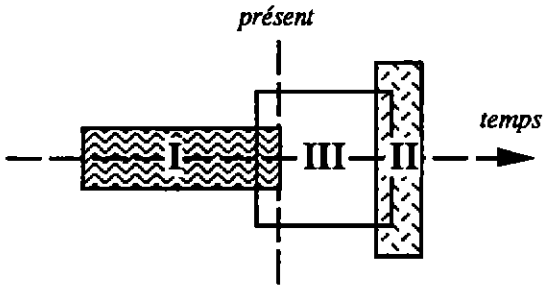
figure 2.12

La partie I de notre thèse a consisté en une étude détaillée des mathématiques, c'est-à-dire d'un domaine offrant de fortes potentialités d'assistance informatique. Cette étude se base principalement sur une analyse

[3] Ce postulat, et son appellation dérivée : Intercom, est issu d'une réflexion commune avec Laurent Chaudron avant d'être développé indépendamment. L'interaction doit être prise en charge dès la conception d'un système, et non prise en compte dans la phase finale où les choix précédents deviennent irrémédiables.

linguistique. Elle a permis d'évaluer la diversité du savoir existant, de cerner ses possibilités de modélisation, et d'apprécier les contraintes liées à son utilisation par le mathématicien.

A partir de l'étude du domaine, cette partie II a choisi un besoin considéré comme essentiel pour le développement des Mathématiques Assistées par Ordinateur. Les premières spécifications furent donc des souhaits réalistes d'assistance à la construction de preuves et démonstrations. Cela a entrainé une évaluation ergonomique des contraintes liées à l'utilisation du savoir du mathématicien, ainsi que la définition du savoir que le système-assistant doit partager.



Notre approche est ainsi prospective afin de fixer un objectif prometteur dans un contexte en évolution rapide. Avec ces deux parties, nous disposons des repères nécessaires à la définition du support informatique étudié partie III. Ces repères permettent aussi de renouveler l'intérêt des travaux concernant les mathématiques.

La partie III amorce donc le développement d'un système répondant aux besoins identifiés partie I et II. Ce développement commence par l'état des lieux et se poursuit par le raffinement des spécifications en tenant compte des techniques informatiques disponibles ou prochaines. Notre abord prospectif permet de faire apparaître les points critiques dans le développement d'outils mathématiques.

L'approche Intercom de conception d'un système d'assistance peut alors se résumer en trois points :

I) analyse d'une activité conceptuelle via le langage ;
II) transcription en une activité "naturelle" d'assistance par un système ;
III) développement expérimental ascendant de ce système.

5.3.2. Spécificités de l'approche Intercom

L'approche Intercom s'applique au développement d'un système "intelligent" pour les problèmes de type "assistance à la conception". L'approche Intercom propose un développement ascendant, en maîtrisant progressivement les tâches élémentaires (à partir d'un seuil de synergie minimal) selon le schéma figure 2.13.

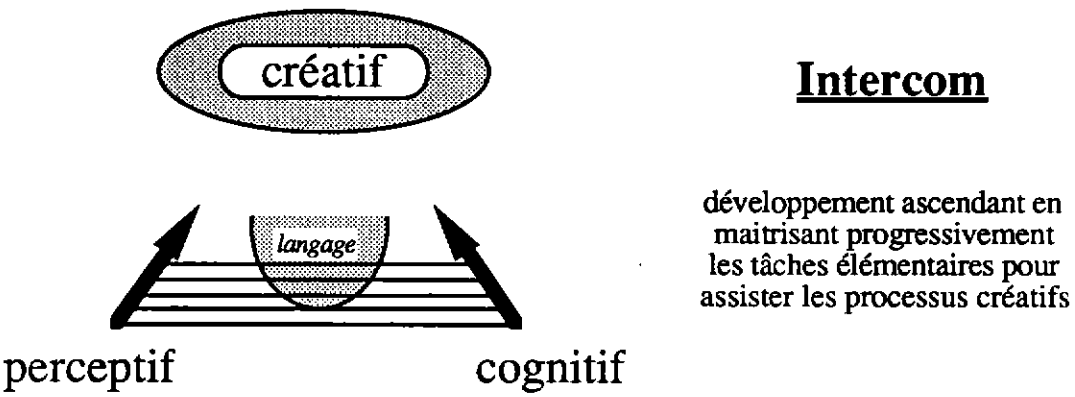


figure 2.13

Pour que le système soit viable, Intercom s'appuie sur la qualité de l'**interaction** et des **connaissances** disponibles. Il requiert notamment la maitrise technique et conceptuelle des langages graphiques et textuels utilisés dans l'activité de conception. La gestion des aspects graphiques est usuellement bien résolue, mais les aspects concernant la modélisation conceptuelle et l'expression textuelle restent insuffisants. La modélisation et la représentation des connaissances sont l'une des difficultés principales des domaines de conception. Les connaissances se distinguent par leur quantité, leur diversité, et la variété des points de vue possibles sur un "même" objet.

Utiliser un objet de mieux en mieux spécifié, adopter des points de vue différents au cours de son développement, faire des traitements couplés (calculs et visualisation graphique d'une formule par exemple) entre ces points de vue, requiert une **intégration** des connaissances autour des concepts communs. La diversité des connaissances est l'aspect essentiel nécessitant une **organisation conceptuelle** claire des objets, de leurs propriétés et de leur usage. Les connaissances sont alors si possible déclaratives : elles sont exprimées aisément dans un langage de description et le contrôle de leur utilisation est géré indépendamment. De plus, les connaissances susceptibles d'être modifiées sont explicitées. L'acquisition des connaissances est ainsi facilitée et leur quantité maîtrisable par la facilité d'extension du micro-domaine initial.

L'intégration du cycle de conception permet d'envisager le couplage et la réutilisation d'informations. Il devient dès lors envisageable d'utiliser comme **critère de décision** toute information relative aux tâches élémentaires. La modélisation conceptuelle permet alors à l'usager d'introduire les paramètres au niveau d'abstraction le plus général. Ce procédé évite le brouillage d'une interaction par un espace d'états trop souple (comme celui de toutes les constructions syntaxiques possibles d'une formule mathématique), et autorise un **guidage efficace** (sans nécessiter l'introduction d'un nombre important de paramètres).

Cette approche par interaction amène à réviser les fonctionnalités d'un système informatique. Lorsque l'espace de recherche est grand, une procédure incomplète est beaucoup plus efficace qu'une procédure de décision. Elle ne présente que des avantages dans une approche d'assistance par rapport à une approche "automatique". Il faut simplement adapter son taux d'insuccès de façon à ce que l'usager soit raisonnablement sollicité dans son interaction globale.

L'autre avantage majeur de l'approche Intercom par rapport à une approche automatique est d'être adaptée aux tâches des usagers, puisque cela a été le but de la conception initiale. Leur réalisation pratique ne demandent pas forcément des capacités informatiques plus complexes, mais surtout des capacités informatiques judicieusement exploitées.

Un dernier avantage est l'extension des possibilités. La diversité des fonctionnalités est utilisée de façon à fournir une gradation conceptuelle entre "le mou et le dur". Le dur représente ce que l'on sait faire, c.-à-d. les contraintes informatiques, le mou ce que l'on veut faire, c'est-à-dire les besoins de l'usager. Ces deux points de vue ne sont pas foncièrement incompatibles, à condition de savoir les rapprocher. Une solution consiste à concevoir des systèmes qui permettent l'intervention de l'usager pour les cas difficiles. Cette **répartition des tâches** permet d'exploiter potentiellement des algorithmes informatiques plus nombreux.

La différence est alors grande entre ce que l'on sait faire, et ce qu'il est possible de faire avec les mêmes techniques. Des algorithmes non-déterministes particulièrement intéressants peuvent alors être exploités à condition que le guidage de ce non-déterminisme puisse être pris en charge par l'usager pour rester dans des limites acceptables. La difficulté principale consiste alors à trouver des algorithmes qui s'insèrent naturellement dans les tâches et les raisonnement des usagers.

Si la conception effectue d'abord une recherche sur les besoins d'un domaine et les modalités de guidage souhaitées, les fonctionnalités désirées des systèmes deviennent différentes de celles que l'on sait réaliser a priori. Cette approche "molle" permet d'implémenter des fonctionnalités moins décidables mais plus puissantes ou novatrices, grâce au recours aux facultés de l'usager.

5.3.3. Vers un Atelier Mathématique Intégré

Un système mathématique développé selon cette approche Intercom fait alors appel à des fonctionnalités très diversifiées, regroupant notamment les fonctionnalités existantes. Nous parlerons alors d'un Atelier Mathématique Intégré (AMI). Notre objectif est bien entendu qu'un de ses modules puisse être qualifié d'Assistant mathématicien intelligent (Ami).

Cette notion d'atelier correspond à la tendance exprimée en Génie Logiciel ou en conception assistée par ordinateur, d'assister le cycle de développement d'un produit. Notre approche dans le cadre d'un AMI, met en avant les caractéristiques suivantes :

- Le rôle prépondérant du **guidage par le mathématicien** dans la résolution de problème (sauf dans des cas très précis bien délimités). Les sous-tâches sont à répartir entre l'homme et la machine, l'homme relevant de tâches plus "conceptuelles", et l'ordinateur plus "syntaxiques" ;
- L'assistance fournie par le système concerne principalement les **tâches élémentaires**.

- La simulation des phases de formulation et de présentation permet à un AMI de proposer au mathématicien un **environnement naturel**.
- Le besoin d'intégrer le **cycle de conception** : toute absence de fonctionnalité réduit l'homme à effectuer un travail important pour y pallier, s'il veut conserver un support informatique.

Dans notre approche Intercom, le développement se concentre d'abord sur la maitrise du langage, des connaissances et des tâches élémentaires. Il s'insère dans un processus permanent d'étude et d'application des techniques scientifiques à l'usage des mathématiques. Ce processus crée une synergie avec des recherches fondamentales en mathématiques, sciences cognitives et informatique. Nous distinguons trois phases de développement d'un AMI :

① avant-projet sommaire : phase préparatoire de conception informatique et de prototypage d'un AMI ;
② avant-projet détaillé : phase de mise en œuvre initiale appliquée aux manipulations d'un domaine mathématique réduit ;
③ expansion : phase de développement par enrichissement des fonctionnalités informatiques et des connaissances mathématiques.

Notre apport vers un AMI consiste à définir des spécifications pour l'avant-projet sommaire.

5.3.4. Discussion de l'approche Intercom

L'approche Intercom se définit par son but d'ergonomie naturelle et les principes de développement qu'elle se fixe pour y parvenir. Notre objectif est de fournir au mathématicien un système d'assistance à engagement direct. Il doit lui permettre en particulier d'exprimer les opérations de construction de preuves et de démonstrations d'une façon similaire à un compte-rendu oral. La réalisation de cet objectif nécessite notamment la mise en œuvre d'un grand nombre de connaissances mathématiques et paramathématiques.

L'approche Intercom est principalement limitée par l'**investissement initial important** et les imperfections de modélisation des connaissances. L'investissement initial est important car les techniques utilisées sont complexes, et le système doit gérer chaque catégorie de tâches élémentaires de l'activité humaine jusqu'à une "masse critique". Cette large palette de fonctionnalités indispensables est rendue délicate par le besoin de s'adapter à d'autres connaissances du domaine, et donc d'être indépendante d'un domaine de connaissances aussi étendu que possible. De plus cette contrainte de modélisation l'empêche de bénéficier directement de systèmes mathématiques existants, et ne permet de réutiliser principalement qu'un savoir externe à leur codage informatique. Enfin, dans un domaine aussi riche, il y a une différence notable entre ce que l'on cherche à dire et ce que l'on peut exprimer dans un langage exécutable.

Cet investissement initial dû à l'approche Intercom est rentable, car il permet aux utilisateurs :
- d'avoir envie de se servir du système ;
- d'introduire eux-mêmes de nouvelles connaissances.

Le premier point conditionne l'usage et la diffusion d'un système : nous avons proposé pour cela d'établir une interaction basée sur l'activité "naturelle". Le second point est déterminant, vu l'énorme quantité de connaissances requises pour faire des mathématiques. Ce quota ne le sera effectivement que si les usagers en introduisent une grande partie, pour l'adapter à leurs besoins. Globalement, les deux points sont conditionnés par la flexibilité du système, qui doit faciliter l'intégration de nouvelles fonctionnalités et l'amélioration des fonctionnalités existantes.

L'approche Intercom encourage le **développement expérimental** de fonctionnalités "intelligentes". L'usager peut étendre les connaissances, composer des tâches élémentaires existantes, ou exprimer des besoins en fonction de son expérience. L'activité de l'usager peut être analysée[4] (automatiquement ou non), des expérimentations effectuées et des outils interdisciplinaires testés et adaptés. L'informaticien dispose enfin d'informations pertinentes pour le développement de logiciels de dialogue, d'explication et de raisonnement, ainsi que leur mise en valeur dans un environnement déjà constitué. La maitrise progressive de tâches élémentaires permet d'enrichir le niveau des tâches assistées ou réalisées par le système, au profit de l'exploitation des processus créatifs humains.

---

[4] par définition d'une activité de conception, il faut analyser le "faire faire" à partir du "faire", car on ne sait pas le dériver à partir de l'énoncé du problème (c'est le cas pour nombre de preuves mathématiques, sans parler de leur réutilisation ou de la production de démonstration textuelles).

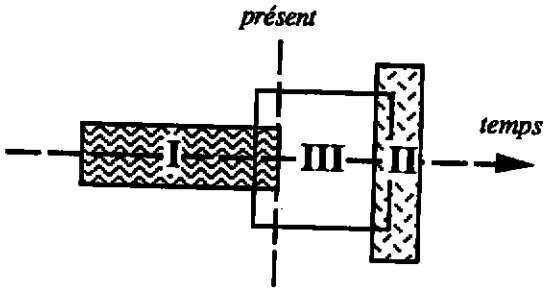
Bien que les tâches prises en compte soient élémentaires et bien définies pour le mathématicien, leur adaptation pour un AMI requiert l'introduction de techniques informatiques sophistiquées mais disponibles. Les fonctionnalités du système d'assistance dans l'étape d'utilisation sont alors aussi un éventail des besoins. Nous avons illustré ici comment l'assistance aux tâches de **manipulations élémentaires** peut être dissociée des tâches de niveau d'abstraction supérieur. Cela permet de fournir au mathématicien un jeu d'outils et de paramètres pour le libérer du fastidieux et de l'ennnuyeux (en gérant élision, omission, notations, réutilisation...). La **flexibilité** d'utilisation ou d'extension du système est pour cela déterminante.

Pour arriver à un engagement direct, nous avons choisi la modélisation de l'activité mathématique naturelle en mettant l'accent sur la **compréhension du langage** et sur l'**intégration du cyle de conception**. Cela permet de rentabiliser l'impact d'une visualisation sophistiquée, moins importante voire impraticable à des fins calculatoires.

Cette intégration offre de plus la perspective d'une compréhension profonde d'une démonstration, avec toutes les facilités d'assistance que cela autorise. La production d'une démonstration dérivée à partir de l'énoncé d'un problème est alors envisageable. L'intervention de l'usager mathématicien est d'autant plus modérée que les connaissances et les capacités de déduction disponibles permettent de n'indiquer que les grandes lignes de cette démonstration.

PARTIE III

Les Techniques Informatiques



1. ÉTAT DES LIEUX DES SYSTEMES INFORMATIQUES

1.1. INTRODUCTION AUX MATHEMATIQUES ASSISTEES PAR ORDINATEUR

1.1.1. Les objectifs du panorama des systèmes existants

Le vocable de Mathématiques Assistées par Ordinateur (MAO) n'est guère répandu car il n'existe actuellement qu'une panoplie d'utilitaires hétéroclites. Des travaux théoriques en mathématiques ont permis de disposer d'algorithmes adaptés à une vaste classe de problèmes à résoudre. Leur implantation permet à des systèmes informatiques de se substituer à la phase de recherche de la résolution de problème. Leur usage est donc principalement de fournir des résultats au bénéfice des autres activités scientifiques.

Dans une optique tout à fait différente, une autre catégorie de systèmes s'intéressent aux phases de formulation et de présentation, sans chercher à résoudre les problèmes mathématiques. Ainsi, les systèmes de traitement de texte mathématique permettent de rédiger une démonstration sans pouvoir faire le lien avec la phase de recherche. Lorsqu'une passerelle est malgré tout établie, les systèmes ne savent pas ce qu'est une preuve et ne produisent pas de démonstration utilisable pour tester sa validité mathématique.

Pour caricaturer la situation présente, le mathématicien dispose d'un "calculateur" et d'une imprimante pour l'aider dans son activité. Ce blocage serait définitif si d'autres catégories de systèmes ne permettaient d'envisager un renouveau prochain. Ce domaine des MAO reste toutefois prospectif, puisqu'il n'existe **pas de recueil des systèmes existants** (à notre connaissance). Plus grave encore, la plupart des outils existants dans ce domaine ne sont pas destinés aux mathématiques[1] !

Ce panorama a donc pour premier objectif de **faire le point** sur la situation existante. Il est classificatoire plutôt qu'exhaustif, et est indépendant d'un domaine mathématique particulier. Il est subjectif car il accorde plus d'importance aux systèmes plus inusuels.

Souvent, les potentialités informatiques sont mal perçues. Notre second objectif est de préciser **l'apport éventuel de l'informatique aux mathématiques.** Cet apport doit souvent être cherché parmi les techniques informatiques, les mathématiques étant rarement l'objet effectif de leurs préoccupations.

Cet apport dépendra principalement des connaissances mathématiques mises en œuvre. Notre troisième objectif est alors de déterminer la **nature des connaissances disponibles.**

Enfin, il n'existe pas de vue globale du domaine des MAO. Notre objectif final est ainsi d'identifier les **efforts de recherche partageables** et de susciter de nouvelles propositions pour les MAO. Ces objectifs seront finalisés pour la conception d'un Atelier Mathématique Intégré (AMI).

Cette étude correspond à un **besoin réel.** Notre but n'est pas d'ajouter un nouvel "XAO", mais d'étudier la spécificité de l'assistance au mathématicien. Nous cherchons à évaluer l'informatisation de l'activité mathématique, sachant que l'essentiel de cette activité est actuellement exercée en dehors de systèmes informatiques. Par ailleurs, il n'existe pas à notre connaissance d'ouvrages ou projets s'intéressant aux usages langagiers, manipulations de formules, et organisation des connaissances du mathématicien, si ce n'est dans le domaine de la géométrie élémentaire.

Nous avons scindé notre panorama en fonction de la nature des tâches principales (figure 3.1).

---

[1] En effet, les recherches portent beaucoup sur les techniques informatiques de déduction, de guidage, d'apprentissage symbolique, d'hypertextes... Les résultats publiés se retrouvent alors dispersés dans la rubrique informatique correspondante. Les mathématiques sont en fin de compte choisies de façon annexe, comme domaine d'illustration valorisant, mais aussi pour la qualité intrinsèque de leur modélisation. Les tuteurs intelligents constituent paradoxalement le pôle d'intérêt le plus impliqué pour des recherches concernant les mathématiques. Les MAO nécessitent pourtant des recherches spécifiques...

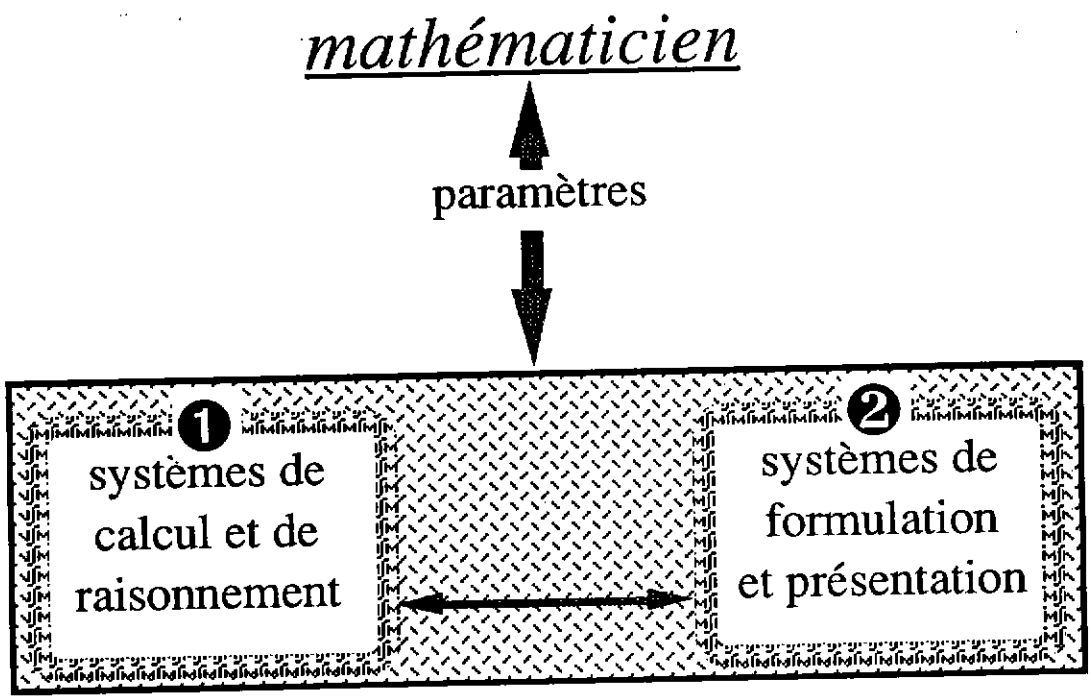


figure 3.1

Cette partition correspond aux finalités principales des systèmes existants. Les systèmes de calcul et de raisonnement mettent l'accent sur les tâches mathématiques, tandis que les systèmes de formulation et présentation ont une vocation principalement paramathématique. Nous proposons une analyse des systèmes existants selon quatre critères :

① analyse conceptuelle des tâches du mathématicien ;
② modélisation des processus mis en œuvre pour les tâches ;
③ faire des mathématiques "à la façon" du mathématicien ;
❹ produire un résultat sans chercher de similarité avec le mathématicien.

Les critères ① et ② relèvent d'une réflexion sur l'activité mathématique. Ce panorama présente brièvement des outils correspondants aux critères ③ et ❹. Nous positionnons ces outils relativement aux critères ① et ②, tout en précisant leur limites. Nous commentons alors leur conception ainsi que les techniques utilisées. Lorsque les outils n'existent pas encore, l'application éventuelle des techniques informatiques aux mathématiques est discutée. Nous évaluons enfin, selon les priorités de développement, l'intégration de l'outil ou de ses fonctionnalités dans un AMI.

1.1.2. Topologie des MAO

Ce panorama étant défini à partir d'outils et de techniques informatiques, il est nécessaire de le définir aussi à partir des composantes de l'activité mathématique. En adoptant une variante informatique du trièdre initial : mathématicien, langage, données (figure 1.1), nous obtenons trois pôles :

- raisonnement ;
- langage ;
- connaissances.

Des systèmes informatiques existent pour reproduire le raisonnement, permettre l'exploitation du langage, ou faciliter l'utilisation des connaissances. Toutefois, ces systèmes se limitent souvent à privilégier exclusivement l'un de ces trois pôles, et à s'intéresser à un aspect trop restrictif de l'activité mathématique. Le travail du mathématicien ne s'en trouve pas mieux assisté pour autant !

Nous avons vu - partie II - quel type d'assistance le mathématicien peut souhaiter dans le cadre de son activité de construction de preuves et de démonstration. C'est ce type d'activité mathématique qui nous intéresse pour envisager des Mathématiques véritablement Assistées par Ordinateur : elle nécessite une synthèse des trois pôles : raisonnement, langage et connaissances.

Les champs d'action des MAO

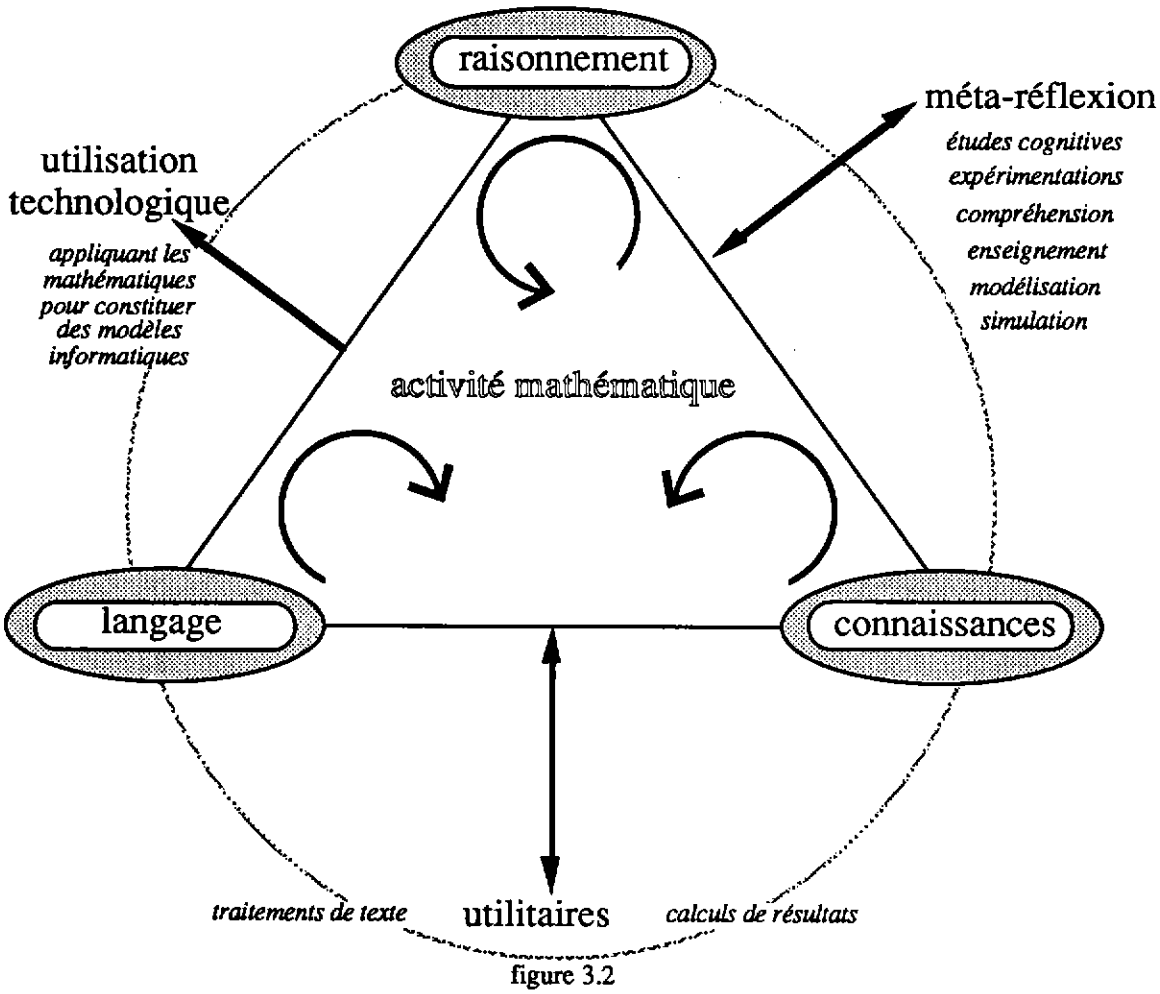


figure 3.2

Nous pouvons alors définir les champs d'action des MAO en fonction du point de vue adopté par l'intervenant principal (figure 3.2) :

- selon un point de vue interne, le mathématicien fait des mathématiques. En insérant ce travail dans son contexte, le mathématicien déploie une **activité mathématique**. Il peut alors bénéficier de systèmes prévus pour participer à cette activité et lui fournir une assistance relative aux trois pôles du trièdre.
- par ailleurs, le mathématicien peut exploiter des **utilitaires** technologiques, chargés de lui apporter une aide ponctuelle lors du déroulement de son activité. Cette panoplie d'outils offre des services limités : lorsqu'ils permettent de développer une activité centrée sur leurs services, elle est suffisamment spécifique pour être éloignée de l'activité mathématique "naturelle". La génération des grands nombres premiers est un exemple typique. La "démonstration" du théorème des quatre couleurs montre néanmoins l'intérêt épistémologique de ces utilitaires.
- d'autre part, les mathématiques sont souvent utilisées comme modèle (par ex. d'une réalité physique) et destinées à être transcrites en un modèle calculable. Cette **utilisation technologique** fait l'objet de recherches théoriques développées au sein de l'activité mathématique pour répondre à leurs besoins algorithmiques spécifiques.
- enfin, le travail effectivement réalisé au sein de l'activité mathématique est analysé à des fins externes dans une **méta-réflexion**. Cette réflexion sur l'activité mathématique cherche à l'analyser et à la modéliser à des fins scientifiques (produire des systèmes d'assistance, améliorer l'enseignement, établir des modèles comportementaux, etc.)

Nous avons centré notre approche des MAO sur l'activité mathématique telle qu'elle est pratiquée dans la "couche mathématique générale". Nous sommes donc spécifiquement concernés par la réalisation d'une

activité mathématique assistée par un module de type Ami, et par les réflexions nécessaires à la modélisation de l'activité. Les MAO suscitent par ce biais un thème de (méta-)réflexion supplémentaire :

l'interaction du mathématicien et d'un système informatique,

pour faciliter l'activité menée par le mathématicien.

1.1.3. Présentation de l'évolution des systèmes informatiques

La métaphore des systèmes informatiques d'antan, avec un interpréteur FORTRAN II, un buffer d'édition et une imprimante constitue une caricature représentative du panorama des systèmes commercialisés. Elle serait toutefois injuste vu le nombre de systèmes novateurs mais indépendants qui pointent pour balayer ces temps révolus.

Comme les environnements de programmation, les systèmes mathématiques s'enrichissent et multiplient leurs fonctionnalités. L'intégration des capacités de calcul et de traitement de texte est significative de cette évolution. La complexité de la gestion de l'interaction entraîne une dissociation de l'interface avec les capacités de traitement. Le besoin de manipuler plusieurs types de données, toutes structurées, permet de concevoir des **outils génériques de manipulation des structures**. Cette évolution est schématisée figure 3.3 ci-dessous.

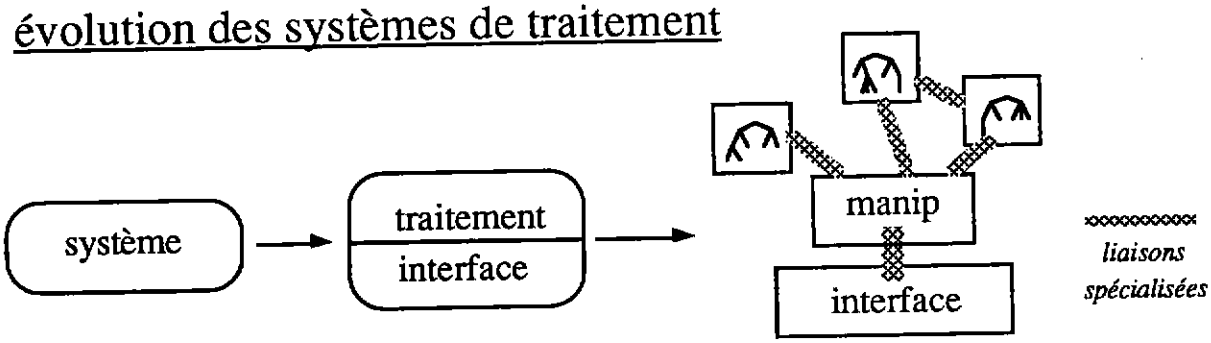


figure 3.3

L'interface et la gestion de l'interaction deviennent ainsi indépendantes des outils de manipulation de données, et cette abstraction s'accompagne conjointement d'un accroissement de leur fonctionnalités. Mais cette réorganisation ne peut s'accomplir elle aussi qu'en reconsidérant les procédés de traitement de chaque module. Le traitement s'effectue lors des liaisons spécialisées représentées en grisé : il consiste là aussi à intervenir sur les structures via leur manipulateur, afin d'obtenir le résultat désiré. Par exemple, la structure "1+(0+3)" est réécrite en "1+3" puis en "4" par un manipulateur qui exécute les ordres de traitement.

Cette évolution des procédés de traitement est schématisée figure 3.4 pour aboutir à un **modèle de Système-Expert (S.E.)**.

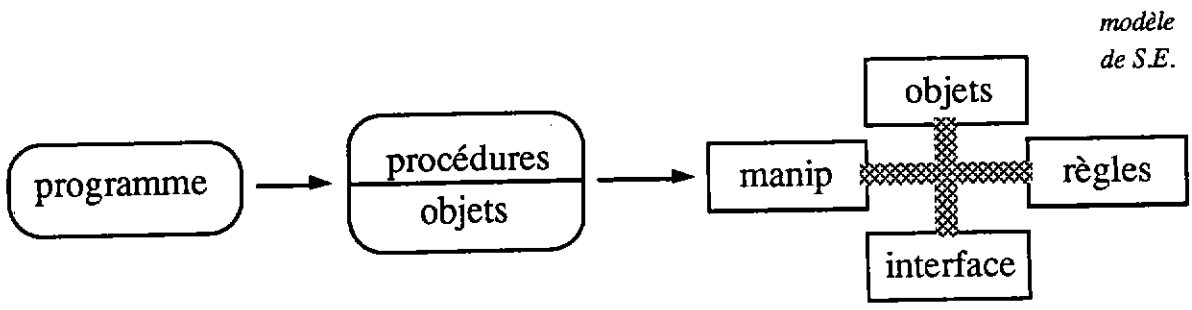


figure 3.4

D'une entité complexe baptisée programme, les informaticiens ont cherché à dissocier les objets (c.-à-d. structures de données) de leurs procédures de traitement (pour les encapsuler par exemple, dans le cas des langages à objets). Puis les procédures ont été elles-même dissociées en « logique + contrôle », c'est-à-dire dissociées en règles de traitement et en exploitation de ces règles.

Par exemple, les inférences d'un S.E. sont produites à l'aide de règles exploitées par un moteur d'inférences (manip). L'interface sert alors à agir sur l'un de ces composantes du traitement. Pour l'usager, l'interface sert à introduire les données du problème ; pour l'expert, il sert à modifier les règles de résolution. Cette évolution a permis de disposer d'objets et de règles, c'est-à-dire d'éléments modulaires de petite taille qu'un procédé de manipulation général va exploiter pour constituer l'équivalent d'un programme.

Ces abstractions successives des systèmes ou des procédés de traitement aboutissent à des **outils génériques de "manipulation"**, selon des mécanismes similaires aux procédés de guidage de raisonnement. Leur intérêt est que ces outils de manipulation deviennent indépendants et du domaine de travail, et d'un traitement particulier (et donc indépendant d'un domaine spécifique).

En contrepartie de leur généralité, ils exigent des **"connaissances"** (objets et règles) afin de décrire le domaine, le problème considéré et son type de traitement. Il y a ainsi notamment autant de paquets de connaissances qu'il y a de langages traités par un système. Néanmoins, les connaissances dédiées à un langage peuvent être utilisées par plusieurs outils comme l'analyseur, l'afficheur ou le démonstrateur. Réciproquement, un outil peut être utilisé pour plusieurs langages.

L'introduction de connaissances n'est pourtant pas le gage d'un système adaptable. Les connaissances qualifiées "**de surface**" tendent à simuler directement le raisonnement d'un "expert" dans une tâche précise. La spécificité de ces connaissances les rend mal adaptées à la réalisation d'une tâche voisine, ou pire encore, à la réalisation d'une même tâche dans une situation légèrement différente. Ces connaissances sont donc compilées pour une tâche, y compris souvent pour une façon de la résoudre. Elles ne permettent pas de déterminer ce qui est pertinent pour une variation donnée. Elles n'ont donc qu'un **domaine de validité très restreint**, quand bien même il est explicite !

Les connaissances dites **"profondes"** - et les mécanismes de traitement qui peuvent les exploiter - tendent à pallier ces inconvénients. Ils nécessitent des systèmes plus sophistiqués, qualifiés de S.E. de seconde génération [Steels 85]. Le qualificatif "profond" provient du fait que le système dispose d'un modèle du domaine d'application, et que ce modèle est utilisé pour la réalisation des tâches. Un des aspects d'un modèle profond est qu'il exploite les causalités entre objets du modèle pour la réalisation des tâches. Les tâches réalisables sont alors plus variées, car elles peuvent exploiter ce modèle pour s'adapter aux variations de situation.

Outre une meilleure variabilité des tâches, les avantages ainsi dégagés sont nombreux. Ils se traduisent par l'amélioration de la qualité des thèmes qui font l'intérêt des Systèmes Experts : modélisation des connaissances, acquisition des connaissances, explications, apprentissage symbolique, etc. Les extensions intéressantes comme l'enseignement ou le traitement du langage sont aussi facilitées. Un éventail assez large de problèmes est abordé dans le numéro spécial Explication [RIA 90], dont trois des neufs articles sont consacrés aux mathématiques (tuteurs).

La nature des connaissances et l'adaptativité nécessaire en mathématiques rend indispensable l'utilisation d'un modèle profond. C'est alors la **modélisation des connaissances** et sa mise en œuvre qui pose problème. Il faut en effet déterminer quels modèles sont disponibles et pourront être réalisés. C'est par exemple ce que nous avons fait pour les preuves et démonstrations, dans le cadre des manipulations de formules.

Un modèle n'est utile que lorsqu'il sert de plateforme commune à plusieurs utilisations. Ainsi, un modèle dédié au raisonnement ne prend souvent pas en compte les pôles "langage" et "connaissances", et ne peut être qualifié de "profond" que relativement à des utilisations très proches de l'explication du raisonnement suivi. Un modèle profond prend notamment en compte les objets de base du domaine (concepts, formules, théorèmes, preuves, etc.) pour y associer des problèmes et un savoir-faire d'utilisation.

Par contre, la "profondeur" du modèle choisi a une influence déterminante sur la faisabilité des raisonnements. Pour l'exploiter, il faut en effet y rajouter les connaissances permettant d'obtenir un raisonnement équivalent à un raisonnement "de surface". Or ce dernier résulte en partie d'une expertise pour laquelle aucun modèle satisfaisant n'est disponible.

Cette **incomplétude intrinsèque** vient limiter les avantages des modèles profonds. Elle est contournée par un éventail de modèles destinés à écarter les zones informes. Néanmoins, le guidage et l'enchaînement de l'ensemble est souvent problématique : l'usager a alors comme recours principal de lancer une tâche, attendre son verdict, et lancer une tâche plus simple en cas d'échec. Il est alors utile, pour être moins sommaire, de disposer de raisonnements qui approximent les raisonnements utilisés dans le domaine.

Le **paradigme d'assistance à l'activité naturelle** que nous avons proposé est prometteur car il permet à l'usager de pallier les incomplétudes des modèles. Il permet le recours à des modèles plus profonds car il n'exige pas une quasi-automatisation. De plus, nous avons pu délimiter, avec les tâches élémentaires, une zone où les causalités sont suffisamment fortes pour que les informations supplémentaires à introduire pour le guidage soient limitées.

1.1.4. Typologie des techniques de conception informatique

Afin de caractériser les techniques utilisées pour concevoir les différents modules de traitement de l'information, nous proposons d'utiliser la typologie suivante :

- **conception a priori**, dont l'archétype est un programme classique.
  - Le concepteur pense et écrit tout le module : la généralité du module provient uniquement de la diversité des données traitées.
  - Les effets du traitement sont déterminés par le concepteur.
  - Des moyens d'action sur le traitement sont souvent fournis grâce à des paramètres d'utilisation.
  - Les extensions de traitement possibles sont facilitées par la modularité du langage de programmation.
  - L'intégration d'autres fonctionnalités s'effectue uniquement par passage de données.
- **conception déclarative**, dont l'archétype est un Système-Expert.
  - Le concepteur bâtit le module en séparant le "quoi" du "comment" dans le traitement[2]. Le "quoi" fait appel à des connaissances de base, dont l'organisation peut éventuellement être complexe. Le "comment" décrit leur utilisation selon une conception a priori ou selon une conception déclarative à partir de règles de déduction.
  - Les effets du traitement sont déterminés par les conditions initiales et les lois d'évolution dynamique du traitement. Ils se distinguent déjà des prévisions du concepteur par leur complexité calculatoire.
  - Les moyens d'action sur le traitement sont incrémentaux : l'usager peut ajouter ou modifier les connaissances ou les paramètres de contrôle. De plus, il peut radicalement changer de domaine, ou de mécanisme de contrôle.
  - Les extensions de traitement possibles dépendent alors principalement de la qualité de représentation des connaissances.
  - L'intégration d'autres fonctionnalités permet le partage de connaissances communes et l'utilisation éventuelle d'un noyau de traitement commun. Le module initial s'est ainsi subdivisé en systèmes.
- **conception multi-ordinale** (à plusieurs niveaux), dont l'archétype est un système d'explications.
  - Le concepteur bâtit aussi par exemple un module chargé d'examiner et de raisonner sur le module effectuant le traitement de base. Le raisonnement prend alors sa véritable dimension multi-ordinale.
  - Les effets observés sont plus complexes puisqu'ils faut intégrer plusieurs niveaux à l'œuvre.
  - Les moyens d'action permettent d'agir plus finement sur les divers traitements en choisissant le niveau approprié.
  - Les extensions de traitement possibles sont facilitées par la possibilité d'adjonction de niveaux supplémentaires lorsqu'une connaissance fait intervenir plusieurs représentations.
  - L'intégration d'autres fonctionnalités est facilitée par la factorisation de modules de traitement partageant des ressources communes (l'interface, les procédés d'explication, un modèle "profond", la description générique des langages, etc.).
- **conception introspective**, dont l'archétype est un système qui s'améliore par apprentissage.

[2] Selon la célèbre équation de Kowalski : algorithme = logique + contrôle.

  - Une fois que plusieurs niveaux de traitement sont introduits, certains peuvent être spécialisés dans l'examen des paramètres de traitement du système. Cette description réflexive d'une partie de lui-même[3] le rend capable de s'auto-analyser et de modifier ses constituants.
  - Les effets observés sont encore plus complexes puisqu'ils faut intégrer la dynamique d'évolution du système en plus de la dynamique de traitement. Par exemple un système peut-être conçu par un procédé de bootstrap à partir d'un noyau de lui-même.
  - Les moyens d'action et l'expressivité sont accrus.
  - Les extensions de traitement possibles ne sont plus toujours facilement maîtrisables.
  - L'intégration de fonctionnalités telles l'apprentissage ont des répercutions importantes sur de nombreux modules du système.
- **conception dynamique**, dont l'archétype est un système multi-agents[4].
  - Le système n'a plus alors une organisation figée en modules, mais se définit au contraire par un équilibre dynamique entre agents qui communiquent[5].
  - Le concepteur ne peut plus envisager la complexité globale du système autrement qu'en la pensant en terme de modules élémentaires et de systèmes de communication.
  - Les moyens d'action sont encore plus éloignés du résultat final et l'étude de tels systèmes constitue une branche ardue des sciences.
  - Les extensions de traitement possibles sont assez erratiques en fonction des modifications des paramètres initiaux.
  - Cette conception entièrement différente a recours au passage de données ou au partage de processus pour intégrer d'autres fonctionnalités qui lui sont complémentaires.

Si cette gradation des conceptions fait appel à des mécanismes de plus en plus sophistiqués, les traitements des systèmes résultants ne sont pas meilleurs pour autant. **La complexité des techniques de conception va plutôt de pair avec la finesse des modélisations conceptuelles**. Une conception dynamique, introspective ou multi-ordinale est utilisée pour ses niveaux d'analyse conceptuelle et de modélisation des processus. Une conception introspective, multi-ordinale ou déclarative est mieux adaptée à des systèmes reproduisant la démarche du mathématicien, tandis qu'une conception multi-ordinale, déclarative, quoique plus fréquemment "a priori", est utilisée pour concevoir des systèmes à usage mathématique. Même si cela serait utile pour la perception ou pour certains processus de raisonnement, aucune conception dynamique ne sera abordée ici.

1.2. LES OUTILS POUR RAISONNER ET CALCULER

1.2.1. Les systèmes de calcul

1.2.1.1. Les catégories de systèmes disponibles

Le traitement informatique de données numériques a conduit a développer un **calcul scientifique** autour de langages informatiques spécialisés (FORTRAN, APL, etc.). Cela a entrainé le développement de techniques numériques de résolution de problème, qui forme l'utilisation technologique principale des mathématiques. Cette branche florissante de l'informatique constitue un marché indépendant pour lequel de nombreux outils d'aide sont disponibles et où des bibliothèques de logiciels scientifiques sont constituées.

Cette utilisation technologique permet subsidairement d'offrir au mathématicien des utilitaires facilitant ses recherches expérimentales (visualisation de courbes approximées, etc.). Elle ne présente pas d'intérêt direct pour son activité mathématique, mais permet de développer de nouvelles formes d'activité.

Le traitement informatique de données symboliques a permis de développer des systèmes d'intérêt plus immédiat pour le mathématicien. Il a fourni les moyens algorithmiques de développer la **Démonstration Automatique de Théorèmes** (DAT). Il a permis l'émergence de la programmation heuristique et le développement mathématique d'algorithmes de calcul symbolique (ou formel). Les

---

[3] La réflexivité est, en effet, la propriété d'un système ou d'un agent de représenter une partie dynamique de son état.

[4] Nous n'aborderons pas le problème de la conception ultime, dont l'archétype est un système "réel" : seuls certains mécanismes sont connus, en particulier les principes de fonctionnement, le fonctionnement détaillé et le bootstrap initial sont entachés d'incertitude.

[5] Jacques Ferber parle d'eco-système composé d'entités, avec un comportement autoépiédique qui recherche l'équilibre [Ferber 89].

systèmes de calcul formel sont immédiatement évoqués maintenant dès que l'on discute de systèmes pour les mathématiciens.

Les systèmes de calcul symbolique sont issus des travaux précurseurs de James R. Slagle sur l'intégrateur symbolique SAINT (Symbolic Automatic INTegrator [Slagle 63]). Il fut suivi du programme SIN de Moses [Moses 71] qui évolua en MACSYMA. SAINT utilisait des techniques heuristiques pour trouver des transformations intéressantes permettant de résoudre le problème "à la façon" du mathématicien. La reconnaissance de formes symboliques constituait, aux dires de l'auteur, une partie très importante du programme.

Le développement des travaux théoriques en algèbre formelle ont permis ultérieurement de trouver l'algorithme de Risch [Risch 69] permettant de résoudre une vaste classe de problèmes d'intégration, par des procédés tout à fait différents rendant l'explication impossible. Le résultat peut néanmoins être prouvé a posteriori par dérivation. Une telle démarche a un intérêt pratique évident, et a été étendue à d'autres types de manipulations symboliques (par ex. la procédure de Wu en géométrie [Chou 88]).

Le regroupement des techniques algorithmiques et heuristiques pour des manipulations symboliques variées (principalement algébriques) a permis de proposer des systèmes à usage généraliste ou spécialisé. Une présentation des systèmes et de leur évolution est par exemple disponible dans [Davenport 86, Vivet 84]. Nous nous intéressons plus particulièrement ici à leurs développements récents.

L'évolution la plus caractéristique n'est pas l'extension de leurs domaines d'application mathématiques, mais plutôt l'amélioration de leur utilisabilité pratique (pour réaliser plus facilement, des tâches plus diversifiées). Nous présentons brièvement ici le dernier né des systèmes à ambition générale : Mathematica™.

1.2.1.2. Présentation et limites de Mathematica

Mathematica™ est présenté par son père conceptuel, Stephen Wolfram, dans l'ouvrage [Wolfram 88] qui sert aussi de manuel de référence. Il vise le créneau scientifique et technologique concernant l'application de méthodes quantitatives. Il fournit à la fois un outil et un langage de programmation spécialisés pour les mathématiques.

Mathematica effectue des calculs numériques et symboliques, ainsi que des affichages de graphiques. Il peut être utilisé comme un résolveur de problème tout comme un outil de création de documents intégrant au texte mathématique des capacités de calcul. De plus, Mathematica est conçu dans un paradigme d'environnement "boite à outil" : il fait partie de UNIX™ et communique avec FORTRAN, C, TEX™ et POSTSCRIPT™.

Mathematica est implémenté à l'aide d'un langage à objets compilé en C. Son principe de fonctionnement est celui de règles de réécritures. Muni d'un pattern-matching efficace, Mathematica permet l'expression de règles déclaratives très "propres". Toute expression est réécrite autant que possible pour parvenir au résultat. La connaissance des objets mathématiques se résume principalement à des règles de réécriture les concernant. L'ambition de Mathematica est de **se substituer aux langages de programmation** pour le traitement d'expressions mathématiques.

Les catégories d'objets recensés par Mathematica sont significatives de cette ambition. Par nombre décroissant on recense :

- les objets mathématiques, c'est à dire les principales fonctions trouvées dans les recueils de tables de valeurs ;
- les "divers", c.-à-d. les objets liés au "système mathematica" ;
- les commandes qui permettent de guider et d'effectuer l'activité mathématique ;
- les opérations d'entrée-sortie ;
- les fonctions et options graphiques ;
- les opérations de manipulation de la structure des objets (modifications de listes...) ;
- les instructions de programmation ;
- les prédicats (opérations booléennes).

La principale critique que nous formulerons à l'encontre de Mathematica est sa volonté de se situer en tant que langage de programmation mathématique, et de ce fait négliger dans un premier temps l'interaction avec l'utilisateur. Ceci est d'autant plus visible que le mode de manipulation des formules est primaire. De plus, son usage dénote une ***polarisation vers un résultat, et NON en faveur de la démarche.***

Un système comme Mathematica introduit nécessairement un biais par rapport à l'usage qu'en aurait un mathématicien qui utilise un "scribe assistant Ami" pour l'aider dans ses tâches simples. Le but du système est d'agir sur la formule présente en vue d'obtenir un résultat. Le type d'action cherché est spécifié par la commande de l'utilisateur. Le mathématicien doit chercher la (séquence de) commande lui permettant d'exprimer son but immédiat ou de parvenir à ses fins.

Comme tout système de calcul formel actuel, Mathematica s'avère souvent déficient dans ses explications ou dans son guidage. Des tentatives comme [Purtilo 89] cherchent à organiser ce guidage et à exprimer des plans. Les problèmes restent néanmoins nombreux, ne serait-ce que parce qu'un guidage très fin est nécessaire. Nous avons regroupés en Annexes quatre façons différentes d'intégrer cos(x), aboutissant chacunes à des résultats différents. Le mathématicien qui connaît l'importance de la forme obtenue, les difficultés de définition d'une "forme exacte", l'importance des "bons" regroupements et le rôle de la détection de similarités entre expressions, aimerait souvent pouvoir guider les manipulations pour contrôler l'obtention du résultat.

Le système Mathematica est intrinsèquement conçu pour mettre en œuvre des méthodes permettant de parvenir à un résultat. Il est fait pour résoudre, plutôt que pour aider à résoudre. Hors de la classe de problèmes qu'il peut résoudre directement, cela constitue un handicap.

1.2.1.3. Évolution des systèmes de calcul formel

Par ailleurs, les recherches sur les procédés de calcul formel restent très actives. Le système SCRATCHPAD [Jenks 84], développé depuis plus de 10 ans au centre de recherche T. J. Watson d'IBM, est particulièrement intéressant pour son organisation utilisant des Types Abstraits Algébriques et permettant la définition de fonctions polymorphes [Sutor 87].

Rejoignant cette approche d'amélioration des structures de typage, l'équipe SYSIPHE de l'INRIA Sophia Antipolis développe depuis 1988 un système plus ouvert que SCRATCHPAD. Leurs efforts portent notamment sur une conception moderne du logiciel et de l'interface, l'intégration rapide de nouveaux algorithmes, et l'utilisation de techniques de réécriture.

L'état actuel de Mathematica ([Wolfram 88]) permet d'envisager des extensions possibles ou des versions ultérieures. Ces remarques s'appliquent aux autres systèmes de calcul formel, et sont d'ailleurs générales car elles prennent en compte des besoins effectifs :

- **aspects spécialisés à des domaines** scientifiques et techniques spécifiques, comme par exemple la cosmologie, la gestion ou la combinatoire des langages réguliers.

- **interface de manipulation moins sommaire**, intégrant une manipulation graphique et directe des formules. Le chapitre 2 qui va suivre illustre par exemple nos souhaits pour SOFTMATH™.[6]

- **usage généralisé du langage et des notations** mathématiques usuelles (celles de Mathematica sont néanmoins très bien pensées, pour un langage de programmation). Sinon, pourquoi ne pas garder aussi la forme "BesselJ[0, 10.5]" au lieu de "$J_0(10.5)$" dans les commentaires [op. cit. page 3]. Pourquoi enfin systématiquement réordonner sous une forme normale moins lisible ou présenter implicitement la factorisation de 18 sous la forme : "Out[11]= {{2, 1}, {3, 2}}" [op. cit.].

- **production de l'argumentation** correspondant à une session. Malheureusement, cela ne présente guère d'intérêt dans le cas présent car si le système sait parfois quelle règle de réécriture il emploie, les méthodes de résolution du système ne sont pas explicites. De plus, l'argumentation est un concept étranger au système.

- **"ouverture" des méthodes utilisées** (contrôle, extensions, explications...). Cela fait écho au besoin précédent. Il est de plus intéressant d'avoir un méta-système de calcul formel. Le méta-niveau

[6] Mathematica, UNIX, TEX, POSTSCRIPT et SOFTMATH sont respectivement des marques déposées de Wolfram Research Inc., AT&T, AMS, Adobe Systems Inc. et de la SOFTtools Generation Inc. !

s'intéresse au choix des méthodes, et le système de calcul compile ces connaissances. Dans cette optique, il est intéressant de disposer d'un "macro-langage" plus "Naturel" pour l'expression de méthodes, tactiques ou stratégies. Pour comparaison, l'expression des règles de Mathematica utilise une forme plus évoluée que le langage de programmation fourni.

- **mise à dispositions d'outils** pour satisfaire des besoins spécifiques ; par exemple, connaître et respecter la précision du calcul (les chiffres significatifs). Il faut pour cela la "pister" tout au long du calcul et implanter quelques heuristiques pour augmenter et rétropropager le nombre de chiffres de travail aux endroits les plus "contraints". Un autre besoin consiste à gérer les conditions de validité du résultat des calculs, afin de ne pas produire de résultats vrais "presque partout".

- **prise en compte du comportement de l'usager.** Cela va de l'existence d'un fichier de notations de fonctions personnalisées, à l'observation, l'étude et la connaissance de certaines de ses actions conceptuelles. Ceci est toutefois limité par le niveau conceptuel du système, c'est-à-dire au mieux ici, à l'expression explicite de tactiques ou de stratégies d'application de (séquences de) commandes.

- **favoriser l'assistance à la résolution.** Ici encore, les points précédents y contribuent en se concentrant sur les phases de formulation et de présentation de problème, quitte à ne pouvoir aider qu'occasionnellement.

- **servir de base** (tel un langage "assembleur") à d'autres outils comme des Systèmes-Experts, systèmes de visualisation, etc. En effet, outre sa capacité d'adaptation propre, la puissance de l'outil alliée à son langage spécifique en font un partenaire recherché pour exploiter ses capacités.

- **intervenir dans une boite à outils** à l'usage des mathématiciens offrant des facilités telles que le courrier pour documents mathématiques, des bases de théorèmes, des bibliothèques de méthodes, des encyclopédies ou manuels de référence, des "espaces" de discussions mathématiques, des systèmes d'enseignement, des outils de production de démonstration, etc.

Malgré des recoupements indispensables, nous penchons pour la **conception de systèmes mathématiques différents** des systèmes de calcul formel actuels. Nous noterons alors pour le mathématicien, selon les domaines d'application :

- en ce qui concerne les domaines habituellement réservés au calcul, tout système qui ne s'intéresse pas directement au résultat. Nous ajouterons aussi un système de calcul étendu en vue d'une assistance aux manipulations, pour les cas où une partie non négligeable de la résolution du problème est effectué par les manipulations de l'usager.
- pour les autres domaines mathématiques où l'essentiel de la résolution est la gestion du raisonnement, des sytèmes plus orientés vers les manipulations d'énoncés mathématiques, de théorèmes et de preuves.
- pour le traitement de l'activité mathématique, telle que nous l'avons définie dans son acceptation générale, des systèmes d'assistance aux finalités multiples, telles que nous allons les aborder dans le reste de ce chapitre.

Dans ces trois cas, l'étude et la prise en compte du Langage Mathématique et des manipulations de formules, joue un rôle déterminant pour ces systèmes.

1.2.2. Les systèmes de production de théorèmes

Les systèmes de production de théorèmes sont de deux types : automatiques ou assistés. Les systèmes de démonstration automatique de théorème (DAT) ont initialement permis d'informatiser des systèmes formels d'inspiration logique. Ils sont donc adaptés dès qu'une théorie logique est satisfaisante pour une application mathématique particulière.

Souvent, l'apport de la logique est beaucoup plus indirect : par exemple, le principe de résolution de Robinson est le fondement théorique d'un programme de manipulation symbolique écrit en PROLOG ! Les systèmes de DAT à usage des mathématiques sont notamment présentés et analysés dans [Pastre 89]. Leurs principes s'inspirent actuellement d'une synthèse d'approches très différentes : règles et méta-règles, méthodes graphiques [Merialdo 79], représentations hybrides avec objets et règles [Laublet 91], etc.

La difficulté essentielle de la production de théorèmes provient de l'impact souvent déterminant de capacités créatives peu maitrisées. Par exemple, la démonstration que l'ensemble des nombres premiers est infini introduit le procédé de génération d'un nombre premier plus grand qu'un nombre donné n comme diviseur de n!+1. Même si le recours à ce procédé de génération fait partie de connaissances générales, d'où provient l'idée de la construction "n!+1" ?

La distinction entre trouver une preuve et vérifier une preuve est donc cruciale. Dominique Pastre recense et illustre alors trois solutions pour aider le système de DAT à trouver une preuve des théorèmes difficiles [Pastre 90a] :

- l'introduction de lemmes intermédiaires, spécifiques à la preuve ;
- l'interactivité, qui consisterait par exemple à introduire "n!+1" ;
- l'introduction de connaissances générales au domaine d'application ;

Ces trois solutions sont complémentaires :

- l'introduction de lemmes intermédiaires revient à laisser au mathématicien le soin de définir les étapes difficiles de la production de la preuve. C'est ce qui se fait, par exemple, dans les problèmes scolaires. Cela suppose néanmoins que le problème est bien posé, et que le mathématicien sait déjà comment le résoudre. C'est donc principalement une facilité de construction de preuve.

- la gestion de l'interactivité est essentielle dès que le système n'est pas entièrement automatique. Nous avons par exemple recherché que l'interaction soit au centre de la problématique d'utilisation d'un système, et ne se résume pas à relancer une démonstration avec des données légèrement étendues.

- l'introduction de connaissances générales au domaine d'application est l'ambition d'un système bien conçu. La difficulté - note D. Pastre - est de rassembler ces connaissances : « L'observation de mathématiciens est utile mais insuffisante parce qu'une énorme quantité de connaissances est nécessaire et qu'une grande partie est inconsciente. »

Notre approche expérimentale Intercom consiste à soutenir que ces connaissances ne peuvent être explicitées que par la pratique, grâce à un système de construction de preuves. Elle propose de plus un paradigme d'analyse de connaissances implicites au moyen d'une linguistique des mathématiques. Elle suppose enfin que les connaissances relatives au raisonnement sont incomplètes et n'aboutissent que dans les cas favorables. Le guidage du raisonnement est alors aussi important que son automatisation pour les tâches élémentaires.

Notre analyse consiste donc à noter que **les systèmes de DAT ont validé leurs capacités à démontrer** des théorèmes difficiles, mais n'ont pas proposé de solution satisfaisante pour leur utilisation sur des problèmes plus étendus. Le principe d'assistance que nous avons proposé pour utiliser les même techniques à une échelle plus large soulève alors des difficultés et des enjeux différents. Ce fut l'objet de notre travail des deux parties précédentes.

Dans cette optique, notre étude se rapproche de la problématique de la construction de théorèmes à des fins de vérification formelle. Cette autre façon de produire des théorèmes s'inspire des travaux de N. G. De Bruijn sur le langage AUTOMATH [De Bruijn 68] et privilégie l'écriture de la preuve sur le raisonnement global de résolution. C'est ce qui se passe dans des systèmes comme Nuprl [Constable 86] ou [Formel 89]. L'analyse du système interactif [Fallot 89] a permis de montrer les difficultés rencontrées pour faire, de la sorte, des mathématiques à l'usage du mathématicien. La vérification formelle de la preuve est d'ailleurs souvent auxiliaire dans l'activité du mathématicien. La construction et la présentation interactive des preuves est, par contre, essentielle.

1.2.3. Les systèmes experts en raisonnement

Un usage fréquent de la DAT n'est plus alors la capacité de prouver des théorèmes difficiles, mais le moyen d'expérimenter et de valider des raisonnements en fonction de leurs stratégies de guidage et d'une façon d'organiser les connaissances. Leur problématique relève alors d'une réflexion sur les techniques informatiques susceptibles de mener à bien des raisonnements sophistiqués : ce n'est plus le théorème qui est important, mais le raisonnement qui l'a obtenu !

Le paradigme dominant de la décennie 1980, lorsque l'on parle de raisonnement, est celui des Systèmes-Experts (S.E.). Il a permis la réalisation de systèmes appliqués à de nombreux domaines mathématiques. Par contre, ces systèmes restent très spécialisés dans le type de problèmes qu'ils résolvent. Il n'est donc pas apparu de système généraliste, comparable aux systèmes de calcul formel. Il n'existe donc pas actuellement de grande "base de connaissances mathématiques" qui serait utilisable en complément d'algorithmes de calcul formel.

L'avantage d'un Système-Expert est de pouvoir contrôler précisément sa démarche de raisonnement. Lorsqu'ils s'appliquent à des manipulations algébriques comparables aux systèmes de calcul formel, les S.E. permettent en particulier :

- de résoudre, par leur raisonnement, des problèmes pour lequel il n'y a pas de procédure de décision satisfaisante.
- de résoudre des problèmes en utilisant une démarche simple et décomposable (facile à expliquer).

Le lecteur intéressé par ce domaine peut consulter la thèse de J.F. Nicaud [Nicaud 87], décrivant son S.E. en factorisation des polynomes : APLUSIX. Il y présente de plus les connaissances et le cycle de raisonnement utilisés pour SEME, CAMELIA, TANGO et PRESS[7].

Les connaissances opératoires en manipulations algébriques étant considérables, Martial Vivet a eu l'originalité de coupler calcul et raisonnement dans CAMELIA [Vivet 84]. Il a adapté un module de raisonnement sur un sytème de calcul formel existant, REDUCE, en vue d'élaborer un plan de résolution grâce à des méta-règles stratégiques.

La maîtrise du raisonnement constitue l'un des défis initiaux de l'IA et fait ainsi l'objet de recherches spécifiques. Le mathématicien utilise dans son activité de nombreuses formes de raisonnement (déduction, induction, par analogie, visuel, etc.) que des systèmes cherchent à simuler. Le point crucial de l'expertise de résolution du mathématicien consiste à guider le raisonnement, et la partie transcrite dans ses productions écrites n'en restitue qu'une faible part. La découverte et le raisonnement en mathématiques sont donc un thème d'investigation privilégié de l'IA.

L'automatisation de ce thème est notamment discutée par Alan Bundy [Bundy 85]. Il y aborde quelques techniques utilisables (entre parenthèses) pour :

- la preuve de théorèmes (déduction et contrôle des recherches) ;
- la formalisation de problèmes (extraction de formules à partir de l'énoncé) ;
- l'apprentissage (assimilation, analyse de preuves, induction à partir d'exemples) ;
- l'utilisation de raisonnements par analogie (appariement).

Les systèmes qu'il présente pour cela illustrent les potentialités, mais laissent entrevoir que cette automatisation est limitée à certains types de problèmes. La difficulté des systèmes experts en raisonnement mathématique est qu'on ne sait pas évaluer a priori dans quelle mesure ils sont applicables pour d'autres problèmes. Cela dépend notamment de la nature des connaissances disponibles et de l'adaptabilité de leurs stratégies de raisonnement. Quoi qu'il en soit, la mise au point des systèmes de raisonnement est longue et minutieuse, et leur transfert vers un domaine différent s'apprécie cas par cas. Une forte composante empirique substiste donc...

Une technique essentielle de raisonnement déductif consiste à raisonner au méta-niveau (la conception du système devient multi-ordinale). Le méta-niveau d'une théorie consiste à pouvoir raisonner sur la théorie. Lorsqu'un mathématicien démontre un théorème dans une théorie, il peut se poser les conditions de sa validité, c'est-à-dire savoir quels sont les axiomes ou les sous-théories qu'il utilise, savoir aussi quelle règle de démonstration a été utilisée, savoir enfin, par exemple, si l'existence est constructive et fournit l'élément cherché[8].

---

[7] Le raisonnement de chaque système est présenté sur un exemple. SEME (M. Baron) évalue $\sum_{k=0}^{n} \frac{(-1)^k}{k+1} C_n^k$; CAMELIA calcule $\int (\text{tg } x)(\text{Log cos } x)\, dx$; TANGO calcule $\int \text{tg}^3 x\, dx$ et PRESS résoud $ax^2+bx+c=0$. Leur intérêt principal est le raisonnement suivi. Ainsi, TANGO est un moteur d'inférences de S.E. qui a été appliqué au calcul des primitives par B. Faller et L. Pottier. De même, PRESS trouve la résolution de cette équation en raisonnant à partir de $(a+b)^2=a^2+2ab+b^2$.

[8] Roger Cuppens [Cuppens 87] analyse ainsi quatre (méthodes de) démonstrations du problème : "soient x et b deux entiers naturels tels que b≥2, montrer l'existence et l'unicité d'un entier n tel que $b^n \leq x < b^{n+1}$ ". Notons au

Néanmoins, le méta-niveau habituellement utilisé dans les systèmes ne porte pas sur la théorie, mais sur la démarche de résolution du problème. Les avantages attendus de cette séparation consistent en une réduction de l'espace de recherche, et en une puissance d'expression, une souplesse, une modularité, une déclarativité, une explicabilité et une lisibilité accrues. Bref, si l'intérêt d'une séparation est avéré, sa mise en œuvre et ses résultats restent soumis à appréciation...

Raisonner au méta-niveau permet au mathématicien d'appliquer des tactiques ou des stratégies de démonstration. Les travaux autour du système PRESS [Bundy 83, Sterling 89], illustrent l'intérêt d'une telle approche pour la résolution d'équations, en structurant les parties informelles du raisonnement par des stratégies d'isolation, collection, attraction, homogénéisation, etc. Cela permet aussi par exemple de généraliser un théorème, selon les techniques esquissées section suivante.

Néanmoins, la modélisation des raisonnements des mathématiciens est souvent biaisée par notre opinion a priori (ou notre étude introspective). C'est ce qui fait l'originalité de la démarche de D. Pastre qui a observé le comportement des mathématiciens sur des problèmes variés [Pastre 78], en a extrait les éléments utiles pour d'autres démonstrations, puis a simulé certaines connaissances sur MUSCADET [Pastre 84] à l'aide de règles, méta-actions et méta-règles utilisées indépendemment par le même moteur d'inférences.

Il semble que seules certaines parties du raisonnement déductif soient automatisables pour des problèmes mathématiques. Par ailleurs, d'autres formes de raisonnements peuvent être utilisées avec profit pour des tâches spécifiques. De plus, la difficulté d'une résolution est différente selon le problème et selon l'agent (homme ou machine) [Pastre 90b]. Si une approche d'assistance semble indispensable dès que le domaine d'application est non restreint, il est irréaliste de fixer a priori les limites des tâches respectives : les tâches élémentaires que nous avons étudiées doivent toutefois être prises en compte, et l'étude de l'activité du mathématicien poursuivie.

1.2.4. Les systèmes qui découvrent ou apprennent

La majeure partie de l'activité du mathématicien est souvent éclipsée par l'attrait psychologique des systèmes de calcul ou de raisonnement. Les systèmes de découverte ou d'apprentissage s'attachent par exemple à d'autres aspects essentiels de l'activité purement mathématique. Ils peuvent **s'appliquer aux trois pôles** : langage, connaissances ou raisonnement, bien que le langage ne soit pas représenté effectivement.

Notre étude n'analyse pas les mécanismes de découverte ou d'apprentissage, bien qu'elle prenne en compte leur existence et leur intérêt. Nous partageons les positions empiristes concernant le développement des mathématiques, bien argumentées par George Pólya et Imre Lakatos. De nombreux exemples mathématiques se prêtant à un raisonnement par généralisation, spécialisation et analogie sont ainsi analysés dans [Pólya 86] ; le mécanisme d'évolution des théories mathématiques est illustré dans [Lakatos 84, Rissland 78], etc.

La plupart de ces mécanismes resteront forcément **sous guidage du mathématicien**, et devront par ailleurs lui rendre compte de leurs choix et de leurs résultats. Ils servent dans les processus créatifs lors de la phase d'utilisation d'une théorie, pour gérer l'organisation conceptuelle des connaissances dans la phase de constitution, ou pour faire progresser le mathématicien et la machine lors de la phase d'apprentissage.

L'apprentissage symbolique distingue la détection de similarités (SBL) et l'apprentissage à partir d'explications (EBL) (cf. par ex. [Greiner 88]). La SBL cherche à classer et organiser des exemples à partir de certaines caractéristiques de description. Nous nous intéressons plus particulièrement à l'EBL, qui permet de généraliser un processus de développement.

Les travaux développés à partir de LEX s'appliquent à **généraliser une preuve**, en généralisant les données de la preuve, dans la limite des conditions de validité connues des opérations de déduction. Par exemple, à partir de l'intégration de $\cos^7(x)$, LEX "trouve" que cette preuve est valable pour $\cos^n(x)$, avec n impair. Cela est intéressant dans un contexte intégrant plusieurs types de raisonnements finalisés pour réutiliser les preuves. Il faut alors bien maîtriser les causalités de création des preuves, et utiliser par

---

passage l'intérêt d'un commentaire "périmathématique" décrivant ce problème comme relié au nombre de chiffres de la représentation d'un entier "x" dans une base "b".

exemple la SBL pour signaler des opportunités à étudier. Ces techniques peuvent aussi être utilisées pour la découverte d'heuristiques.

Le système AM de D. Lenat est un système de découverte d'exemples, de conjectures et de concepts en mathématiques [Davis&Lenat 82]. Il a eu une influence considérable et est largement décrit dans la littérature. Le lecteur intéressé par un éventail plus complet de tels systèmes mathématiques peut consulter [Laublet 88].

Les techniques d'amélioration locales sont prometteuses pour l'assistance au mathématicien. Par ailleurs, pour atteindre une taille conséquente, un système opérationnel doit aider à constituer ses connaissances, et parfois les remettre en question de façon à améliorer ses performances. Toutefois, un AMI fournirait avant tout un environnement d'expérimentation de ces techniques et favoriserait l'émergence de nouveaux besoins. Les développements des recherches seraient plutôt concomittants. L'important est d'abord de **trouver et mettre en place informatiquement les situations** dans lesquelles ces techniques soient applicables.

1.2.5. Les systèmes d'enseignement et de modélisation cognitive

Alors que les systèmes précédents pouvaient garder une certaine distance avec la **modélisation cognitive**, il n'en est pas de même des systèmes d'enseignement qui doivent diagnostiquer et remédier aux comportements non conventionnels de leurs étudiants (cf. [Dumont 89]). Ainsi, BUGGY ([Brown 78]) contenait une théorie purement descriptive des "bugs" observables (en soustraction). Cette théorie fut étendue au diagnostic de ces bugs et aux mécanismes cognitifs expliquant leur génèse[9]. Dans ce cas la motivation cognitive a précédé la réalisation de systèmes pour l'enseignement. Souvent, les besoins soulevés lors de la réalisation d'un système implique des études de modélisation cognitive.

Hormis leur couplage avec des études de modélisation cognitive, la réalisation de systèmes dits "***tuteurs intelligents***" (cf. [Sleeman 82]) a suscité de nombreuses innovations dues à leur volonté d'adaptation à l'usager :

- Les tuteurs se sont préoccupés d'instaurer un dialogue en Langage Naturel pour compléter les autres modes d'interaction ;
- Les tuteurs ont créé et utilisé un modèle de l'étudiant ;
- Les tuteurs ont besoin d'une bonne maîtrise de leur domaine de travail, d'où la notion de ***micro-monde*** dans lequel l'étudiant est amené à évoluer. La présence d'un tel monde virtuel est d'ailleurs la seule condition indispensable à une utilisation pédagogique ;
- Les tuteurs effectuent des choix d'enseignement (directif, par découverte, mixte) qui peuvent être insérés dans un raisonnement pédagogique supervisant l'apprentissage de l'étudiant.

Ces points reflètent les quatre macro-composantes (reproduits figure 3.5), identifiées dans [Nicaud 88] lors d'une synthèse sur les systèmes tuteurs intelligents :

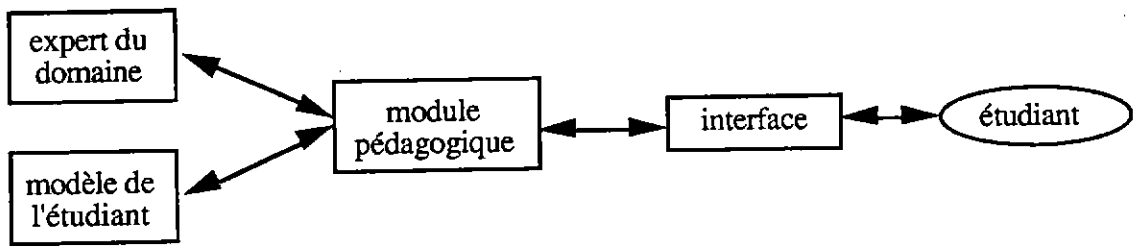


figure 3.5

Le recueil des systèmes et les divers domaines mathématiques abordés, sont répertoriés dans [Bruillard 91]. Les actes [IREM 90] montrent la diversité et la qualité du travail réalisé dans un domaine comme la géométrie. Les solutions techniques sont parfois simples, montrant ainsi la prépondérance de l'adéquation au problème sur une sophistication informatique très coûteuse.

La présence d'une base de connaissances et la possibilité d'opérer dans un domaine mathématique circonscrit permet, par ailleurs, aux systèmes experts en raisonnement d'être utilisés pour l'enseignement.

---

[9] Une présentation synthétique de ces recherches est, par exemple, accessible dans [Cuppens 88a].

Le raisonnement n'est - bien sûr - que l'un des thèmes possibles de l'apprentissage de l'étudiant, et ces systèmes doivent souvent être repensés afin de produire des explications pertinentes. Il est intéressant de noter, en effet, que le rôle privilégié du dialogue sur le résultat à obtenir, introduit une modélisation cognitive plus fine et plus proche de l'usager que celle qui était utilisée auparavant (cf. partie I). Conjointement, les techniques informatiques requises deviennent plus sophistiquées.

Les tuteurs intelligents constituent donc une **source d'innovation** importante en vue de la réalisation d'une assistance au mathématicien et d'un AMI. Par exemple, les systèmes tuteurs sont les seuls à s'être directement investis dans les mécanismes de génération. EDUSYM [Jurkovic 86] ou APLUSIX [Nicaud 87] cherchent à détailler des calculs en fonction du niveau de l'étudiant. De même, [Zukerman 86] cherche à adapter ses conseils pédagogiques en Langage Mathématique, afin de faciliter la compréhension de l'exercice.

Par ailleurs, les systèmes d'enseignement sont le fruit d'une **réflexion centrée sur l'usager** : ils ont évalué l'intérêt que présentent les techniques informatiques disponibles - non seulement pour les exploiter - mais pour répondre avant tout à un besoin pédagogique identifié. Ainsi, ils utilisent fréquemment des mécanismes informatiques sophistiqués pour le guidage de la planification de leurs actions. ***Ils constituent donc les prémisses des Mathématiques Assistées par Ordinateur.***

1.3. LES OUTILS DE FORMULATION ET DE PRESENTATION

1.3.1. Comparaison entre mathématiques et programmation

Nous nous remémorons tous les affres des environnements d'antan proposant un éditeur de fichier qui était compilé puis exécuté, et qui, en cas d'erreur, nous torturait l'imagination par ses messages abscons. Les environnements dédiés des machines Lisp ont permis de goûter aux **environnements interactifs intégrés**. Ils autorisent notamment une mise au point *incrémentale* (morceau par morceau) sur des objets *incomplets*, à l'aide d'éditeurs "intelligents" permettant des calculs (évaluations) pour **remplacer l'édition textuelle** en processus de construction et d'observation de la dynamique de la preuve (= programme).

Ce besoin d'un système **assistant à la programmation** a été exprimé dès 1973 par Terry Winograd : "Breaking the Complexity Barrier (Again)". Il sert d'article d'introduction au livre sur les environnements interactifs concrétisant cette voie la décennie suivante [Barston 84]. Le besoin est actuellement le même pour l'activité mathématique "naturelle"...

L'analogie entre l'activité mathématique et l'activité de programmation a souvent été invoquée à des fins informatiques, pour le Génie Logiciel (cf. par ex. [Bates 85], ou [De Millo 79] et sa critique [Fetzer 88]). Pour cela, le développement d'un programme est comparé au développement d'une preuve. Les structures de données du programme correspondent alors aux formules de la preuve. L'évaluation d'un programme est alors une succession de transformations mathématiques permettant de construire une solution. La limite de cette analogie réside en ce que les mathématiques définissent une preuve en précisant les données (les formules) mieux que les transitions (il n'y a pas "d'instructions du langage").

Cette analogie peut, à son tour, servir les mathématiques :

- pour les outils : récupérer les outils et techniques développées pour la programmation ;
- pour les problèmes : l'industrie informatique peut réaliser qu'une production à un haut niveau d'abstraction peut s'enrichir d'une comparaison avec l'usage pratique du Langage Mathématique.

D'autre part, l'informatique dispose d'un avantage déterminant sur les mathématiques, quant à son potentiel de création d'un système d'assistance :

- l'informatique est organisée en industrie ;
- les besoins sont immédiats et chiffrés financièrement...
- il y a appariement direct entre les besoins empiriquement dégagés et les concepteurs potentiels.

Il est ainsi significatif de constater que la plupart des techniques et des logiciels innovants ont été d'abord conçus pour faciliter la programmation. Alors que les travaux issus du Langage Naturel concernent principalement la modélisation et la représentation de connaissances, les langages de programmation contribuent actuellement aux **manipulations d'objets structurés**. Les objets mathématiques structurés, construits lors de la phase d'utilisation, se retrouvent à deux niveaux : formules et preuves.

Par contre, tous les aspects informatiques du système d'assistance introduisent, à leur tour, des structures spécifiques.

La diversité des structures à prendre en compte entraîne une **diversité** des points de vue sur la construction d'un programme ou d'une preuve. A chaque point de vue correspond un jeu d'outils intégrés, à commencer par des moyens d'édition graphique. Ainsi, la qualité des visualisations proposées est sans doute l'un des facteurs du succès du générateur de S.E. Nexpert-Object™. De plus, chaque point de vue est aussi associé à un *langage de description* approprié. Une tâche de construction est alors intrinsèquement associé à plusieurs langages, d'où l'intérêt d'*outils génériques* de traitement, indépendants d'un formalisme particulier.

CENTAUR [Borras 87] est l'un de ces systèmes prometteurs appelés **environnements interactifs génériques**. Ils produisent notamment un éditeur structurel, un analyseur de type et un interpréteur-dévermineur d'un langage exprimé à partir des spécifications de sa syntaxe et de sa sémantique. La puissance du système dépend des langages de descriptions proposés et est en constante amélioration.

Un tel mécanisme de généricité est **indispensable** en mathématiques puisqu'un langage est plutôt défini à partir des objets qui le composent. Pour introduire dynamiquement des définitions et des notations, le mathématicien est amené à spécifier leurs propriétés. Lorsqu'on change de personne, de problème ou de théorie, le jeu de concepts et de notations est aussi à redéfinir. De plus ces environnements sont multi-langages et peuvent servir de passerelle entre les mathématiques et des langages de programmation.

Tous ces outils logiciels demandent un développement considérable en homme/années. Sans système pour concrétiser notre recherche, nous avons considéré que ces techniques étaient disponibles afin de présenter, dans cette dernière partie, une description de certaines spécificités des domaines mathématiques. Nous l'avons voulue **déclarative** jusqu'à un niveau d'abstraction général, afin qu'elle puisse être adaptée à un environnement interactif générique à venir.

1.3.2. Les systèmes de traitement de texte et de formules

Jusqu'à la décennie 1980, il était difficile de produire des documents mathématiques typographiés : rien ne semblait remettre en question l'écriture dactylographiée. Par ailleurs, les tentatives informatiques de ramener l'écriture mathématique à une saisie clavier de caractères ASCII aboutissaient à des produits désormais oubliés.

C'est dans ce contexte que Donald E. Knuth produisit la première avancée significative vers une prise en compte textuelle du Langage Mathématique [Knuth 79]. Son langage informatique TEX permettait de décrire les aspects graphiques spécifiques aux mathématiques : il gère automatiquement la composition du texte ainsi que certaines caractéristiques typographiques telles la taille plus petite des indices ou des bornes. Il représente la **structure du texte** (telle l'auteur, le titre, etc. pour la bibliographie), et calcule la présentation graphique finale en utilisant son savoir-faire typographique.

Toutefois, ce langage TEX était si spécialisé et si complexe, que les outils successifs n'ont eu de cesse de le remplacer par des moyens d'expression plus accessibles. Ce fut notamment le cas d'un des premiers logiciels interactifs de traitement des formules : EDIMATH [Quint 83]. Il utilise un langage simple ainsi que des facilités de construction graphiques. L'état des lieux actuel permet de construire des intégrales ou des matrices avec presque autant de facilité que des chiffres, car les paramètres supplémentaires à saisir sont facilement exprimés. En une décennie, la production informatique de documents mathématiques est devenue à la portée de tous !

La forme étant maîtrisée, il semble alors qu'aucun obstacle ne puisse tempérer l'ardeur des mathématiciens. Pourtant, la production de documents respectant une cohérence mathématique accessible par une machine reste un casse-tête à la charge du mathématicien. Comment le mathématicien va-t-il procéder actuellement pour rédiger sa démonstration ? Le lecteur est invité à prendre des exemples comme :

Démonstration:
Montrons l'associativité du produit :

$$a*b(x,\omega) = \int a(y,\omega)\, b(-y+x,-y.\omega)\, \sigma(y,-y+x)\, dy \quad (1)$$

$$[(a*b)*c](x,\omega) = \int (a*b)(z,\omega)\, c(-z+x,-z.\omega)\, \sigma(z,-z+x)\, dz \quad (2)$$

$$[(a*b)*c](x,\omega) = \int\int a(y,\omega)\, b(-y+z,-y.\omega)\, \sigma(y,-y+z)\, c(-z+x,-z.\omega)\, \sigma(z,-z+x)\, dy\, dz \quad (\ldots) \quad (3)$$

Il est plus facile de produire cela en s'aidant d'opérations mathématiques, à condition que le système puisse faciliter cette démarche et gérer les paramètres. Comme pour la maîtrise typographique, cette possibilité requiert deux conditions :

- la capacité intrinsèque d'effectuer ces opérations mathématiques et paramathématiques ;
- la facilité de commande et de mise en œuvre, nécessitant des connaissances pragmatiques spécifiques à l'assistance.

De même, le changement de variable de j en j-1, utilisé partie II dans la démonstration de la formule du binôme est :

$$\sum_{j=0}^{n-1} C_n^j \, a^{j+1} \, b^{n-j} = \sum_{j=1}^{n} C_n^{j-1} \, a^j \, b^{n-j+1}$$

alors que celui obtenu par le présent traitement de texte est : $= \sum_{j-1=0}^{n-1} C_n^{j-1} \, a^{j-1+1} \, b^{n-j-1}$

Pour obtenir un résultat convenable, la structure typographique du texte est insuffisante, il faut utiliser une structure de représentation qui isole les caractéristiques mathématiques qui sont manipulées. Prenons l'exemple :

$$\int tg(t)\, dt$$

et faisons le changement de variable t en y+a. Le remplacement fourni par mon système de traitement de texte donne :

$$\int y+ag(y+a)\, dy+a$$

Il faut donc **reconnaître les symboles mathématiques** et **gérer les parenthésages**, le remplacement donne alors :

$$\int tg(y+a)\, d(y+a)$$

Si ce remplacement sait que "a" est un symbole de constante, et sait **effectuer des simplifications élémentaires**, il peut obtenir directement :

$$\int tg(y+a)\, dy$$

Enfin, s'il a les **connaissances mathématiques** supplémentaires, il peut gérer les bornes de l'intégrale, et exploiter les propriétés de la fonction tangente lorsque $a=\pi$[10]. Ces connaissances peuvent gérer, par ailleurs, les aspects syntaxiques afin d'éviter les collisions de symboles, tels les "t" externes à l'intégrale ou la présence éventuelle de "y" dans les bornes. Ces connaissances spécialisées sont nécessaires pour participer à la présentation et à la validité mathématique afin d'obtenir comme forme visuelle :

$$\exists x, \forall y, \forall z, q(x,y,z) \Rightarrow \forall y, \forall z, \exists x, q(x,y,z)$$

alors que le système de calcul utilisé aurait obtenu :

$$\exists l_1, \forall l_2, \forall l_3, q(l_1,l_2,l_3) \Rightarrow \forall l_1, \forall l_2, \exists l_3, q(l_3,l_1,l_2).$$

Enfin, ces connaissances ne concernent pas uniquement les formules, mais aussi la gestion d'une preuve et d'une démonstration afin de produire des déductions comme :

$MA = \rho$
or $MA = p-\rho\cos(\pi-\theta)$,
d'où $\rho = p+\rho\cdot\cos\theta$.

Ces manipulations n'imposent pas de travailler uniquement avec des opérations mathématiques. Ainsi, lors de l'introduction des données d'un problème, il arrive fréquemment qu'il soit plus rapide de les saisir en **récupérant, puis en modifiant les données** d'un problème similaire, rapidement accessible.

De telles manipulations "extra-mathématiques" sont aussi envisageables pour faciliter les moyens d'expression dans une résolution. Ainsi, nous avons abondamment utilisé, partie II, la possibilité de

[10] Le lecteur informaticien notera la similarité des problèmes et des techniques à employer avec celles des environnements de programmation structurée, l'évaluation remplaçant alors le calcul mathématique.

sélectionner des sous-expressions pour fournir des arguments ou délimiter les zones d'intérêt. Ces sélections peuvent s'accompagner de "couper-coller" comparables à ceux des traitements de texte. Par exemple, pour indiquer la voie qu'il choisit dans la relation entre les coefficients du binôme :

Le second membre donne : $\frac{(n-1)!}{(p-1)!\,(n-p)!} + \frac{(n-1)!}{p!\,(n-1-p)!}$

le mathématicien peut soit indiquer une opération de mise en facteur totale, soit recopier le terme de gauche, insérer "-1" dans "(n-p)!", et demander au système la mise en facteur du terme ainsi obtenu :

en factorisant on obtient : $\frac{(n-1)!}{(p-1)!\,(n-p-1)!}$ XXX (le "XXX" résulte de la complétion du système)

Les opérations qu'utilisent usuellement le mathématicien n'ont donc qu'une vague similarité avec des opérations réalisables textuellement. Il est donc normal que l'intérêt que les concepteurs d'outils informatiques portent à la communauté scientifique les amènent à se rapprocher des besoins. Nous avons donc proposé des extensions utiles aux systèmes de traitement de texte et aux systèmes de calcul, selon l'état des lieux figure 3.6 :

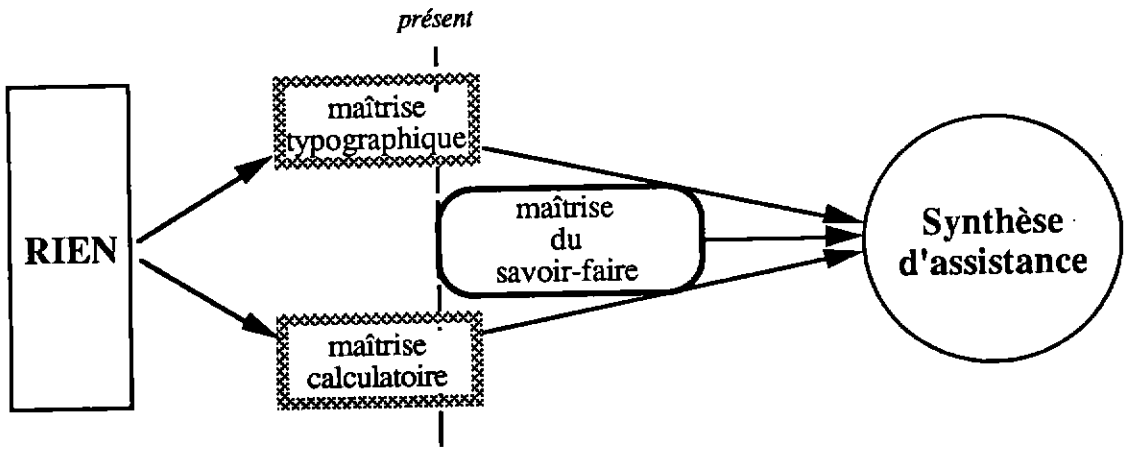


figure 3.6

La compréhension et le raisonnement escomptés du système d'assistance permettent une grande richesse de transformations mathématiques ou textuelles (il est toujours possible d'agir à ce niveau). En particulier, dès qu'il y a corrélation d'une nouvelle formule avec une (ou des) ancienne, comme cela est usuel, il est souvent préférable d'expliciter la nature du lien - et laisser le système effectuer seul la transformation - que de détailler la nouvelle formule. Un texte mathématique peut ainsi être plus aisément rédigé, ***pourvu que le système en perçoive le fil directeur.***

C'est sur ce point, en effet, que l'intérêt d'une assistance doit être discuté. La plupart des productions scientifiques des mathématiciens sont trop sophistiquées et trop abrégées pour qu'un système puisse raisonnablement escompter les effectuer. D'autre part, les besoins soulevés par une telle assistance permettent d'envisager mieux qu'une édition du texte (cf. programmation). Il reste alors essentiellement deux voies :

- ajouter des fonctionnalités susceptibles d'aider le mathématicien dans son édition de texte ;
- décomposer les productions jusqu'à un niveau traitable par une machine.

Cette deuxième voie que nous analysons dans cette thèse, ne devient intéressante que si le système peut **prendre en charge une plus grande part de l'activité du mathématicien**. Cela correspond d'ailleurs au besoin même des mathématiques. L'exemple [Graham 89, p33] de la partie I-5, montre que les résultats mathématiques ne sont pas réutilisables - en général - sans disposer de leur preuve. Ainsi, lorsque le mathématicien dispose du résultat :

$$\sum_{0 \le k \le n} k\,2^k = (n-1)\,2^{n+1} + 2$$

il ne peut rien conclure sur : $\sum_{0\le k\le n} k\,x^k$

sans connaître précisément le rôle de 2^{k+1} dans la preuve qui l'a produit (voire par un raisonnement englobant le problème lui-même). C'est en particulier en traçant les opérations utilisées pour la preuve, que le mathématicien - ou le système - aboutit au problème consistant à résoudre l'équation :

$$S_n + (n+1)\,x^{n+1} = xS_n + (x - x^{n+2}) / (1 - x)$$

ce qui donne : (S_n=) $$\sum_{0\le k\le n} k\,x^k = \frac{x - (n+1)x^{n+1} + nx^{n+2}}{(1 - x)^2} \quad (\text{pour } x\neq 1)$$

Nous nous sommes donc progressivement éloignés des systèmes de manipulation de documents, dont l'extension des fonctionnalités - pour ce qui concerne l'usage mathématique et scientifique - ne s'effectuera que par une prise en compte plus importante des caractéristiques de l'activité scientifique concernée.

1.3.3. Les systèmes d'aide, de documentation, d'archivage et de dialogue

1.3.3.1. Le paradigme hypertexte

La gestion de **l'accès et de l'utilisation d'une grande quantité d'information** est un problème général. La spécificité des mathématiques est qu'une cohérence beaucoup plus "formelle" peut être établie entre ces items d'information. Les problèmes abordés pour la documentation sur un monde mal précisé, peuvent donc éventuellement être réévalués lorsque l'interdépendance entre les informations disponibles devient forte.

La réponse actuelle à ce problème d'exploitation de l'information est le paradigme hypertexte. Les hypertextes (c.-à-d. textes non-linéraires), et plus généralement les hypermédia (texte, son, images), constituent un domaine d'activité visant à libérer l'information d'une organisation hiérarchique donnée. L'article de Conklin [Conklin 87] présente une introduction et un panorama de référence. Les numéros spéciaux des CACM (juillet 88) et TSI (vol 9, n°6, 1990) présentent aussi plusieurs réalisations, ainsi que des réflexions sur l'état des recherches. Par ailleurs, la diffusion d'Hypercard a contribué à sensibiliser la communauté scientifique.

La philosophie des hypermédia permet de réunir des documents non-linéaires en une structure souple permettant de multiples points de vue. Cela permet notamment :

- la **consultation et l'analyse d'informations** selon les besoins et l'inspiration, comme pour une encyclopédie ou un manuel technique (par ex. la documentation de mathematica). Cette souplesse se prête bien à la phase exploratoire d'un problème, en autorisant une pensée peu structurée par association d'idées.
- une certaine forme **d'enseignement** encyclopédique, comme l'utilisation d'un atlas ou/et une chronologie historique. La thèse [Bruillard 91] cherche à appliquer la vision hypertexte aux outils d'enseignement en mathématiques.
- une **intercommunication** de type "brassage d'idées" entre intervenants sur un projet, par exemple l'écriture, l'argumentation et la critique d'une "télépublication" scientifique.

Les hypertextes se caractérisent par l'intégration de plusieurs points :

- base de données de *nœuds* ;
- divers types de *liens* entre les nœuds, formant un réseau d'interconnection ;
- un mode de **navigation** et d'**interaction visuelle** élaboré comprenant :
  - un gestionnaire de fenêtres, icônes, menus, souris ;
  - un "*browser*" pour présenter globalement ou localement le réseau ;
  - la possibilité d'accéder aisément à d'autre nœuds via le browser, les liens ou une requête ;
  - la notion de parcours qui traduit la navigation choisie.

Une analyse plus fine permet de distinguer :

- des **nœuds** d'information, typés selon la nature des données (texte, graphique, animation, action via une programme, senteur!?, etc.) ;

- des **liens** (avec nom, type et autres propriétés). La nature du lien est le seul moyen de représenter la relation sémantique entre deux nœuds. Pour un débat, les liens introduits sont par exemple : répond à, questionne, renforce, objecte, spécialise, généralise, réfère, remplace, est suggéré par, etc. Des raisonnements peuvent alors être effectués grâce à ces informations ;
- un aspect **évolutif**, par l'ajout de nœuds et de liens correspondant à une réorganisation ou à d'autres informations ;
- un aspect **multi-utilisateurs**, via la consultation simultanée du réseau comme en Base de Données, et sa **personnalisation** en fonction d'un utilisateur, d'un thème (prise en compte d'un thème comme filtrage contextuel des liens) ;
- des moyens de **regrouper des nœuds**, eux-même réalisés à l'aide de nœuds ou de liens ;
- des **facilités** (liens gérés par le système...) telles que l'historique des versions d'un nœud, d'un parcours (nœuds visités, points en suspend, motivations, etc.), vue partielle selon un thème, etc.

La présentation traditionnelle des hypertextes consiste à associer une fenêtre à chaque nœud. Cela présente les mêmes **limitations** que s'il s'agissait de produire un document par assemblage de paragraphes prédéfinis. Par ailleurs, le choix de représentation est crucial : quelle structure adopter pour les liens, faut-il expliciter ou déduire telle information, si oui comment, etc. Cette approche butte de plus sur les problèmes classiques d'efficacité, cohérence, modification, etc.

1.3.3.1. Les développements possibles

A partir de réflexions sur Notecards, F.G. Halasz [Halasz 88] relève sept points à prendre en compte pour la **prochaine génération d'hypermédia** :

- un accès par requête prédicative sur le contenu ou la structure de l'information (cf. l'analogie avec les réponses coopératives en Base de Données [Cuppens F. 88]) ;
- une extension du modèle afin de considérer un réseau comme un nœud et un nœud comme composition de constituants (afin de permettre l'abstraction et la composition, cf. par ex. [Beaudoin-Lafon 85]) ;
- une reconfiguration dynamique des liens ou nœuds par une définition intensionnelle et non extensionnelle ;
- des capacités de calcul sur le réseau pour un traitement de l'information ;
- la gestion des versions ;
- un travail d'équipe via un accès multi-utilisateur et des moyens d'interaction sociale ;
- des capacités d'adaptation et d'extension ;

Globalement, ces points traduisent un enrichissement de la nature et de la structure de l'information, en vue d'une capacité de traitement et de meilleures caractéristiques dynamiques. On peut résumer la **transition** de génération par le passage d'un point de vue *dynamique* sur une représentation *statique*, **vers un point de vue *dynamique* sur une représentation *dynamique***.

La génération actuelle d'hypertexte se contente de fournir des informations visualisables à l'aide d'outils de navigation sophistiqués. Le contrôle est ainsi laissé à l'utilisateur qui est le seul à pouvoir interpréter une information et savoir si elle correspond à ce qu'il cherche. Cette approche suppose des domaines d'application où l'ordinateur n'a pas les moyens de traiter la connaissance relative à ces informations, c'est-à-dire où il se contente d'archiver et de présenter, bref d'**exploiter l'information**. Par le vocable "**connaissance**", nous distinguons de cette forme inerte dite "information", l'information associée à une clé qui permet de l'utiliser. Lorsque cette clé est accessible à la machine, l'ordinateur peut effectuer un traitement incluant la sémantique de l'information.

Une fois ce cadre de système d'information ainsi défini, les **inconvénients majeurs** des hypermédia sont de désorienter (faire perdre le fil conducteur), et d'impliquer une surcharge cognitive d'appréhension du réseau pour la création de nouveaux liens [Conklin 87]. Nous ajouterons l'aspect fastidieux, voire irréaliste de la conception et la réalisation d'un grand nombre de liens, ceux-ci ne pouvant être créés que par une intervention humaine directe.

Les remèdes possibles consistent d'une part à améliorer l'interaction homme-ordinateur et d'autre part à pallier au manque de **connaissance sémantique**. L'analogie avec les réseaux sémantiques (où des calculs peuvent être effectués sur les nœuds et les liens) montre la dimension manquante des hypermédia. En effet, la plupart des applications actuelles se contentent d'une simple mise en correspondance syntaxique entre nœuds : l'hypermédia agit sur les liens en considérant les nœuds comme une matière inerte.

Or, il est envisageable d'inverser le procédé en faisant jouer aux nœuds un rôle sémantiquement actif, qui induise la création de liens entre eux selon tel ou tel point de vue. C'est ce qui se passe, par exemple, lors de la construction d'une preuve ou lors de son parcours ultérieur. Cela permet de **distinguer la représentation, du point de vue** selon lequel les connaissances sont utilisées.

Nous aboutissons ainsi à un cadre étendu que nous distinguerons par le vocable ***métamédia***, pour sa connaissance du contenu de l'information. L'interaction devient elle-même assistée par ordinateur : l'homme et la machine coopèrent dans une tâche d'**exploitation de la connaissance**. Cette approche métamédia s'insère dans la lignée des sept points précédemment exposés de la prochaine génération d'hypertextes. Elle allie de bonnes caractéristiques dynamiques à un traitement basé sur une représentation des connaissances plus sophistiquée.

Le mode d'exploitation "libre" des connaissances implique une atomicité suffisante des nœuds, et des mécanismes de génération permettant de les lier selon tel ou tel parcours. Contrairement à [Halasz 88], qui se contente d'esquisser la notion de rhétorique pour la prochaine génération d'hypermédia, nous considérons que cette **rhétorique de présentation** constitue l'un des points clés des métamédia. Ce concept est d'ailleurs incontournable pour tout système interactif. Par exemple, le deverminage partage cette problématique d'observation et recueil de l'information, et de présentation et d'explication à des niveaux d'abstraction adéquats.

Pour résumer les perspectives de développement, les **fonctionnalités** d'organisation attendues de la part d'un système métamédia sont :

- **associatives** au niveau de l'usage d'un document par des thèmes différents, de liens entre un document et d'autres sur un thème donné, de l'interconnection de groupes de documents, etc.
- **chronologiques** au niveau des versions, de leur datation et d'un contexte général structuré par périodes, etc.
- **classificatoires** au niveau des grands thèmes généraux, des points de vue spécifiques intra ou interdisciplinaires, d'une linéarité (ou arborescence) conceptuelle, d'un mot-clé particulier, etc.
- **méta-classificatoires** au niveau des motivations de création d'un document, de l'introduction d'un point de vue différent, de la réorganisation du contexte général, etc.
- **déductives**, pour la recherche d'information à l'aide de renseignements partiels et disparates, ses capacités à exploiter sa structure souple, produire un document linéaire, etc.
- **inter-active**, par le partage des tâches et plans d'action, une aide à la formulation, la présentation des résultats, les facilités d'édition graphique et de manipulation directe, etc.
- **dynamiques**, par l'adjonction incrémentales de documents, les modifications destructives ou cohérentes de l'existant, etc.

1.3.3.2. Apports possibles des hypertextes aux Mathématiques

L'intérêt des hypermédia pour la communauté scientifique est largement répandu parmi les partisans de ces systèmes. L'article [Conklin 87] présente diverses idées que nous avons adaptées en ce qui concerne les Mathématiques. Ainsi, la notion d'hypertexte est particulièrement appropriée comme support à venir des **traités mathématiques**. En effet, les nombreuses références croisées entre définitions et usage d'autres théories, forment une structure non-linéaire par excellence. De même la quantité d'information, l'usage combiné de figures, notations graphiques, et du texte autour d'un même thème constitue une application rêvée pour les hypermédia.

L'intérêt des hypermédia est de chercher à proposer plusieurs présentations et permettre ainsi plusieurs lectures. La flexibilité de représentations permet de s'adapter à des usages multiples. Cette approche éviterait une présentation monolithique, qui n'est utile que pour des problèmes bien définis, et donc anticipés a priori. Lors d'un débat radiophonique, certains membres de Nicolas Bourbaki reconnaissaient notamment que la définition très rigoureuse a priori de leur programme le rendait peu propice aux adaptations ultérieures (par. ex. pour introduire les catégories) et sous-estimait l'importance de certaines branches des mathématiques (telles la théorie des probabilités ou l'algèbre numérique). Ce plan laissait trop peu de place à l'insertion d'idées et de directions nouvelles.

D'autre part, il est aisé d'imaginer l'utilité d'un **serveur de théorèmes** selon une organisation hypermédia. Un théorème peut :

- être défini macroscopiquement comme issu d'une théorie regroupant définitions et notations ;
- avoir un énoncé avec prémisses et conclusions décomposables ;
- disposer d'une preuve (au moins) dont les étapes non triviales font appel à d'autres théorèmes ou définitions.

- être décrit en fonction de son utilisation et de ses applications ;
- permettre l'accès à ses utilisations connues, etc.

Dans cette hypothèse du serveur de théorèmes, le support principal n'est toujours qu'un texte mathématique, mais les principales articulations de ce support inerte sont mises en valeur par les facilités hypermédia. Vu la production de théorèmes[11], systématiser leur saisie dès la production aurait le mérite de clarifier l'état des lieux, améliorer la productivité, permettre des statistiques, voire détecter certaines incohérences. Ce besoin d'**amélioration qualitative** de l'information par regroupement, organisation, confrontation et communication se retrouve dans tous les domaines expérimentaux comme le logiciel informatique, la chimie, la biologie génétique, etc.

Indépendamment de - ou mieux - intégré à ces banques d'informations, se pose le problème de la **collaboration scientifique**. Le courier électronique a permis, en plus du courier individuel, des envois collectifs dits "news", regroupés par thèmes tels "sci.math", "sci.math.symbolic". Des résultats ou questions y sont soumis à discussions. Ce mode de fonctionnement entremêle plusieurs sujets impliquant un suivi temporel. Au lieu d'un multiplexage temporel, une sorte de téléconférence pourrait spatialement regrouper un débat dans un emplacement donné d'un hypermédia. Chacun disposerait alors de la totalité de l'information pertinente pour le sujet, et éviterait de saturer ainsi l'ensemble des abonnés. Des synthèses issues de ces débats pourraient alors être consultées ultérieurement.

Par ailleurs, le mathématicien aimerait pouvoir relier une preuve à une démonstration correspondante. Mais la notion même de démonstration, comme présentation d'une preuve, est soumise à révision par l'utilisation d'un outil actif comme l'ordinateur. Une démonstration serait alors un objet non-linéaire dont les étapes d'argumentation/présentation peuvent être déployées au gré de leur destinataire. C'est l'approche utilisée par Marc Kaltenbach pour le **logiciel d'animation de démonstrations DYNABOARD**, à usage pédagogique. Selon [Kaltenbach 87], DYNABOARD utilise :

> une représentation spatiale du raisonnement capable d'évoluer en fonction des choix et préférences de l'étudiant. L'accent est mis sur le rôle actif de l'étudiant dans l'évolution de la représentation en accord avec l'état de compréhension qu'il a atteint.

Les facilités hypermédia de DYNABOARD sont utilisées pour explorer une preuve à partir de l'énoncé. Ce dernier est raffiné à la demande en un squelette de plus en plus précis de la preuve totale. Son interface graphique, alliée à une manipulation directe, mettent en valeur trois points importants en pédagogie :

- la structure **non linéaire** des arguments mathématiques ;
- les divers **niveaux d'abstraction** sous lesquels une preuve peut être envisagée ;
- le **rôle actif de l'étudiant** dans la personnalisation de l'information visualisée.

Ces points montrent la proximité des motivations de DYNABOARD et de l'approche hypermédia classique. En revanche, DYNABOARD se démarque par sa présentation plus sophistiquée de l'information : en ce sens, il constitue un pas vers les métamédia.

Pour **gérer l'évolution** des supports visuels, l'ordinateur doit disposer d'une connaissance du contenu de l'information, des mécanismes de présentation, et du processus global amenant à cette visualisation. L'aspect "métamédia" interviendrait pleinement si DYNABOARD pouvait produire et valider ces preuves en fonctions de connaissances générales. En effet, l'aspect formel de certains liens, tel ceux de l'application d'un théorème ou la vérification d'une preuve échappent à la "philosophie" même des hypermédia. Il est hors de leur propos de connaître le pourquoi, le comment, et la valeur du résultat de l'application d"un théorème sur une formule. Pour un hypertexte, il y a les nœuds "formule-précédente", "théorème" et un lien vers "formule-suivante". Leur indépendance fait que "formule-suivante" est construite comme le serait "n-importe-quoi".

La limitation énoncée apparaît car la gestion et la réalisation de cette animation sont produites manuellement par le concepteur, et non à partir de la preuve et de connaissances mathématiques. La transcription de ce cas à une autre démonstration doit alors se faire par modification du logiciel. Des techniques restent donc à développer pour les mathématiques, afin de la rendre indépendante d'un énoncé donné, voire d'un domaine particulier.

---

[11] 200 000 théorèmes publiés chaque année selon l'estimation de 1980 citée dans [Davis 85].

Il serait regrettable de **se priver des potentialités "dynamiques" et calculatoires de l'ordinateur** en se contentant d'un support passif, certes plus flexible et plus élaboré qu'un livre. Toutefois, les techniques informatiques pour y parvenir sont sophistiquées et les réalisations restent à venir. Ainsi, DYNABOARD était un prototype dont aucune version commerciale n'est disponible.

1.3.4. Les systèmes de brouillon, de gestion d'idées et de visualisation

Le thème de la section précédente visait à assouplir l'exploitation des informations disponibles. Celui de cette section cherche essentiellement à favoriser la créativité. Elle met en avant la pensée visuelle et le principe du brouillon.

Le **principe du brouillon** consiste à :

- disposer d'un support d'expression permettant de suivre le fil de la pensée ;
- permettre de travailler des informations souvent faiblement structurées ;
- fournir des moyens d'organisation et de synthèse des informations.

Les deux premiers points perdent grandement leur intérêt sans les techniques actuelles comme saisie vocale et digitaliseurs, voire compréhension de la parole et de l'écriture. L'idée serait alors de pouvoir travailler ces informations sur machine, sous leur forme écrite. Les systèmes interactifs actuels ont fait en quelques années des progrès considérables en ce sens. Toutefois, ils n'offrent pas encore toute la souplesse nécessaire. Il devraient souvent être plus réceptifs aux intentions de l'usager, et améliorer la qualité et le traitement de leurs observables (par ex. inclure la dimension temporelle).

La **pensée visuelle** est, en revanche, déjà mise en valeur par les techniques de visualisation existantes. Ainsi, la visualisation des programmes et de leur exécution, facilite l'enseignement de la programmation [Liem 89] et influence l'évolution des environnements de programmation [Ambler 89]. Les mêmes idées sont utilisées pour offrir plusieurs représentations lors de raisonnements mathématiques simples [McArthur 88]. Il y a là une branche d'étude des moyens, des méthodes et des modes de synthèse et de présentation de l'information qu'un esprit audacieux serait tenté de qualifier de "*visualétique*". Le problème principal des outils de visualisation reste actuellement un temps de développement important pour leur exploitation.

La simulation visuelle est un support puissant couramment utilisé pour l'utilisation de mathématiques appliquées à d'autres domaines scientifiques. Ses potentialités permettent de matérialiser l'espace et le temps. Les applications géométriques sont directement concernées, mais un peu d'imagination permet de tirer parti d'un tel support pour représenter des entités plus abstraites et favoriser le développement de l'intuition. Plusieurs **modes de construction** sont envisageables, allant du film vidéo à la génération d'image par ordinateur. De même, les **modes de visualisation** vont de la vue statique à une présentation dynamique commandée par l'action de l'usager.

Le cas du "retournement de la sphère" constitue une illustration exemplaire du pouvoir intuitif et persuasif de l'animation : l'animation vidéo a levé le scepticisme qu'une preuve trop complexe laissait planer. Plus généralement, les propriétés topologiques des surfaces s'expriment à l'aide de déformations successives permettant une transition continue de l'état initial à l'état final. Ainsi tel ouvrage de sensibilisation présente une séquence d'images illustrant le retournement d'une chambre à air, ou le fait que le gilet n'est nullement "à l'intérieur" de la veste. Cette présentation par étapes est généralement suffisante pour l'appréhension du phénomène.

Lorsque les objets impliqués sont de dimension supérieure à deux, la complexité de la visualisation devient importante. La coupe 2-D constitue au mieux un point de vue secondaire à la bonne compréhension de l'ensemble. Les techniques utilisées en CAO sont à la base de toute investigation dans ce domaine. A noter que la bonne appréhension de l'espace qu'elles autorisent permet de matérialiser la quatrième dimension. Ainsi, un hypercube de dimension 4 acquiert une concrétisation intuitive par des rotations et autres manipulations visualisées par les projections 3-D.

Les systèmes de visualisation ne permettent pas seulement la compréhension de phénomènes physiques : ils servent à mettre en évidence des relations. DYNABOARD s'y intéressait pour les preuves, Cabri-géomètre (Cahier de brouillon informatique) permet de visualiser des relations en géométrie plane. Avec Cabri-géomètre [Baulac 89], l'élève dispose d'un micro-monde de constructions géométriques. Par exemple, en déplacant par manipulation directe le sommet d'un triangle, l'élève voit les médiatrices se déplacer mais rester concourantes. Ces manipulations non dirigées permettent à l'étudiant d'évoluer librement dans l'espace aménagé par le concepteur.

1.3.5. Les systèmes de représentation et de production de démonstrations

Cette section concerne la production de textes qualifiés de démonstration, et s'appliquerait aussi à d'autres sciences et techniques. Elle explore les solutions proposées pour aller au delà d'un traitement de texte "ordinaire". Elle se rapproche à cet effet de la problématique du Langage Naturel, dont une synthèse est accessible dans [Coulon 86].

Il faut d'abord constater qu'il n'existe rien de très avancé, c'est-à-dire que toutes les solutions proposées sont intéressantes. Nous avons recensé quatre approches effectives :

- manipuler des démonstrations pour l'enseignement ;
- proposer une grammaire pour les démonstrations ;
- produire un texte qui exploite des outils de calcul ;
- générer des démonstrations ;

DYNABOARD est un exemple de logiciel destiné à **manipuler des démonstrations pour l'enseignement**. Dans une optique différente, ARRIA [Bruillard 91] permet à l'étudiant d'assembler des fragments de démonstrations en utilisant des connecteurs logiques. Dans ces deux cas, les liens possibles entre les fragments doivent être prédéfinis et leur contenu est fixé a priori.

Une autre façon de modéliser les démonstrations consiste à **définir un langage avec sa grammaire**. Nous connaissons une tentative pour le langage écrit et une pour le langage oral. Ainsi, pour recueillir et représenter les discours mathématiques, P. Dybjer a proposé une grammaire permettant de représenter les déclarations et les enchaînements déductifs sans se préoccuper de leur cohérence mathématique [Dybjer 82]. Pour le langage écrit, André Arnold avait réalisé un système pour manipuler "un langage de formalisation de démonstrations mathématiques naturelles" [Arnold 68].

La solution générale, dirigée par les techniques disponibles consiste à **utiliser des outils de calcul** (ou de raisonnement) pour produire un texte. C'est notamment l'approche annonciatrice adoptée par CaminoReal dans l'environnement Cedar développé à Xerox PARC [Arnon 88]. Il permet de décrire un texte avec des calculs symboliques à effectuer. Le document ainsi constitué peut alors être mathématiquement évalué, en garantissant ainsi sa correction. Dans le même esprit, l'expression d'un problème de simulation aérodynamique avec Mathematica était suffisamment claire pour servir d'auto-documentation et de validation de sa correction vis à vis du modèle disponible dans la documentation de l'avion AIRBUS A300.

La dernière approche s'inspire des techniques de génération du Langage Naturel pour **générer des démonstrations**. Le problème abordé est celui de la génération d'un texte de démonstration à partir du problème, dans le cadre pédagogique de la résolution d'équations [Zukerman 86]. Un module de raisonnement fournit une solution qui est ensuite adaptée à l'étudiant et au cours. Un raisonnement pédagogique y insère alors des "expressions méta-techniques" (telles "toutefois", "en remarquant que", "ce sujet est très important", etc.) divisées en trois thèmes :

- organiser l'argumentation mathématique ;
- faciliter l'enseignement mathématique ;
- soutenir le moral.

Le système obtient ainsi une représentation codée qu'un générateur de phrases utilise pour produire un texte de démonstration. Ce système permet de cerner les problèmes techniques soulevés par la génération de phrases, et de valider la faisabilité de la production d'une démonstration en mathématiques. Ce travail repose sur une analyse fine des connecteurs mathématiques. Les paramètres de codage ainsi développés contribueraient à répondre aux besoins d'un langage intermédiaire pour les mathématiques [Strichartz 89][12]. Bien qu'un tel langage se contente de critères essentiellement syntaxiques, il contribuerait au développement de **recherches sur le traitement du Langage Mathématique**, rejoignant en cela la multiplicité des approches du Langage Naturel.

Finalement, ce besoin de production de démonstrations textuelles ne doit pas masquer les limites du Langage Naturel : indépendamment de la complexité de son traitement informatique, sa faculté de

[12] Ce langage intermédiaire Intermath (cf. langage pivot de la traduction automatique) constituerait un langage chargé d'assurer l'indépendance d'un texte mathématique vis à vis d'une langue donnée. Ces objectifs sont différents des nôtres, mais des recherches communes resteraient à entreprendre.

traduire des intentions très diversifiées rend la compréhension des messages souvent délicate. Par exemple la phrase :

Effectuer la simplification de la formule avec la substitution de x en $\rho \cos \theta$.

peut donner lieu à trois interprétations syntaxiques selon qu'elle concerne :

- la formule :
  afin de préciser la formule à simplifier (ambiguïté sur la formule), par exemple ici une formule réifiant la substitution : $[x \longrightarrow \rho \cos \theta]\ \varphi(x)$;
- la règle de simplification :
  afin de préciser la substitution à effectuer pour appliquer une règle supposée connue ;
- l'opération à effectuer :
  ici, effectuer la substitution directement sur la formule considérée.

Le système doit donc lever les ambiguïtés possibles et reconstituer le sens complet du message à partir de sa connaissance de l'état du système Homme-Machine (aspects pragmatiques). Il est souvent plus simple, pour l'homme et pour la machine, de désigner directement l'objet concerné par le message, et de spécifier uniquement les paramètres indispensables pour son utilisation. Nous préférons ainsi fréquemment une saisie dans un **langage de dialogue spécialisé pour un système informatique.**

1.3.6. Les systèmes de modélisation de problèmes spécialisés

L'état des lieux actuel permet le développement de systèmes traitant des problèmes ponctuels, et intégrant des capacités de calcul et de raisonnement avec des capacités de formulation et de présentation. C'est notamment le cas pour les tuteurs intelligents, mais de tels systèmes se développement aussi en biologie, pour la description de scènes d'infographie, pour effectuer des maillages, etc. Nous présentons brièvement un système de Mécanique, et un autre en Automatique.

MECHO est un système destiné à traiter des énoncés de Mécanique (cordes et poulies) exprimés en anglais. Il utilise des connaissances de "bon sens" spécialisées dans le traitement de ces énoncés pour produire une représentation intermédaire décrivant le problème en PROLOG. MECHO extrait alors de cette représentation des équations à résoudre qu'il soumet à PRESS : tous deux sont notamment décrits dans [Bundy 83]. MECHO était destiné à montrer la faisabilité de la formation d'un Modèle mathématique à partir de la spécification informelle d'un problème.

PANDORE est un Système Expert pour l'optimisation des systèmes dynamiques [Quadrat 89] :

> Le but du système *Pandore* est d'automatiser l'ensemble des tâches à accomplir pour faire une étude d'optimisation de systèmes dynamiques, utilisant des techniques de calcul formel, d'inférence, de manipulations symboliques et d'analyse numérique. Le système est écrit en *Macsyma*, *Lisp* et *Prolog* et tourne sur une machine Lisp Symbolics. Deux langages : MACROFORT et MACROTEX ont été développés permettant la génération de programmes *Fortran* et LATEX à partir de *Macsyma*.
>
> L'utilisateur spécifie son problème sous forme symbolique au moyen d'un éditeur spécialisé. Un langage de commandes permet d'interroger le système et de modifier la base de données du problème.
>
> Lorsque le problème est bien posé, une méthode de résolution est choisie par le système. Les critères de choix des méthodes de résolution sont codés sous forme de clauses *Prolog*. Une étude théorique de l'existence et de l'unicité de solutions d'équations aux dérivées partielles associées à ces problèmes est menée dans certains cas. *Pandore* génère ensuite les programmes *Fortran* pour l'étude numérique, puis un rapport LATEX comprenant le modèle, l'explication de la méthode de résolution choisie, l'étude numérique, les graphes des résultats.

PANDORE est donc essentiellement un système de traitement d'un problème et de génération du compte-rendu de sa résolution. Le rapport présenté par PANDORE en conclusion de son étude est remarquable. Il est donc possible de créer des systèmes spécialisés intégrant des fonctionnalités très variées. Toutefois, un tel système est conçu pour automatiser un traitement plutôt que pour assister une tâche : les choix de conception sont alors différents de ceux que nous cherchons. Nous retiendrons néanmoins que l'intégration de fonctionnalités est un problème complexe.

1.4. LIMITES DE L'EXISTANT

Ce panorama des systèmes informatiques utilisés pour les mathématiques nous a permis de **classifier des sous-systèmes** utiles dans le cadre d'un Atelier Mathématique Intégré (AMI). Il offre des repères au lecteur uniquement intéressé par les Mathématiques Assistées par Ordinateur (MAO). La bibliographie est nullement exhaustive : elle est, par exemple, loin d'égaler la bibliographie recueillie par Roger Cuppens pour les textes qui présentent un intérêt didactique.

Ce panorama montre indirectement l'influence des techniques informatiques disponibles sur les systèmes mathématiques. Par contre, les mathématiques constituent un champ d'expérimentation et d'application de ces techniques plutôt qu'une source d'innovations. Les systèmes les plus prometteurs informatiquement sont alors souvent des systèmes aux capacités mathématiques modestes, auxquels nous avons fait une large part.

Cette hétérogénéité des produits nous montre qu'il n'est pas encore possible de parler de MAO sauf sur des problèmes mathématiques très particuliers. Il reste à **surmonter une complexité** quantitative (extension des tâches et des domaines d'application) et qualitative (amélioration des techniques et des fonctionnalités) en vue d'une intégration. Les MAO pourraient alors être définis comme les recherches et les systèmes visant à assister l'activité du mathématicien en vue d'une intégration éventuelle dans un AMI. Les recherches ne font que commencer...

Les outils pour raisonner et calculer offrent des possibilités intéressantes mais, trop souvent, ils n'offrent pas une réponse adaptée aux problèmes des mathématiciens. Les systèmes de calcul formel sont finalisés vers l'automatisation d'un résultat et s'avèrent déficients dans leurs explications ou leur guidage. Par ailleurs, leurs critères de conception pose des problèmes de validité : il n'ont pas besoin de preuve, ils sont "pré-validés" avec leur méthode de calcul. Cette notion de preuve étant absente, ils n'ont pas besoin de connaissances susceptibles de justifier leurs actes. Par exemple, [Fateman 85] tente onze preuves de $\sin^2 x+\cos^2 x = 1$, alors que Macsyma ne dispose pas directement de ce résultat.

Les systèmes de calcul formels sont donc très utiles pour les cas où une méthode prédéfinie permet d'obtenir un résultat. Dans la plupart des cas, le mathématicien ayant besoin de raisonner et d'exhiber une preuve, ils servent au mieux d'outil de calcul auxiliaire. Les systèmes de raisonnement apportent alors une réponse plus sophistiquée. Nous considérons cet acquis comme majeur, mais là aussi, les systèmes buttent sur la complexité des besoins dégagés pour le raisonnement.

L'article de W. W. Bledsoe [Bledsoe 86] recense les thèmes de recherche nécessaires pour obtenir des systèmes plus étendus. Il montre qu'un système expert en raisonnement doit **simuler le mathématicien**. Par exemple, la démonstration du théorème qui indique que la limite d'une somme de deux fonctions est la somme de leurs limites, ne peut se contenter de la donnée d'une liste de clauses de Horn à utiliser. Il met ainsi en avant le rôle central d'une base de connaissances mathématique structurée. Il reste néanmoins à en déterminer le contenu et l'organisation...

Il nous semble qu'une réponse réaliste aux problèmes rencontrés doit prendre en compte prioritairement les moyens d'assistance au mathématicien. Les outils de formulation et de présentation offrent une diversité très enrichissante en ce sens. Ainsi, l'approche d'André Arnold [Arnold 68] était particulièrement pertinente pour notre objectif d'assistance à la construction de preuves et démonstrations. Ce dernier utilisait une approche pratique cherchant à vérifier une preuve décrite en logique dite "déduction naturelle". Il avait alors introduit un langage de présentation structuré très lisible, utilisant des expressions en Langage de Phrases.

Nous nous sommes donc interrogés sur l'absence d'utilisation actuelle d'un tel système par les mathématiciens. Hormis les moyens humains et matériels nécessaires, nous pensons que les **problèmes principaux** concernent l'ergonomie du système et les fonctionnalités d'assistance au mathématicien. Nous avons donc choisi une approche prospective, similaire de celle recherchée par Jacques Calmet pour les systèmes de calcul formel [Calmet 87]. Nos objectifs sont néanmoins complémentaires car notre approche est essentiellement cognitive et centrée sur le mathématicien.

L'un des acquis majeurs de l'analyse de l'activité mathématique des parties I et II est que les connaissances usuellement modélisées dans les systèmes restent superficielles et se cantonnent à des situations stéréotypées. Les solutions proposées se limitent alors à quelques domaines restreints, sans s'intéresser directement aux problèmes généraux posés par l'usage des formules, du langage et la modélisation d'un domaine.

Dans cette perspective généraliste, il paraît surprenant que le traitement du langage et des connaissances mathématiques ne fassent pas l'objet d'un travail de recherche spécifique destiné à **factoriser ce qu'il y a de commun** dans l'activité mathématique. Bien que les problèmes se posent pour tout système informatique, cette factorisation potentielle bute sur les caractéristiques suivantes :

❶ la plupart des systèmes cherchent une commercialisation à court terme ;
❷ les techniques d'outils génériques sont ardues et encore peu répandues ;
❸ la conception des systèmes existants serait en majeure partie à revoir ;
❹ l'apport de cette factorisation n'est guère justifiable pour une fonctionnalité isolée.

La première solution informatique envisageable est la récupération des modules existants et leur association par des passerelles logicielles. Cela satisfait les points ❶ et ❸, mais ne résoud guère notre problème.

Une autre solution est la juxtaposition de fonctionnalités au sein d'un même système, accompagnée d'une conception et d'une écriture homogène de l'ensemble en utilisant les techniques disponibles. C'est la solution adoptée par Mathematica, qui satisfait en partie les points ❷ à ❹, mais pour lequel le court terme ❶ est rédhibitoire à une ambition plus générale. Le point ❶ est ainsi incompatible avec notre objectif, ce qui résoud les points ❷ et ❸ dans la foulée.

L'**intégration des fonctionnalités** est une solution envisageable dès que l'on s'accorde le temps d'effectuer les recherches nécessaires. Cela est le cas lorsque l'on songe que les mathématiques ne sont toujours pas vraiment assistées : l'activité du mathématicien est loin de pouvoir être transférée d'une activité papier-crayon vers une activité en majorité assistable informatiquement. L'ampleur des besoins en mathématiques requiert une organisation en projets parallèlement au développement de systèmes.

L'intérêt d'un AMI futur se justifie notamment par l'intégration d'outils autour du cycle de production mathématique, depuis les problèmes jusqu'aux démonstrations produisant leurs résultats. Cette informatisation des données manipulées est, en effet, remise en cause si le mathématicien doit effectuer une quantité importante de codage pour le compte de la machine. Tout maillon manquant dans le cycle de production serait ainsi contraire à l'intérêt d'un support informatique prioritaire.

Notre contribution lors de cette partie III restera modeste relativement à ces objectifs. Nous nous sommes focalisés sur les problèmes soulevés par les manipulations de formules. Nous avons cherché des mécanismes généraux en vue de réaliser des manipulations aisées pour le mathématicien, ce qui est une condition critique pour la faisabilité d'une assistance au mathématicien. Les techniques informatiques y sont abordées comme un aspect important de la conception d'un système, sachant que seule la faisabilité nous concerne directement.

2. LES MANIPULATIONS DE FORMULES

2.1. SOFTMATH ET LES MANIPULATIONS DE FORMULES

2.1.1. Nos objectifs pour les manipulation de formules

Nous avons effectué précédemment un panorama des Mathématiques Assistées par Ordinateur. Mais proposer une **assistance au mathématicien** est aussi vaste et aussi ambitieux que de proposer une assistance à la résolution de problème (l'un des défis de l'IA, qui s'est progressivement spécialisé à des situations précises). Afin de déboucher sur une réalisation concrète, nous étudions maintenant la manipulation de formules mathématiques.

Un environnement de manipulation de formules, baptisé SOFTMATH, constituera le noyau d'un Atelier Mathématique Intégré (figure 3.7). Rappelons néanmoins que les spécifications externes présentées dans ce chapitre ne sont pas implémentées : il s'agit donc de propositions afin de réaliser une manipulation directe des formules, qu'un Assistant mathématicien intelligent (Ami) va aussi être en mesure d'effectuer.

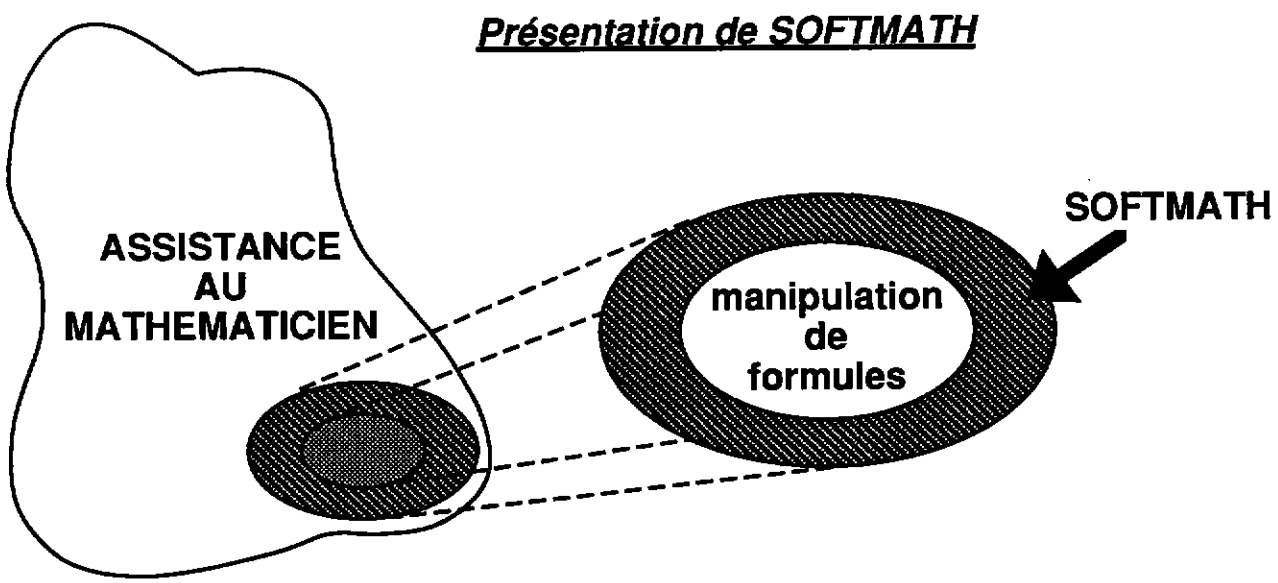


figure 3.7

La manipulation de formules est une pierre angulaire de l'activité mathématique. Elle est aux mathématiques ce que la manipulation de programmes est à la programmation : elle ponctue le cycle de conception dans sa diversité.

Le "cœur" purement mathématique des manipulations de formules est constitué de calcul symbolique et de raisonnement mathématique. Nombre de produits et de méthodes ont été développés en mathématiques dans le cadre de cette approche "calculatoire". Nous nous intéressons à la manipulation de formules selon une **approche "Génie Logiciel" d'environnement de manipulation.**

L'activité mathématique utilise fréquement des formules et la plupart des tâches ne peuvent se satisfaire de la seule apparence graphique des formules. Si l'apparence est importante pour son appréhension du problème, le mathématicien vise le contenu mathématique d'une formule. Pourtant, malgré leur sophistication actuelle, peu de logiciels traitent correctement les deux aspects : la forme, et le contenu. Les meilleurs demandent une forte intervention de l'homme quant à la construction de la forme, ou manipulent le contenu sans se préoccuper de la façon de procéder d'un mathématicien ni de la forme qui en résulterait.

Nous voulons éviter les deux principaux pièges des Mathématiques Assistées par Ordinateur :

- bipolariser les activités mathématiques en systèmes de traitement de texte pour une activité de secrétariat, et en systèmes d'exploitation de résultats de calcul ;
- laisser le mathématicien effectuer la majeure partie de ses productions écrites en dehors d'un système informatique.

La **bipolarisation** est une conséquence des besoins immédiats et des techniques informatiques passées. Les mathématiciens ne se reconnaissent guère dans les activités proposées, et se désintéressent de l'informatique. Une reconsidération du rôle de l'informatique permettrait d'éviter ce piège, à condition de proposer d'autres perspectives. Aussi, ce chapitre étudie les possibilités d'enrichir les manipulations de formules et d'améliorer leur réutilisation.

L'autre piège est plus redoutable car plus général : l'**éloignement du mathématicien** vis à vis de l'informatique l'oblige à introduire et formuler à chaque fois son problème s'il veut se servir d'un assistant informatique. Nombre de réutilisations potentielles sont ainsi inexploitées faute de disposer de leur contexte d'exploitation. Plus grave encore, le mathématicien est amené à remplacer mécaniquement le "chaînon manquant".

Nous voulons donc un système informatique qui puisse servir le mathématicien en permanence, et nous avons proposé d'intégrer les manipulations textuelles et mathématiques au sein d'un processus de construction de preuves et démonstrations. Pour que cela soit réalisable, nous voulons :

Calquer les manipulations de formules sur l'activité mathématique naturelle.

Bien que les manipulations de formules proposées pourraient servir d'interface à un système de calcul formel de prochaine génération, elles n'en sont pas moins indissociables des connaissances langagières identifiées en première partie. Notre objectif : "calquer les manipulations de formules sur l'activité mathématique naturelle" est en effet bien différent de l'objectif : "exploiter une bibiothèque de résolution de problèmes combinatoires". Notre approche cognitive met l'accent, non pas sur les algorithmes connus, mais sur les connaissances mathématiques et la façon dont un système peut les exploiter pour aller de pair avec les intentions du mathématicien.

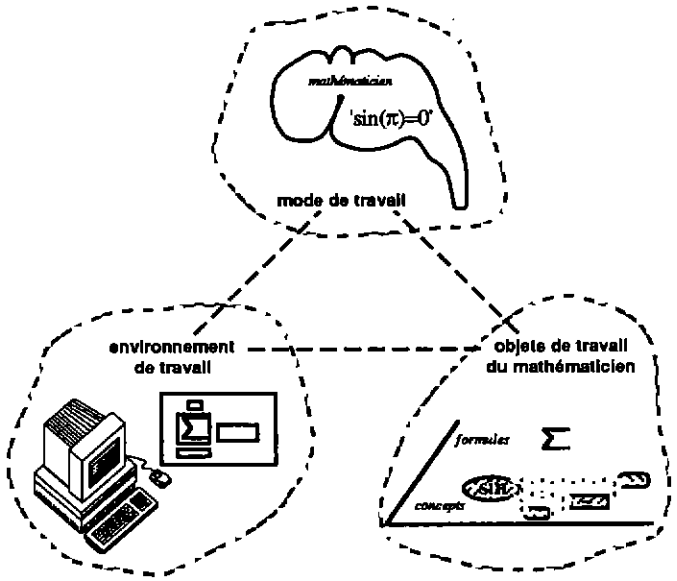
Pour cela, le meilleur modèle du mathématicien est un modèle précis des dépendances entre objets manipulés et de la façon dont un mathématicien les exploite. Il s'agit donc d'exploiter la cohérence et l'usage mathématiques : cela cadre la zone d'intervention du système aux tâches élémentaires, la gestion des décisions de niveau supérieur revenant au mathématicien.

Ce découpage des tâches ne rend pas le système d'assistance inapte à toute activité mathématique. Au contraire, il partitionne les tâches en fonction de ce que le système sait faire, ce qui permet au mathématicien de bénéficier de procédés de calcul symbolique. Il est même possible que le système prenne en charge le déroulement de calculs lorsque les composantes heuristiques lui sont accessibles. Mais cela ne concerne que des micro-domaines, et le système doit supporter les tâches élémentaires d'un domaine plus vaste. Son ambition est de s'adapter notamment aux domaines trop conceptuels pour que des heuristiques autonomes soient efficaces.

2.1.2. Notre approche pour SOFTMATH

Nous décrivons dans ce chapitre des manipulations de formules à réaliser avec SOFTMATH, en rappelant qu'elles ne sont pas implantées dans le cadre de notre travail. Les objectifs sont de fournir au mathématicien un environnement de saisie, d'analyse et de transformation de formules aux niveaux mathématiques et textuels, utilisant des connaissances adaptables à des domaines variés.

Nos principes de conception sont d'assister le mathématicien en s'adaptant à son environnement "naturel" (selon le rappel de la figure 2.9). Nous voulons pour cela faciliter les intentions, les décisions et les actions du mathématicien.



Assister le mathématicien signifie notamment travailler simultanément sur la forme et sur le fond, et fournir au mathématicien les moyens d'expression auxquel il est familier. Afin que les décisions du mathématiciens soient aisées, nous avons cherché à ce que le guidage s'exprime sur les objets, avec les termes et d'une façon comparable à celle qu'utilise le mathématicien. Ce dernier sait alors exprimer ce qu'il souhaite, tandis que le système est capable de lui expliquer ce qu'il fait, pourquoi et comment il le fait.

Vu ces objectifs, nos propositions privilégient donc **l'ergonomie cognitive** en dépit des difficultés informatiques afin d'obtenir une indispensable **qualité d'interaction.**

L'optique d'assistance au mathématicien, par opposition aux environnements de calcul "classiques", permet de bénéficier de plus de souplesse dans les manipulations. Ainsi, des connaissances pragmatiques aident considérablement l'usager dans sa tâche, la prise en compte des aspects cognitifs favorise l'interaction, l'organisation basée sur des connaissances favorise le développement, l'extension ou la restructuration d'un système mathématique de taille réaliste.

Enfin, le mécanisme d'analyse n'a pas besoin d'être effectif à 100%. Paradoxalement, cette perte d'optimalité se traduit par une souplesse permettant un gain d'expressivité. Elle permet d'exploiter la plage d'analyse située entre la quantité d'information à fournir obligatoirement pour aboutir, et celle suffisante pour que le système puisse fréquemment compléter et aboutir quand même.

Or, l'usager n'a pas à réfléchir sur le fonctionnement effectif de l'analyseur. Il ne doit donc pas chercher à estimer si le système dispose d'informations suffisantes pour conclure. Au contraire, il doit utiliser l'intuition acquise par la pratique du système pour tenter des "ellipses d'information". Cela se fait d'autant plus naturellement si le système ne provoque pas d'échec brutal et demande des précisions à l'usager. Ainsi, un système "spécialisé" et déductif peut réduire la quantité d'information à produire en saisie, par rapport à celle qui est effectivement désirée pour la Représentation Interne.

Les techniques que nous proposons d'utiliser pour SOFTMATH sont issues de l'état de la recherche informatique. Elles ont été développées initialement pour les **environnements de manipulation de langages** comme MENTOR, ADELE, CONCERTO [Donzeau-Gouge 83, Coutaz 84, Conchon 86] et pour la conception d'interfaces interactives [Conchon 85, Coutaz 86 et 90, Barthet 88]. Elles sont

extensibles aux caractéristiques textuelles et graphiques spécifiques aux manipulations de formules du langage mathématique. Par exemple, l'éditeur interactif EDIMATH [Quint 83] présente et adapte ces principes pour la représentation, l'affichage et la manipulation textuelle de formules.

Finalement, les deux types de critères retenus dans les choix de conception sont :

① l'aspect naturel et le caractère général de SOFTMATH :
- accepter en entrée/sortie un formalisme proche du langage mathématique usuel ;
- fournir un environnement de manipulation convivial ;
- prendre en compte le sens, la forme et le style ;
- supporter la famille des formalismes informatiques destinés aux mathématiques.

② modélisation et représentation déclarative du langage et des connaissances :
- aspects mathématiques et textuels ;
- connaissances pragmatiques explicitées (complétions par défaut, etc.) ;
- extension incrémentale des formalismes et connaissances en cours de session ;
- prise en compte de la lisibilité, cohérence de notations, génération de variables...

SOFTMATH fournit au mathématicien un **environnement** et les **connaissances** (para-) mathématiques nécessaires pour saisir, analyser et transformer des formules. Dans un premier temps, SOFTMATH se contente de vérifier la cohérence des étapes de manipulation. Cela est suffisant pour envisager l'intégration des techniques de calcul formel existantes.

Les **fonctionnalités** attendues de SOFTMATH sont :
- gérer des formules et vérifier leur cohérence mathématique ;
- procurer des manipulations riches par l'utilisation des connaissances et de la structure mathématique des formules ;
- expliquer ce qu'il fait, pourquoi et comment il le fait en des termes équivalents à ceux qu'emploierait un mathématicien ;
- acquérir des connaissances de manipulation et de raisonnement mathématique par l'enseignement ou par l'observation de mathématiciens ;

SOFTMATH est donc un environnement de manipulation de formules prévu pour s'étendre aux manipulations de preuves et démonstrations dans le cadre d'un AMI (Atelier Mathématique Intégré) décrit en première partie. La gestion des représentations et les manipulations de formules sont propres à SOFTMATH, tandis que l'exploitation et l'organisation des connaissances s'appliquent tout autant pour un AMI.

Même si nous n'en cachons pas les difficultés techniques, SOFTMATH est un environnement dont l'originalité des manipulations de formules préfigure des constructions naturelles de preuves et démonstrations. Il s'intègre ainsi dans les objectifs d'étude expérimentale de la première partie, et précise pour les manipulations de formules comment donner au mathématicien le sentiment d'engagement direct dans son activité de résolution de problème.

Les enjeux résultants de la conception de SOFTMATH sont :
- étudier les mathématiques, leur pratique et expliciter leurs connaissances ;
- obtenir un support expérimental pour des techniques interaction et Intelligence Artificielle;
- disposer d'un noyau extensible vers un ensemble d'applications traditionnellement dévolues aux mathématiques.

2.1.3. Un modèle de formules basé sur trois structures couplées

Le mathématicien manipule couramment **trois structures** lors de son activité. La *structure apparente* est la forme bidimensionnelle d'une formule. La *structure textuelle* correspond à l'organisation syntaxique et la *structure mathématique* décrit l'information permettant de déterminer le sens mathématique. Ces deux dernières sont qualifiées de structures *sous-jacentes* de la formule.

La figure 3.8 développe l'exemple du sigma. La structure apparente, ou forme bidimensionnelle, matérialise conjointement des éléments des autres structures. Les zones de la structure textuelle sont représentées par des encadrements. Par ailleurs, les caractères "k" de la variable décrite dans la structure mathématique peuvent être mis en valeur.

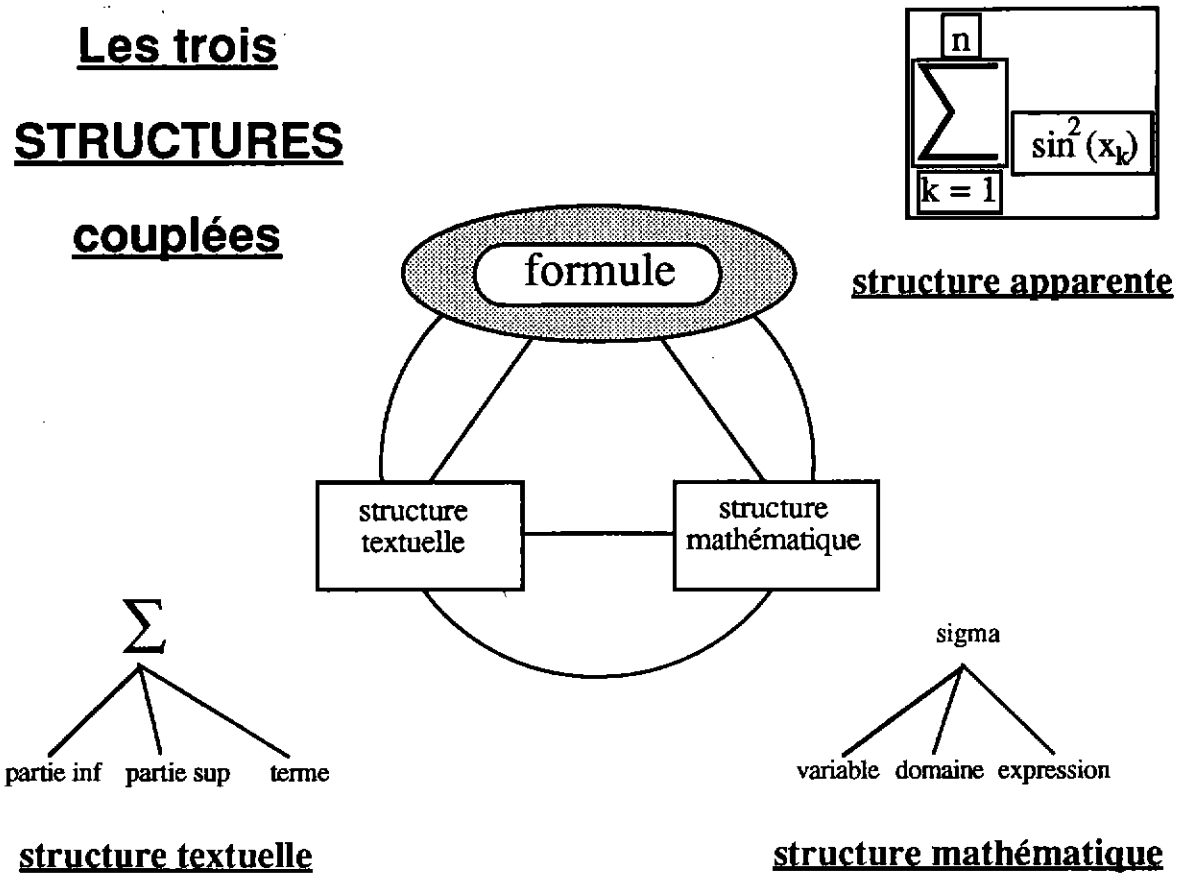


figure 3.8

La structure textuelle décrit les arguments de l'opérateur textuel sigma : une partie inférieure, une partie supérieure (éventuellement vide), et le terme principal. Elle correspond au style de présentation syntaxique choisi, et est utilisée pour générer la forme bidimensionnelle.

La structure mathématique décrit les arguments de l'opérateur mathématique. Par exemple l'argument domaine est ici un intervalle décrit par ses bornes dans l'ensemble des entiers naturels, lui-même choisi par défaut.

Le **couplage des trois structures** se fait à partir de la structure mathématique, qui est la structure de travail des opérations mathématiques, pour aboutir à la structure apparente, qui est le support de travail du mathématicien. Chaque structure mémorise un certain type d'information, même si elles restent corrélées.

Toutes les informations de la structure textuelle sont utilisées pour générer la structure apparente. Par contre seuls certains éléments da la structure mathématique interviennent dans l'information codée dans la structure textuelle. Ainsi dans l'exemple du sigma, le domaine "D" n'intervient pas explicitement dans la présentation de la structure textuelle, comme ce serait le cas avec une partie inférieure telle que "k∈D".

Toutefois, les informations de la structure mathématique influencent sensiblement le choix de la structure textuelle. Par exemple, dans les éditeurs de formules "classiques", "2(ax+b)" est envisagé comme un bloc textuel contenant la chaîne de caractères "2(ax+b)". Avec les trois structures, il est hiérarchiquement structuré à partir de sa structure mathématique arborescente. Le parenthésage est ainsi considéré comme un étiquetage syntaxique choisi pour l'expression mathématique "ax+b".

2.1.4. Le cycle de manipulation de formules

Les principaux utilitaires actuels de manipulations de formules se limitent soit à la structure textuelle, soit à une structure mathématique en délaissant les manipulations visuelles. L'introduction des trois

structures couplées, et les mécanismes qu'elles permettent d'exprimer, représente un progrès certain dans l'accomplissement de la tâche du mathématicien.

Tout d'abord, l'introduction de manipulations textuelles structurées forme le premier pas vers des manipulations cohérentes. De nombreux éditeurs n'utilisent pas encore cette technique, car elle requiert une complexité informatique très importante. De plus, elle n'est viable que dans des environnement interactifs sophistiqués.

Enfin, l'introduction d'une représentation structurée se traduit par une rigidité de manipulation à maitriser. Faut-il rappeler que les premiers éditeurs structurés de formules imposaient une saisie hiérarchique ou obligeaient d'effacer partiellement une saisie pour modifier une erreur typographique ! L'introduction de manipulations de la structure mathématique forme le deuxième pas qualitatif vers les manipulations mathématiques.

L'**introduction de trois structures** est indispensable pour l'usage mathématique :

- Supprimer la structure apparente et n'avoir que des arbres relève de l'hérésie...
- La structure textuelle est nécessaire bien que certains systèmes de calcul s'en dispensent. Ils se privent ainsi des aspects naturels du langage mathématique que sont sa lisibilité, sa concision et son expressivité. Les manipulations de formules sont souvent laborieuses car le mathématicien perd ses repères naturels.
- La structure mathématique est omise dans les traitements de texte. Ils se privent ainsi de toute source d'information de cohérence sémantique. Quand la structure textuelle sert à décrire des opérations mathématiques, elle est analysée pour obtenir la structure mathématique. Pourquoi alors le faire implicitement et ne pas gérer cette correspondance de façon incrémentale ? Les manipulations textuelles y gagnent grandement, car leurs motivations mathématiques peuvent être exprimées.

Nous qualifions de *manipulation* toute action sur les formules ou leurs constituants. En opérant sur l'une des trois structures, les manipulations de formules de SOFTMATH se font selon le cycle de manipulation présenté figure 3.9.

Le cycle de manipulation

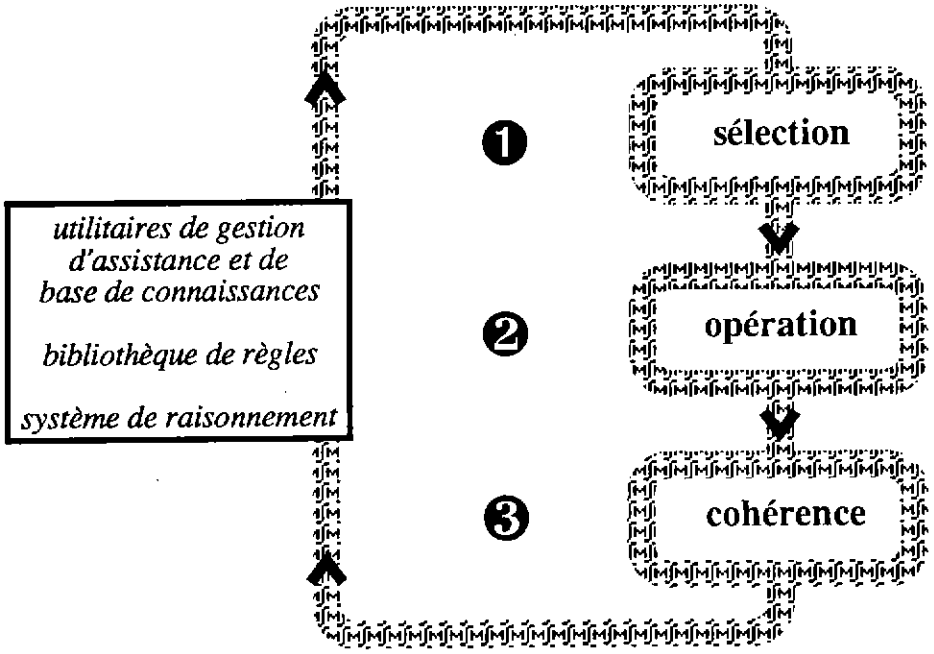


figure 3.9

La phase ❶ effectue une ***sélection*** à l'aide d'une *désignation*. La phase ❷ regroupe les ***opérations*** en vue d'une transition vers une autre formule. Le *calcul* est l'opération qui nous vient principalement à

l'esprit. Il peut être utilisé pour assister la *présentation* d'une formule ou contribuer à sa *résolution*. Enfin, la phase ❸ gère la ***cohérence*** syntaxique et mathématique de l'opération avec le contexte de manipulation (introduction d'une nouvelle formule, modification d'un symbole...). Ces manipulations font intervenir des outils intégrés à l'environnement au cours du cycle de manipulation.

2.2. TYPOLOGIE DES MANIPULATIONS DE FORMULES

2.2.1. Exemples de manipulations

Notre démarche a consisté à fournir potentiellement toutes les fonctionnalités nécessaires à la réalisation d'une interface à engagement direct à l'usage des mathématiques. Le design et l'ergonomie vont caractériser le **"rendu"** de l'interface, mais ils varient selon les techniques (et les technologies) employées. Nous en esquissons ici quelques traits généraux en considérant à notre disposition les techniques les plus sophistiquées. Nous présentons successivement deux enchaînements.

Le premier enchaînement figure 3.10 présente des exemples de manipulation destinés à illustrer l'**avantage du couplage** des structures avec la structure mathématique, par rapport aux traitements basés sur une structure textuelle "classique". Il paraît artificiel, car il se concentre sur ce que les systèmes ne peuvent pas effectuer usuellement. Pour recadrer son apport, il faut imaginer qu'il schématise une formule de dimension et de complexité réaliste, et que les manipulations obéissent à un obscur dessein (selon le point de vue de l'assistant).

Le second enchaînement propose une piste innovante pour des manipulations. Il propose une **sémantique des manipulations** permettant de décrire les intentions du mathématicien. Cela est possible grâce au support des trois structures couplées.

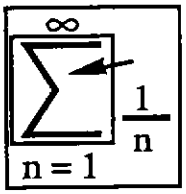
L'efficacité de l'édition structurelle est conditionnée par la mise en œuvre d'une interface sophistiquée permettant de bien visualiser les structures. La structure apparente bidimensionnelle d'une formule sert de support principal aux manipulations de formules, qui travaillent sur les structures textuelles et mathématiques. Ces structures sous-jacentes peuvent toutefois être utilisées comme supports annexes, dans des fenêtres différentes. L'environnement de travail peut ainsi comporter des formules différentes avec des vues multiples et cohérentes d'une même formule.

Les deux enchaînements que nous présentons sont limités à l'expressivité de la fenêtre principale dans laquelle sont manipulées ces formules. Les actions que nous présentons sont souvent renforcées par un **comportement contextuel** touchant l'interface toute entière. Par exemple, pointer des arguments pour une manipulation déclenche un processus de mise en valeur. Cela peut être un procédé visuel (couleur, clignotement, animation, zone spécialisée de description...), sonore (bip, musique...), ou mixte. Il signale éventuellement la nature de l'objet désigné (structure mathématique, syntaxique ou visuelle), affiche la fenêtre de la structure correspondante, et émet un message précisant les informations perdues ou oubliées dans la manipulation (cohérence mathématique, caractéristiques syntaxiques...).

Le lecteur peu familier avec la pratique des mathématiques ne devra pas se laisser abuser par le **déroulement surprenant de cet enchaînement**. Il trouvera, en Annexes, un exemple illustrant des étapes de manipulation similaires respectant une cohérence globale des manipulations au sein d'une démarche de résolution d'un problème ([Graham 89, p 446]).

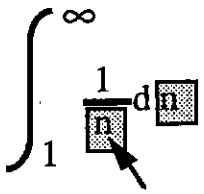
Il faut rappeler aussi qu'un assistant ne maîtrise pas forcément les causalités qui régissent un enchaînement, et que ce déroulement surprenant respecte les observations et les inférences que le système assistant est à même d'effectuer. Enfin, le lecteur mathématicien notera que la transformation de sigma en intégrale est présentée comme une méthode mathématique dans cette Annexe : elle correspond à la transformation I vers III. De plus les branches d'exploration (cf. IV et V) - qu'elles soient fructueuses ou non - correspondent à la démarche heuristique employée par les auteurs.

Exemples de manipulations



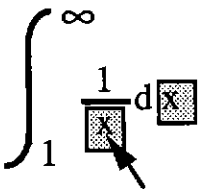
I

remplacement d'opérateur
$\Sigma \rightarrow \int$



II

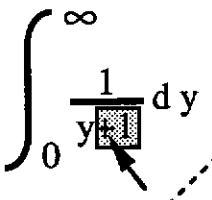
modification de variable
$n \rightarrow x$



III

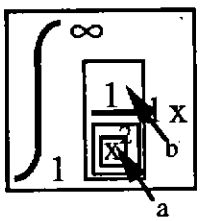
changement de variable
$x \rightarrow (y+1)$

remplacement d'arbre
$x \rightarrow x^2$



IV

retour à III



V

changement de notation

VI

une étape de calcul

$$\int_1^{\infty} x^{-2} dx$$

VII

calcul total

$$\left[-x^{-1} \right]_1^{\infty}$$

VIII

$$1$$

figure 3.10

2.2.2. Diverses possibilités d'opérer

Nous distinguons l'accès, du destinataire. L'accès désigne la structure que voit le mathématicien, le destinataire désigne la structure qu'il manipule mentalement. La configuration papier-crayon a ainsi un accès par la structure apparente et une manipulation principalement sur la structure mathématique. Cela introduit la notion de "travail à distance".

Avec l'informatique, la structure d'accès peut être plus variée, notamment par des facilités de présentation conjointe d'une même formule dans des fenêtres différentes. Il est ainsi possible d'utiliser une présentation arborescente de la structure mathématique de la formule. Toutefois, l'usage préférentiel reste la structure apparente qui est plus synthétique et sert de référence.

Pour ce qui concerne le destinataire, les manipulations doivent pouvoir se penser sur l'une quelconque des trois structures. Le choix dépend du mathématicien et de la structure avec laquelle il pense sa tâche effective.

Les 8 étapes de l'enchaînement figure 3.10 utilisent la **structure apparente comme voie d'accès** aux structures sous-jacentes. La correspondance se fait usuellement par la structure textuelle pour manipuler la structure mathématique et ses constituants. Le mode prioritaire par défaut est de laisser l'analyseur répercuter les actions directement sur la structure mathématique. Cela ne signifie pas pour autant que les manipulations effectuées préservent l'équivalence mathématique entre formules.

Les limites de la structure apparente pour les manipulations résident dans le fait que le mathématicien ne peut pas toujours accéder à la structure mathématique à partir de la structure textuelle associée. Par exemple dans le cas du sigma, le mathématicien ne peut désigner le domaine de la variable directement : il a accès aux bornes via les champs inférieurs et supérieurs de la structure textuelle "Σ". S'il désire changer le domaine de la variable "k" de [1,n] à N (et non à [0,∞[) dans l'exemple : $\sum_{k=1}^{n} \sin^{-n}(x_k)$, il a deux solutions :

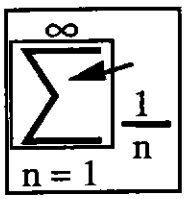
① agir sur la structure textuelle et changer "k=1" en "k∈N". Il obtient une structure incohérente syntaxiquement, c.-à-d. la structure textuelle est signalée ne pas être couplée à une structure mathématique (même avec l'aide d'une complétion éventuelle). Enfin, en supprimant la boîte correspondant à la "partie sup" "n", il rétablit la cohérence ;

② travailler directement sur la structure mathématique et changer le sous-arbre "(intervalle 1 n)" en N. L'accès aux structures sous-jacentes est notamment le seul moyen direct de sélectionner un opérateur "caché" comme la multiplication dans "2n".

Il est possible de **pallier les limites de la structure apparente** par des solutions informatiques. La solution ② de changement du domaine du sigma peut ainsi être spécifiée à partir de la structure apparente. Cela se fait en sélectionnant l'expression de sommation toute entière, demandant la structure mathématique (apparition temporaire du squelette mathématique local), puis en sélectionnant et remplacant le deuxième fils de cette structure mathématique.

Une "trace" visuelle est simultanément produite qui souligne de façon spécifique (couleur...) les parties visibles dans la structure apparente du sous-arbre mathématique, en l'occurence "1" et "n". Lors du remplacement de ce sous-arbre "domaine" par le sous-arbre feuille "N", les nouvelles structures textuelle et apparente associées sont affichées automatiquement dans leur nouvelle présentation.

2.2.3. Les manipulations de construction de formule



I remplacement d'opérateur $\Sigma \rightarrow \int$ II $\int_1^{\infty} \frac{1}{n} d\,n$

L'étape I désigne l'emplacement du symbole "Σ". Lors de toute opération de désignation d'un emplacement grâce au couplage de la structure apparente à la structure textuelle, les encadrements de la boîte correspondante et de ses supérieures hiérarchiques apparaissent. L'étape II est obtenue par sélection puis remplacement du nœud "sigma" par une intégrale : le seul changement effectif concerne donc l'opérateur principal de la structure mathématique, ses arguments restant inchangés.

La structure mathématique de l'intégrale a la même arité que celle du sigma. C'est bien la structure mathématique qui a été sélectionnée et modifiée ici, puisque la présentation de la variable par "n=" a laissé place au "dn" de l'intégrale. Toutefois l'expression II est signalée mathématiquement incorrecte puisque "n" était précédemment convenu de type entier. Les connaissances relatives à la présentation et au type des arguments sont ici rattachées à l'opérateur principal.

Le mécanisme de **remplacement d'un opérateur** par un autre est simple et pratique lorsque le mathématicien cherche à récupérer aussi les arguments. Il permet une transition élégante dans l'étape I. Les arguments de la structure mathématique associée à l'ancien opérateur sont récupérés par le nouveau dans la mesure du possible.

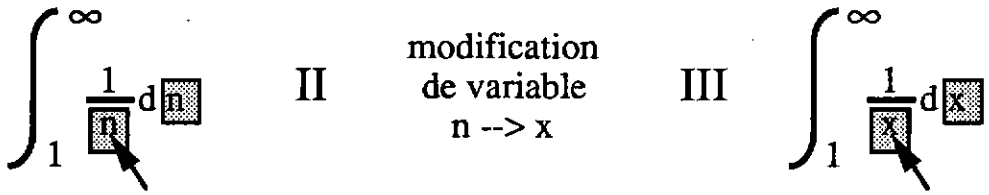
Lorsque le mathématicien remplace l'opérateur "+" de "n+p" par un autre opérateur comme "factorielle", le nombre d'arguments est adapté, dans ce cas seul le premier - "n" - est utilisé. Ces tranformations utilisent la connaissance des arguments des opérateurs ainsi que leur ordre.

Lorsque le "+" est remplacé par "Σ", la structure mathématique est incomplète (le terme principal reste inconnu) et incohérente ("p" n'est pas un ensemble). Cela est détecté en phase ❸ de cohérence du cycle de manipulation de formule. Une structure textuelle est néanmoins construite avec un terme principal indéterminé et comme partie inférieure : "n∈p" ("variable appartient au domaine"). Une autre option à fournir consiste à éliminer les parties incohérentes pour les remplacer par un terme mathématiquement bien formé : ici, introduire un domaine "D" pour obtenir "n∈D". Tout ceci n'est guère intéressant dans cette situation : il vaut donc mieux remplacer l'expression "n+p" toute entière.

Cette transition illustre la distinction entre ***désignation*** et ***sélection*** au sein de la phase ❶ de "sélection" du cycle de manipulation. C'est la distinction entre accès et destinataire. La désignation travaille sur la structure apparente présentée dans la fenêtre. Elle y désigne une entité bidimensionnelle. Cette entité sert de point d'entrée à une ***fonction de translation*** dont le résultat est la sélection effective. Cette fonction de translation permet principalement la correspondance entre les trois structures couplées. La structure apparente contient la trace des boîtes et des pointeurs issus de la structure textuelle.

La fonction de translation utilisée pour la sélection est ici l'*accès direct*, c'est à dire qu'elle cherche à aller à la structure la plus profonde tant qu'il y a une correspondance directe. Dans l'étape I, la désignation bidimensionnelle de "Σ" a transité par la structure textuelle pour aboutir à l'opérateur "sigma" de la structure mathématique. Par contre la désignation de "n=1" aurait aboutit à la structure textuelle car elle n'a pas d'entité mathématique directement associée. Cette fonction de translation simple autorise les manipulations telles que le changement "k=1" en "k∈N". D'autres fonctions de translation sont proposées pour faciliter les manipulations.

Cette transition illustre donc le rôle du **couplage** pour atteindre les structures sous-jacentes dans la phase ❶ et permettre à un changement de la structure mathématique (phase ❷ : opération) d'être répercuté sur les autres structures dans la phase ❸ par des connaissances de mise en forme syntaxique. Enfin, elle montre la souplesse obtenue en gérant et en confinant les incohérences.



La transition de II vers l'étape III s'effectue par modification de la variable "n" par la saisie du caractère "x". Le système va attribuer par défaut, à ce nouveau symbole de variable introduit, le type réel correspondant aux choix pragmatiques actuels. Il va ainsi éliminer "n" en le remplacant par "x". Cette action implicite sur la base de connaissances mathématiques est plus simple que d'éditer les caractéristiques de la variable "n" afin de les ajuster. La formule III est maintenant cohérente.

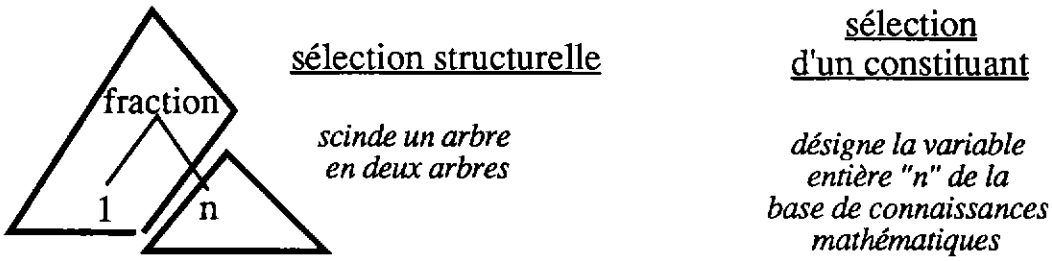


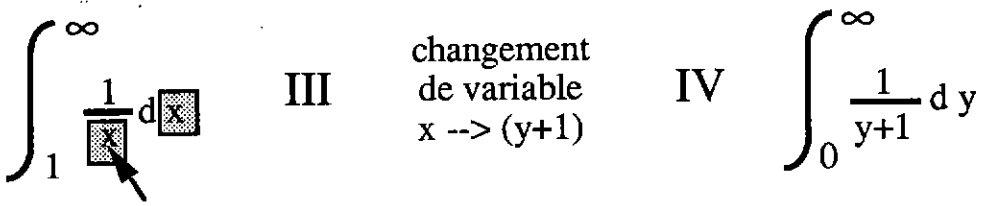
figure 3.11

L'étape II a fait l'objet d'une sélection de la variable "n" afin de rétablir la cohérence mathématique. La figure 3.11 ci dessus présente les deux sortes de sélection sur une structure, dans le cas de la structure mathématique. La **sélection structurelle** désigne un emplacement de la structure mathématique. C'est le procédé visualisé dans l'étape I. La **sélection d'un constituant** désigne le contenu (le "sens") du sous-arbre qui occupe cet emplacement, c'est-à-dire une entité mathématique représentée en dehors de la structure, dans une base de connaissances. Cela peut être une variable, un terme indicé ou une expression.

La sélection d'un constituant est une manipulation importante dans l'activité mathématique. Les structures sous-jacentes ne sont en effet que des structures permettant de représenter les formules : le sens de ces structures est déterminé par les caractéristiques de leurs constituants. Ces constituants (dont les variables) sont regroupés et organisés dans une base de connaissances mathématiques, en partie évolutive en cours de session. Cette base de connaissance contient aussi les lois de formation de structures et de détermination de leur cohérence et de leur sens.

Les informations de cette base de connaissances sont partagées entre plusieurs formules ou vues d'une formule. Ainsi, la sélection de la variable "n", indiquée en grisé étape II, concerne a priori toutes ses occurences, y compris celles présentes éventuellement dans les autres fenêtres (fenêtre contenant une démonstration utilisant cette formule, fenêtre de description des objets mathématiques connus à cet emplacement..., en respectant bien sûr la portée locale sous l'intégrale du symbole "n").

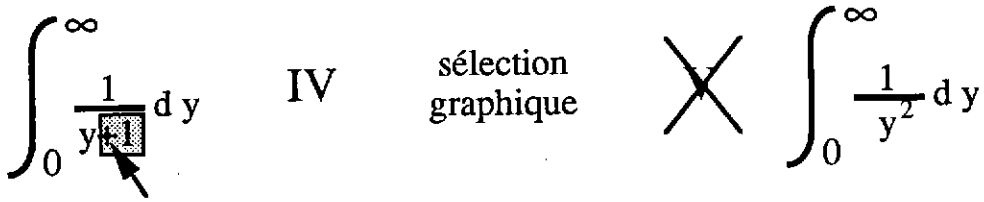
Cette transition illustre les potentialités offertes par les structures sous-jacentes lors des désignations et des sélections. Outre ces manipulations, elles peuvent servir à la mise en œuvre de processus automatiques tel la détection de structures communes. Cela permet de simplifier de longues formules par factorisation, en les substituant par exemple grâce à un paramètre ou à une fonction intermédiaire. De plus, l'étape II aurait pu être évitée par un remplacement d'opérateur plus sophistiqué qui affecte le type de ses opérandes. Les structures proposées permettent la définition - et le recueil dans une bibliothèque - de telles **macro-opérations** qui facilitent l'usage d'un système d'assistance.



Le passage vers l'étape IV se fait par changement de la variable "x" en "(y+1)". Cette substitution qui préserve l'équivalence entre formules est à la base de toute opération mathématique, car elle permet l'instanciation de théorèmes. Elle agit à la fois sur les aspects mathématiques et textuels. Le parenthésage de "(y+1)" est adapté, le "d(y+1)" simplifié, et les bornes du domaine de variation calculées. En cas d'échec du calcul, des constantes "a" et "b" sont générées et introduites comme bornes, accompagnées de leurs équations de définition.

Cette transition illustre les potentialités offertes pour le calcul formel et la mise en œuvre de procédés de raisonnement dans les phases ❷ et ❸ du cycle de manipulation. Ces déductions utilisent des règles ou des théorèmes, et le changement de variable est le procédé de base d'appariement avec une formule. Certaines règles peuvent être directement associées aux manipulations, telles celles de simplification de l'élément différentiel ou celles d'appel à la résolution d'équations simples ("y+1= ∞").

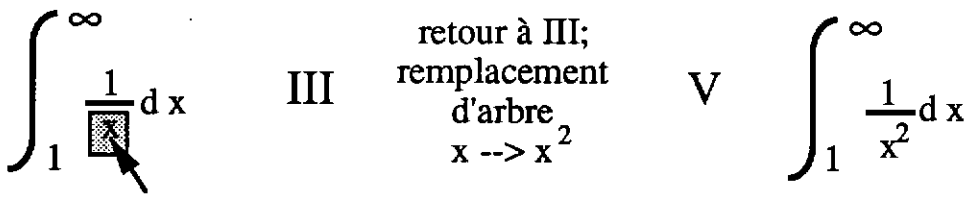
Les structures couplées introduisent une dimension supplémentaire dans ces déductions, par la prise en compte de la **forme** (usuellement normalisée et intangible) et la gestion d'un ***style de présentation*** (phase ❸). Cela se traduit au niveau de la génération de variables, du choix des notations, du parenthésage ou des positions respectives entre constituants. Ces **règles** font aussi partie de la base de connaissances mathématiques. La gestion de la forme et du style intervient lors de la génération de la structure textuelle, mais aussi dans la propagation des manipulations.



L'étape IV montre l'intérêt d'une sélection graphique afin de raisonner directement sur la forme bidimensionnelle pour remplacer "y+1". Elle permet "d'oublier" les structures sous-jacentes pour raisonner uniquement en terme d'entités terminales adjacentes graphiquement. La suppression de la sélection "+1" remplace "y+1" par "y". Alternativement, le remplacement de la sélection se fait par saisie d'un remplacement "...". Elle remplace "y+1" par "y...", en l'occurence ici "y^2". Une action sur la structure textuelle aurait dû remplacer "y+1" après avoir saisi "y" à nouveau. Une autre façon de remplacer partiellement "y+1" est de remplacer directement l'opérateur comme dans l'étape I. Ici, remplacer "+" par "puissance" donne "y^1".

Cette transition illustre la possibilité d'utiliser interactivement la forme bidimensionnelle comme sur les éditeurs classiques. Cette possibilité est renforcée par une analyse incrémentale de la saisie, qu'elle soit linéaire, effectuée au clavier, ou structurée à partir de menus.

Parmi les **facilités** proposées, tout opérateur peut être saisi à partir de son nom (tel "partie entière"), il sera graphiquement représenté par celle de ses notations qui correspond au style de présentation. De plus, ses **propriétés** (de la base de connaissances mathématiques) sont accessibles. Le mathématicien peut par exemple consulter sa définition mathématique, ses notations, ou un résumé informel (disant si c'est par défaut ou par excès, et ce qui se passe pour les nombres négatifs).



L'usager peut se tromper ou effectuer une tentative "pour voir". Ce retour à III rappelle l'intérêt de la gestion d'un **historique** et d'un retour arrière, comme cela est couramment pratiqué en manipulation de programmes et de documents.

L'étape V est construite par remplacement de l'arbre occupant l'emplacement du dénominateur en III. A la différence de la sélection III antérieure, seule cette occurence de "x" est sélectionnée. Cela illustre l'intérêt de la sélection structurelle, qui est un procédé aussi utile que la sélection d'un constituant. La sélection structurelle constitue d'ailleurs le point de passage obligé de toute fonction de translation visant à sélectionner un constituant lors de la phase ❶.

Cette transition illustre l'intérêt de manier les structures sous-jacentes depuis la structure apparente bidimensionnelle. L'**équivalence** mathématique d'une transition est un "effet de bord" possible selon que le mathématicien cherche à construire une formule ou à calculer. Elle fut respectée de III à IV, et aurait pu l'être ici de III à V ("x" en "-(-x)" ou "x^1"). Cette équivalence est l'une des informations donnée à l'issue de la phase ❸ de cohérence, et dépend de l'opération choisie en phase ❷.

2.2.4. Les manipulations d'usage mathématique

Nous travaillons maintenant sur des étapes mathématiquement équivalentes. La phase ❶ de sélection d'un prédécesseur hiérarchique dans une structure sous-jacente à partir d'une désignation en position (a), se fait par appui continu du bouton de la souris jusqu'en (b). La fraction toute entière est alors sélectionnée. Une des manipulations possibles proposée par le menu de *changement de présentation* est l'écriture linéaire de la fraction à l'aide du symbole "/" (phase ❷).

La solution adoptée vers l'étape VI est un changement de notation transformant la structure mathématique par multiplication de l'inverse du dénominateur. Elle est choisie dans le cas où le terme du numérateur est un symbole (avec son signe), alors qu'une solution élémentaire était proposée pour un numérateur égal à 1. A noter, vu la solution adoptée, le simple changement de signe qui condense l'inverse d'une puissance"$(x^2)^{-1}$", déclenché par une règle de calcul formel rattachée à la présentation (phase ❸).

Cette transition illustre la diversité des ressources nécessaires à la mise en œuvre de manipulations naturelles. Les **trois structures** utilisées constamment forment un support indispensable à toute manipulation interactive souple.

Une autre solution consiste à découpler les structures en saisie. C'est à dire, travailler sur l'arborescence des structures, avec visualisation immédiate en génération sur une formule. C'est d'ailleurs un sous-mode d'utilisation des trois structures. Il n'est préférable en général que si les possibilités d'accès aux structures sous-jacentes sont peu "naturelles" (du point de vue expressivité et ergonomie). De plus la structure apparente est beaucoup plus "compacte", ce qui évite de jongler en déplacements structurels dès les formules "moyennes". Rappelons que les cas présentés ici peuvent n'être qu'une partie simplifiée ou schématisée d'une formule de dimension et de complexité réaliste.

Cette transition introduit des éléments d'un **style de présentation**, activé en phase ❸. Ce style est basé sur le choix des notations et des procédés de présentation. Comme c'était déjà le cas pour le changement de variable étape III, les manipulations utilisent des règles de calcul formel usuelles pour simplifier les expressions et adapter la présentation (elles relient ici division, multiplication et puissance).

$$\int_1^{\infty} x^{-2}\,dx \qquad \text{VI} \qquad \begin{matrix}\text{une étape}\\ \text{de calcul}\end{matrix} \qquad \text{VII} \qquad \left[-x^{-1}\right]_1^{\infty}$$

Cette étape VI marque la fin de la séquence de **construction** de la formule, qui se poursuit par une séquence de **résolution** "purement mathématique" de celle-ci au sein d'un contexte de calcul (lois et contraintes entre entités). On remarque ici encore l'utilisation de règles de calcul formel. Alors que SOFTMATH les employait précédemment pour agir sur la présentation des formules (phase ❸), il peut bien sûr s'en servir pour des calculs (phase ❷). Le passage de VI à VII matérialise une étape de calcul, c'est-à-dire l'application d'une règle disponible dans la base de connaissances mathématiques. Elle diffère des règles précédentes par son aspect macroscopique permettant un avancement qualitatif (d'où fréquemment un "titre" de théorème).

Cette transition illustre l'ouverture recherchée par SOFTMATH. Le calcul symbolique n'est pas une boîte noire, mais un ensemble de règles explicites se prêtant à des explications et n'imposant pas de forme normale. Cette caractéristique est bien sûr indispensable pour un usage éducatif comme pour un assistant doté de capacités de communication.

Une gestion spécifique des objets mathématiques variables permet de définir un **mécanisme d'appariement entre deux arbres** de structures différentes dont l'un est plus général que l'autre (comme par exemple ici "$f(y)^n$" et "x^{-2}"). L'utilisation d'une règle est alors l'instanciation d'une formule, et réciproquement toute égalité peut définir une règle. La formule "formule VI égale 1" peut ainsi être introduite comme théorème, et éventuellement justifiée par des transitions.

$$\left[-x^{-1}\right]_1^{\infty} \qquad \text{VII} \qquad \begin{matrix}\text{calcul}\\ \text{total}\end{matrix} \qquad \text{VIII} \qquad 1$$

Les étapes VII et VIII profitent simplement de la structure mathématique pour un couplage avec des capacités de calcul symbolique. L'évaluation par défaut se fait sur la formule toute entière, sinon il faut, comme pour toute opération, désigner l'argument à prendre en compte.

Cette transition illustre simplement l'ouverture de l'approche vers les systèmes de calcul symbolique et de raisonnement. Nombre d'algorithmes et de travaux sont disponibles dans ce domaine, et le développement choisi pour SOFTMATH prévoit une fusion à moyen terme.

2.3. SYNTHESE DES FONCTIONNALITES DE MANIPULATION OBSERVEES

2.3.1. Description des modes de sélection

Afin d'effectuer une synthèse de cet enchaînement de manipulation, nous abordons les trois phases du cycle de manipulation de formule. Les opérations de manipulation interactives reposent principalement sur la phase ❶ de désignation puis sélection grâce à une fonction de translation (distinction entre accès et destinataire), comme cela est présenté figure 3.12.

Il est envisageable de proposer de nombreuses variations sur la sélection, de la même façon que les systèmes de règles de réécriture font des appariements **modulo** certaines règles : par exemple une sélection modulo la commutativité, l'associativité, etc. La sélection modulo la commutativité revient à

sélectionner aussi, si l'opérateur de la sélection principale est commutatif, tous les arbres identiques à une symétrie près (comme "a+b" et "b+a"). Elle permet par exemple de remplacer tous les "0 + ?" et "? + 0" par "?", quelle que soit la valeur de "?".

Plus difficile encore, une sélection modulo des propriétés des constituants, telle l'égalité (à différencier de l'identité : l'égalité fait intervenir un calcul, qui est indécidable dans le cas le plus général). Une autre extension possible est d'associer la racine d'une désignation structurelle comme constituant sélectionné. Cela permet de sélectionner l'opérateur de multiplication à partir de sa forme implicite "2n". La premiere extension est plus utile pour les manipulations mathématiques, la seconde facilite les manipulations textuelles.

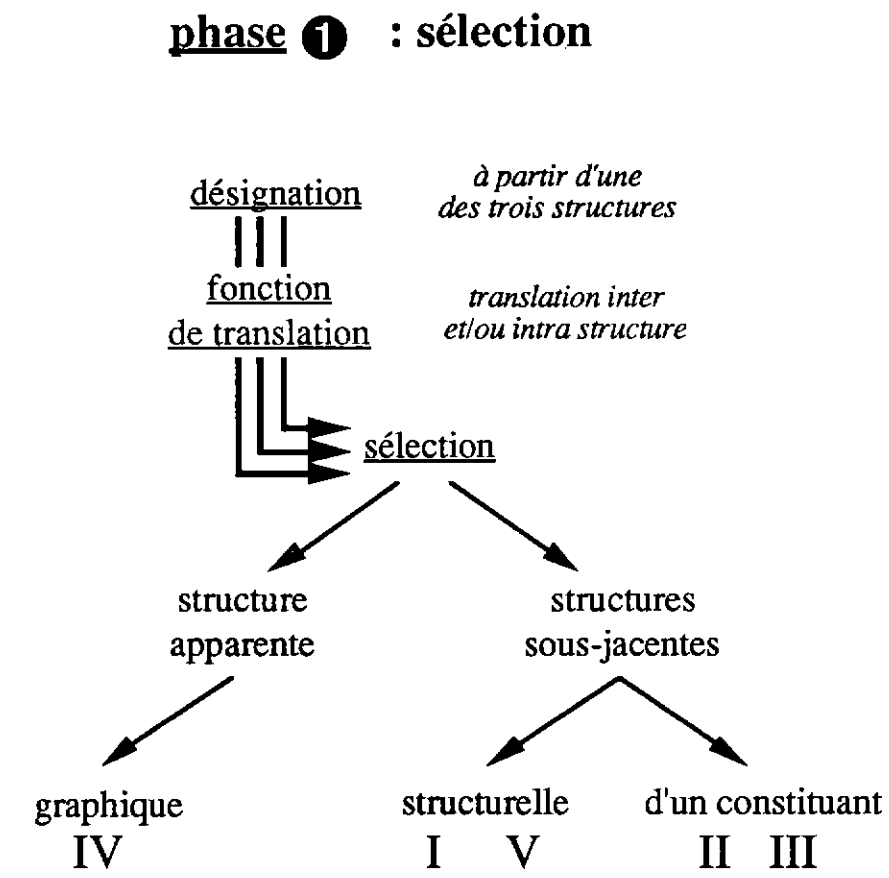


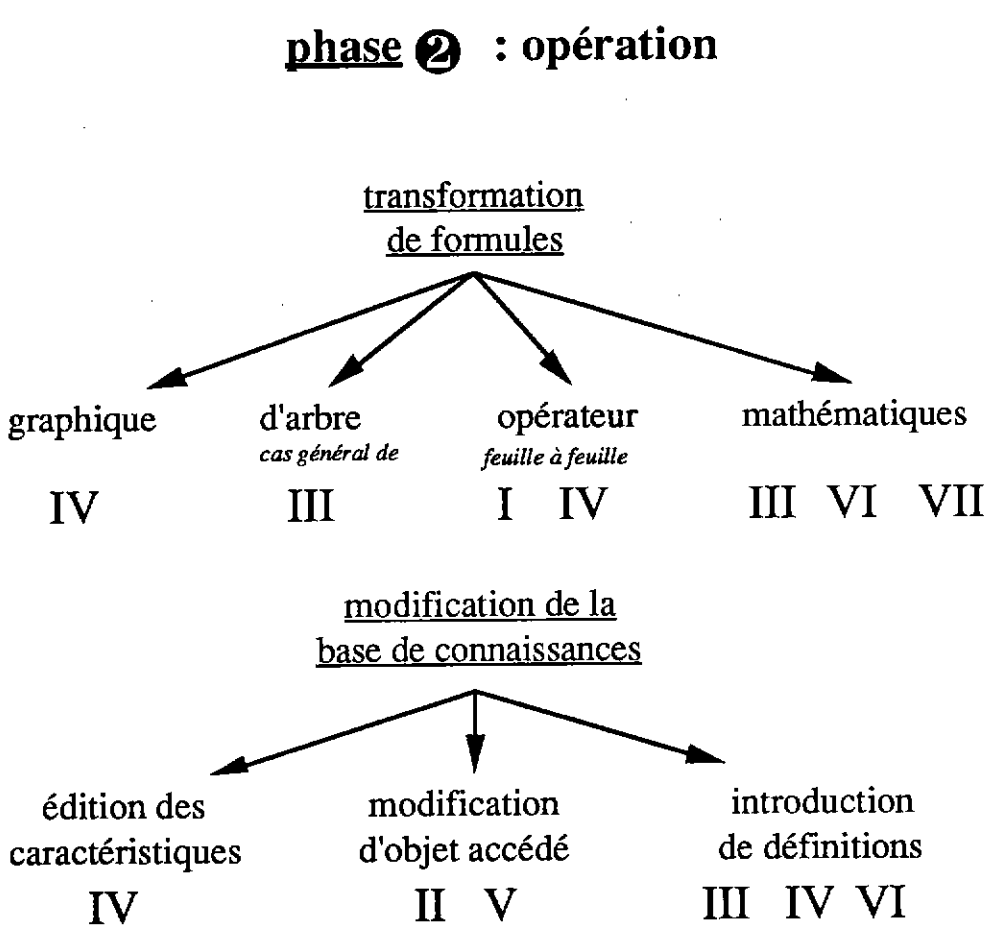
figure 3.12

Certaines fonctions de translation permettent non seulement une translation inter-structure, mais encore simulanément une translation à l'intérieur d'une structure (intra). C'est utile par exemple si le mathématicien veut travailler sur des expressions de telle forme, sur la structure prégnante ou sur des zones d'intérêt. L'assistant a alors la charge de cette translation intra-structure à partir d'une information générée par le mathématicien indiquant qu'il faut chercher dans tel secteur désigné.

En résumé, la **sélection** grâce à des **fonctions de translation** associées, est l'opération primordiale des manipulations. Elle utilise quelques variables d'état :

- structure principale présente dans la fenêtre (formule bidimensionnelle dans cet enchaînement) ;
- hiérarchie des structures manipulées (par défaut l'ordre de priorité décroissant pour la structure qui visée par la sélection est : mathématique > textuelle > apparente) ;
- description et combinaison de fonctions de translation (accès direct, les fils de..., le chemin..., les feuilles, etc.). Il faut en particulier savoir si la sélection est structurelle ou du constituant.

2.3.2. Taxonomie des opérations disponibles



Ce schéma récapitulatif est annoté par le numéro des étapes concernées.

figure 3.13

Les opérations de la phase ❷ peuvent être scindées en deux catégories présentées figure 3.13 :
- les opérations à portée locale concernant la transformation d'une formule ;
- les opérations modifiant le contexte, qui agissent sur la base de connaissances mathématiques. Elles se répercutent sur les formules et agissent ainsi indirectement sur la formule courante.

Les opérations **modifiant la base de connaissances** courante sont :
- édition des caractéristiques, l'opération la plus générale car elle agit directement sur la description informatique d'un objet ;
- modification d'une variable ou d'une règle en vigueur ;
- introduction de définitions : mathématiques, syntaxiques ou macro-opérations de manipulation.

Les opérations de **transformation de formules** sont classifiées en fonction de la distance entre la formule de départ et la formule d'arrivée :
- l'altération de la structure apparente lorsque les deux formules sont voisines graphiquement ;
- remplacement d'opérateur dans la formule, cela peut se partitionner en trois cas, leurs similarités pouvant se situer au niveau de la structure mathématique ou/et textuelle :
  - opérateurs semblables tels "+", "*" et "f(x,y)", ou "Σ" "Π" "$\int$", etc ;
  - généralisation d'opérateur tels "+" —> "Σ", "*" —> "Π", mais aussi "*" —> "puissance", "*" —> "!", "Σ" —> "$\int$" ou "+" —> "polynome", etc ;
  - divers relevant de règles générales ;
- insertion, destruction ou remplacement d'arbre, agissant sur une des deux structures de la formule (opération dans l'univers des structures) ;

- changement de variable et transformations basées sur des règles de calcul (opération mathématique faisant intervenir le sens). Elles incluent la :
  - reconnaissance d'une structure plus générale (appariement ou "matching") ;
  - utilisation d'une règle ou définition (pliage ou dépliage) ;
  - évaluation par un système de calcul formel ou de démonstration automatique.

Les fonctionnalités primitives du noyau de manipulation des structures sous-jacentes sont celles de tout noyau de manipulation d'arbres :

- navigation et sélection dans les arbres ;
- création et destruction d'un arbre c.-à-d. d'une feuille ou d'un nœud avec ses sous-arbres ;
- copie d'un arbre ;
- saisie ou remplacement d'un arbre par un autre.

En revanche, le couplage des deux structures sous-jacentes et la présence de la base de connaissances mathématiques font envisager (via : abstraire et nommer) des manipulations d'une richesse permettant de répondre aux désirs du mathématicien. Cela autorise la mise en place d'un **langage d'expression de commandes** de "haut niveau".

2.3.3. La phase de maintien de la cohérence

La phase ❸ de cohérence se subdivise en une partie dynamique liée au **maintien de la cohérence**, et en une partie statique indiquant l'**état de cohérence** de la formule. Le maintien de la cohérence cherche à déterminer un état cohérent répondant à des critères de présentation prédéterminés. Ce maintien de la cohérence peut être simplement l'analyse de la saisie, une complétion d'information, une analyse de type. Il peut déclencher aussi des règles de calcul pour éliminer une combinaison inopportune ou pour exprimer ce résultat avec un jeu d'opérateurs équivalents. En ce sens, cette phase est l'analogue de la simplification après une opération de calcul symbolique.

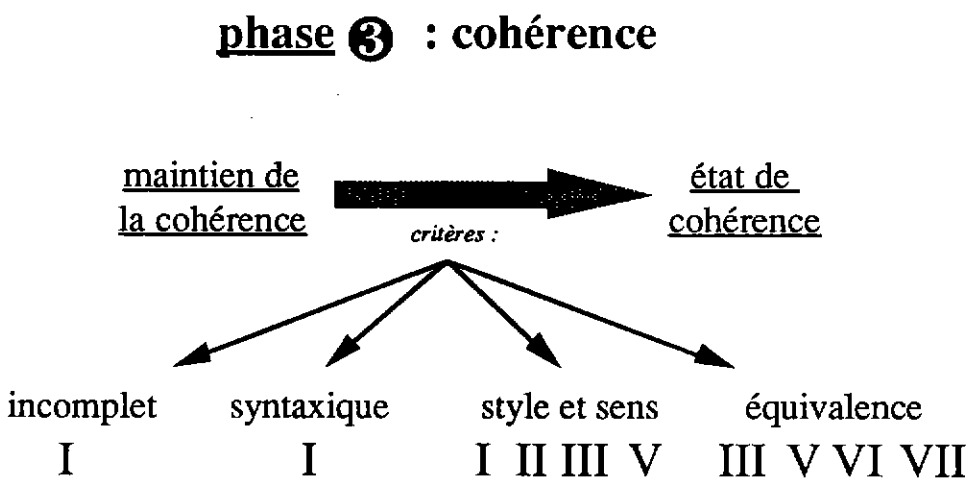


figure 3.14

Comme cela est illustré figure 3.14, cette phase ❸ peut être décomposée en fonction de l'état de la formule par rapport à :

① un manque de constituants (incomplétude) ;

② couplage des trois structures (incohérence syntaxique) ;

③ structure syntaxique ou mathématique (incohérence de style ou de typage) ;

④ la séquence d'opérations en cours (séquence de construction de la formule, ou séquence de résolution préservant l'équivalence mathématique).

Bien entendu, cet état de cohérence dépend en général de l'état précédent, de l'opération effectuée (y compris de ses arguments) et des actions de maintien de la cohérence accomplies.

Bien qu'elles soient de nature classique (création, modification...) les actions proposées par SOFTMATH sont spécifiques aux mathématiques. Il s'agit d'utiliser une définition, générer un ou cent symboles de

variables, ou proposer "f(x,y)" lorsque le système a besoin d'une fonction quelconque à deux arguments. Un **langage d'expression des connaissances** (si possible déclaratif) permet de définir la syntaxe d'un opérateur, d'indiquer comment lever une ambiguïté, etc. La connaissance de la structure mathématique permet d'exprimer des règles de complétion d'informations incomplètes comme l'identification de la variable et le calcul des bornes dans :

$$\sum_i \sin^i(a_i) \text{ et } \sum \sin^{-i}(x_2) \text{ (la variable n'est pas toujours l'indice !).}$$

2.3.4. Récapitulatif des fonctionnalités de manipulation

Notre premier résultat a été de bâtir un **modèle de formules** permettant au mathématicien d'agir simultanément sur la forme et le contenu. Une forme et des choix de présentation adéquats favorisent le déroulement de l'activité mathématique. De plus, la forme sert de point d'entrée aux désignations d'objets mathématiques et autres manipulations paramathématiques. Bien sûr, le contenu mathématique d'une formule est indispensable au raisonnement, mais il peut aussi faciliter les manipulations de forme.

Nous avons alors caractérisé une formule par trois structures couplées : une structure bidimensionnelle apparente, une structure syntaxique et une structure mathématique. Nous présentons quelques mécanismes d'exploitation de ces formules, ainsi que les facilités de manipulation originales qu'elles procurent.

Nous avons défini les connaissances nécessaires à l'exécution des manipulations mathématiques usuelles. Les manipulations de formules s'effectuent selon un cycle ternaire. Ces manipulations seraient alors intégrées à des capacités de calcul formel afin de réaliser, par exemple, des changements de variables, des applications de théorèmes, ou de gérer les bornes des intégrales.

Après la description des opérations de la phase ❷, nous proposons un recensement des **fonctionnalités spécifiques** à l'interface de SOFTMATH. Les diverses étapes de ce premier enchaînement ont illustré les potentialités de la forme bidimensionnelle comme **support principal** de manipulation des structures sous-jacentes. Cet enchaînement a montré comment occasionnellement une structure sous-jacente pouvait être visualisée pour servir de support principal. Quelques points originaux sont :

- action sur l'une des trois structures : mathématique, textuelle ou apparente ; sur les constituants ou leurs propriétés présents dans la base de connaissances ;
- mode d'action prioritaire agissant sur la structure mathématique à partir de l'analyse d'objets variés en entrée (caractères, structure textuelle...) ;
- désignation puis sélection à l'aide d'une fonction de translation ;
- sélection d'un emplacement ou de son contenu (structurelle ou d'un constituant), de certaines ou de toutes les occurences ;
- création d'une nouvelle formule dupliquée ou modification de la même, opération qui préserve ou non l'équivalence mathématique lors de la transition ;
- incomplétude d'une structure (il manque des arguments) et incohérence syntaxique ou mathématique temporaire (les arguments ne conviennent pas) autorisées et gérées ;
- visualisation des structures sous-jacentes et de pointeurs grâce à des procédés de mise en valeur sur la forme apparente ;
- langage d'expression des commandes implicite ou explicite sous une forme "à la PROLOG", afin de manipuler les structures et créer des macro-opérations ;
- utilisation d'une base de connaissances pragmatiques pour faciliter les manipulations mathématiques.

A l'issue de ce premier enchaînement, une taxonomie des **formules "voisines"** peut être formulée. Comme cela a été proposé pour les démonstrations [Chaudron 89], une topologie peut être introduite selon les transformations à effectuer. Les formules se comparent en fonction de leur différences :

- les vues différentes (et partielles) d'une même formule ;
- les formules dont la structure mathématique est identique, mais dont la structure textuelle est différente ;
- les formules dont les structures mathématiques ou textuelles sont "proches" ;
- les formules dont les structures mathématiques sont mathématiquement équivalentes dans un contexte de calcul.

2.4. LA DESCRIPTION DES INTENTIONS PAR L'EXEMPLE

2.4.1. Présentation de mécanismes originaux

Alors que les parties I et II ont décrit la réutilisation de preuves, nous nous concentrons ici sur l'intérêt et les possibilités de réutilisation des manipulations de formules. Lorsque cette réutilisation n'est pas simplement de la récupération, il est nécessaire de généraliser les manipulations de formules.

L'enchaînement 1 a présenté les manipulations primitives autorisées par l'apport des trois structures couplées. Ce deuxième enchaînement illustre la puissance potentielle des manipulations de formules en introduisant des abstractions susceptibles de les rapprocher des préoccupations du mathématicien. Il s'agit de proposer une sémantique des manipulations permettant :

La description des intentions du mathématicien par l'exemple

Cet enchaînement 2 présente des transformations de formules basées sur une succession de manipulations élémentaires. Il propose d'appliquer des techniques d'apprentissage symbolique à l'usage des manipulations de formules. Cet enchaînement est conçu pour susciter l'intérêt des **manipulations pour les manipulations**, et non seulement pour un calcul. Cette envie est la meilleure justification possible de l'approche Génie Logiciel proposée, par opposition à l'approche "calculatoire".

enchaînement de transformations

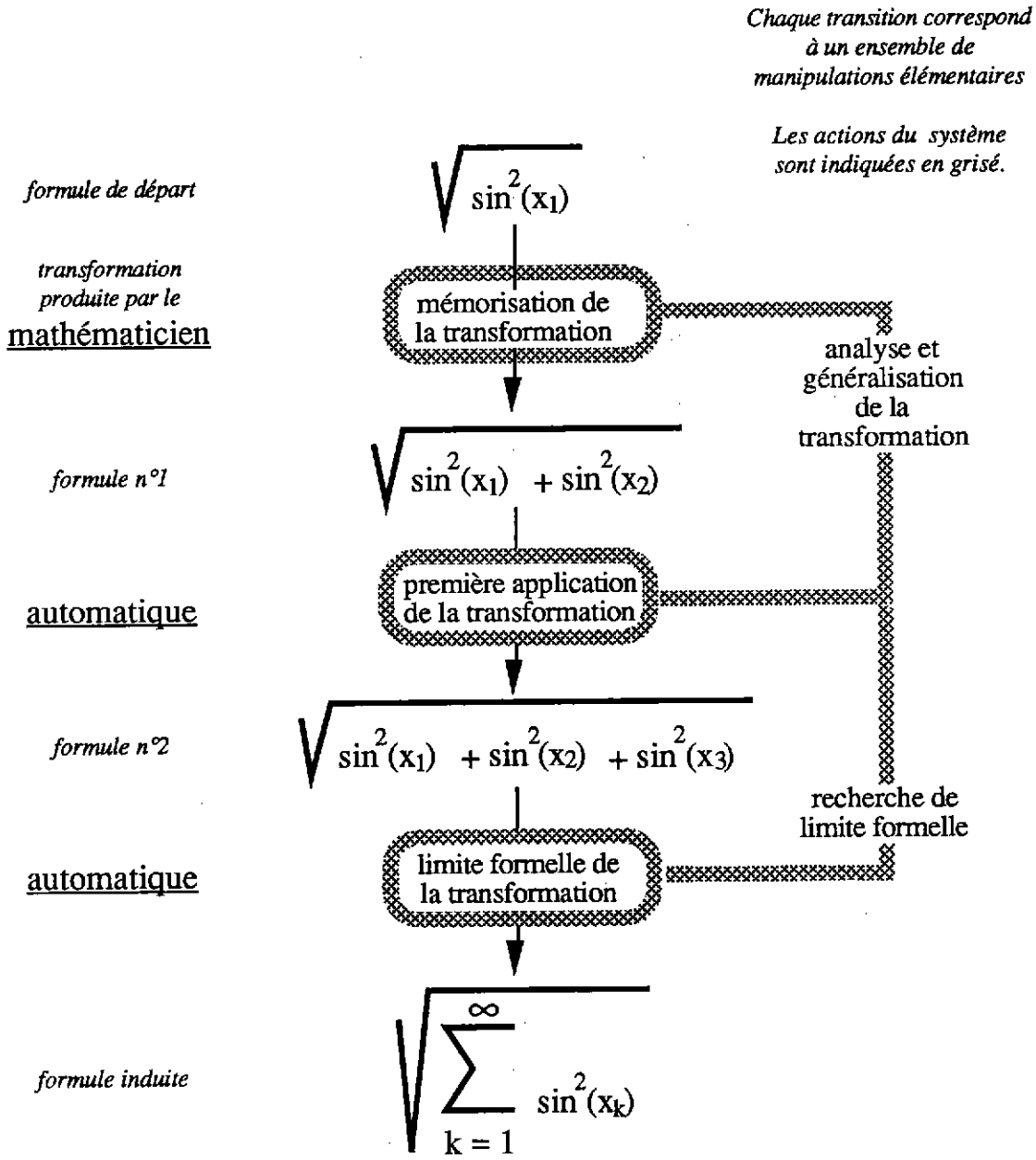


figure 3.15

L'enchaînement 2, dont la séquence de transformations perçue par l'utilisateur est schématisée figure 3.15, présente deux applications nouvelles :

- L'**apprentissage par manipulation directe** comprenant une phase de mémorisation de la transformation qui sera reproduite dans les phases d'application.
- Un **mécanisme de généralisation**, indispensable pour l'extraction du terme général d'une suite à partir de ses premiers éléments.

Les fonctionnalités nécessaires à la réalisation de cet enchaînement nous permettent de détailler respectivement ces deux points. Nous commençons donc par détailler le point de vue du système.

2.4.2. Apprentissage par manipulation directe

L'apprentissage par manipulation directe permet au mathématicien d'utiliser les manipulations de formules qu'il doit nécessairement effectuer comme un langage de description de ses intentions. A partir de l'observation des manipulations directes du mathématicien, le système doit être capable de :

- déterminer la classe de formules pour laquelle ces transformations sont applicables.
- généraliser la manipulation effectuée à partir des intentions reconnues.
- appliquer des manipulations intentionnelles reconnues ou mémorisées.
- traduire les invariants et la dynamique d'une manipulation par une formule générique.

Il y a alors apprentissage en ce sens que le système observe le mathématicien, interprète et généralise ses actions, et mémorise le résultat ainsi obtenu.

L'intérêt d'un tel système est qu'il est très simple à utiliser du point de vue du mathématicien, grâce à une manipulation "banale". La qualité de l'information exploitable vient de l'observation du mathématicien par le système. Le système mémorise toutes les étapes de manipulation, et non seulement les formules initiales et finales. Le mathématicien peut alors **préciser le sens de la transformation grâce à son procédé de construction.** Cela limite fortement les possibilités de généralisation et transforme ainsi un problème contextuel en problème formel. En effet, prenons le cas trivial de la séquence de nombres :

$\boxed{1}$ $\boxed{2}$ $\boxed{3}$?

Le problème classique qui consiste à la caractériser entièrement à partir de son début est objectivement insoluble. Le nombre suivant est : $\boxed{5}$, si la séquence représente l'ensemble des diviseurs de 30, les entiers non nuls et non carrés, les nombres premiers, ou les nombres n tels que $2^n - 1$ soit premier, etc.

Les séquences de formules sont nettement plus complexes que les séquences de nombres. Toutefois, le mathématicien a indiqué beaucoup d'informations sur la facon de construire $\boxed{2}$ à partir de $\boxed{1}$: il a construit la ***fonction de transformation***. C'est cette fonction de transformation qui est utilisée comme intention pour calculer les termes suivants de la séquence. Comme nous allons le voir, un tel procédé est généralisable pour des formules plus complexes, non réduites à un seul symbole.

Notre objectif est de fournir des manipulations des trois structures suffisamment riches pour que la construction par manipulation directe puisse produire des fonctions de transformation respectant les intentions du mathématicien. Cela nécessite donc des connaissances concernant les séquences de nombres usuelles, les manipulations usuelles, et surtout concernant le traitement des suites récurrentes.

Les actions du système lors de cet enchaînement sont illustrées figure 3.16. Les actions internes sont numérotées en noir tandis que les productions de l'enchaînement apparent le sont en blanc.

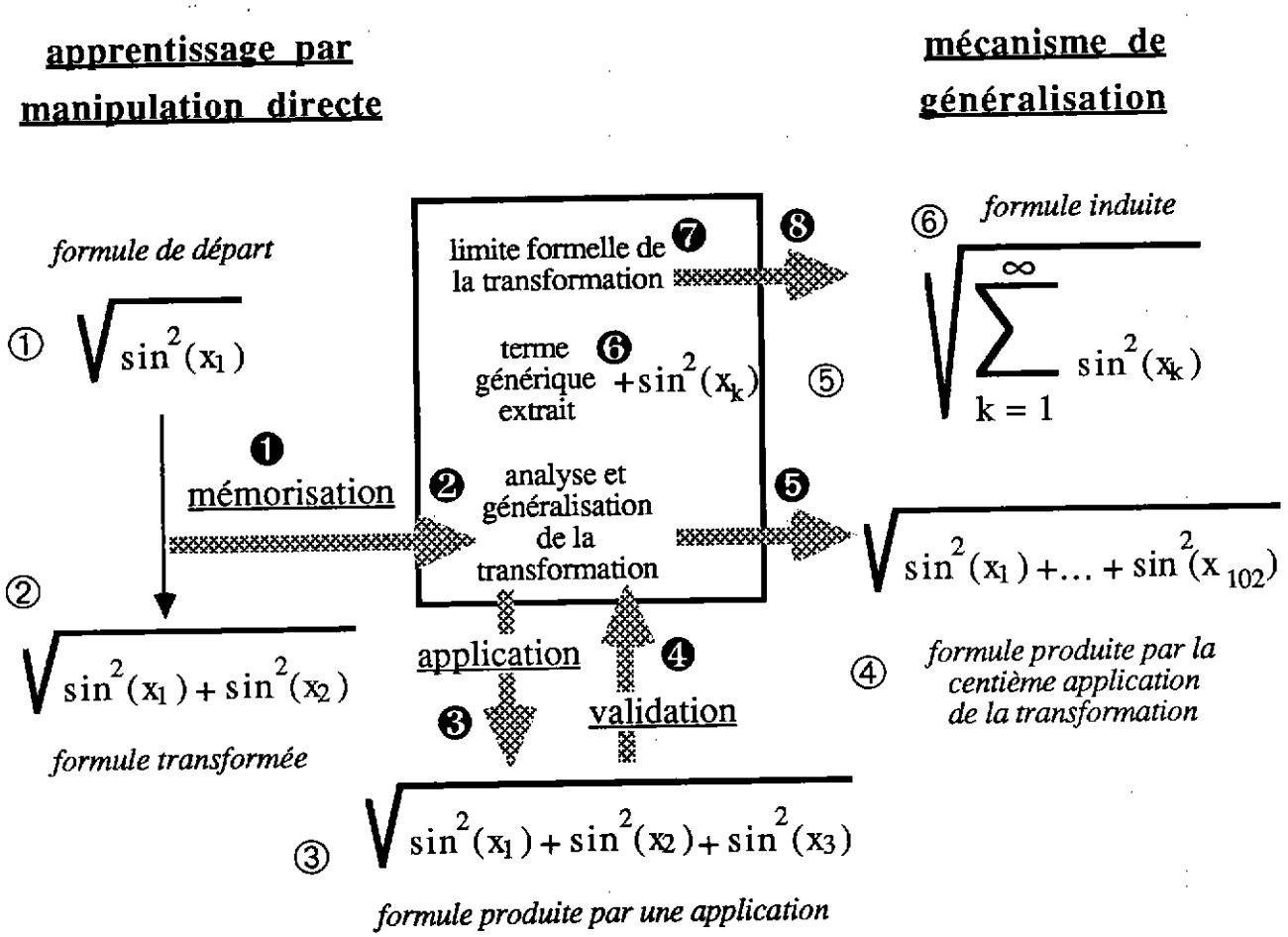


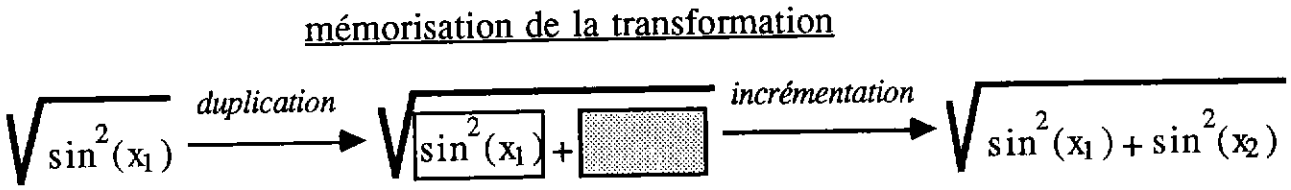
figure 3.16

Le terme "apprentissage" est ici employé au sens d'observation d'une transformation. La transformation de ① à ② effectuée par le mathématicien, est mémorisée par le système (❶) et constitue la base des autres transformations. Cette construction ❶ est analysée et les manipulations dépendantes de l'étape d'application sont généralisées. Cette transformation "rectifiée" ❷ est alors appliquée (❸) pour produire l'étape suivante. Cette formule ③ est validée par l'utilisateur si elle correspond à son intention (❹).

La transformation peut alors être appliquée de nombreuses fois (❺) pour produire une formule de rang important (④). L'intérêt principal est néanmoins la possibilité (❻) d'**extraire le terme générique** (⑤) de la transformation (s'il existe) et d'aboutir à la limite formelle (❼) représentant le point fixe de la transformation. Son application ❽ permet de produire la formule induite ⑥ à partir de la transformation de la formule de départ ①.

2.4.3. La généralisation d'une transformation

Les principales étapes de transformation de l'enchaînement 2 sont maintenant reprises et détaillées afin d'en comprendre les mécanismes et d'apprécier leur intérêt.



Cette première étape de l'apprentissage par manipulation directe détermine le procédé de construction qui est mémorisé sous forme d'une suite de commandes ❶, puis analysé ❷ afin d'être reproduit dans les phases d'application. La formule de départ à gauche sert de base pour la production de la première étape. Un opérateur "+" est inséré, et la construction utilise ici une portion de la formule de départ par duplication. Cette portion dupliquée est alors adaptée par accès puis incrémentation de l'indice de "x_1".

Cette transition illustre le rôle des trois structures couplées pour l'expressivité et l'exécution de cette phase de mémorisation. Dans l'exemple proposé, il faut insérer l'opérateur "+" puis sélectionner et insérer la partie à dupliquer. Ces manipulations sont plus naturellement effectuées sur la structure textuelle. Dans un deuxième temps, il faut accéder à l'indice et l'incrémenter de un. Ces deux opérations correspondent à une manipulation de la structure mathématique.

Le but d'une telle transformation est de **construire aisément une nouvelle formule** mathématique plus complexe. L'équivalence mathématique n'est bien sûr pas de mise. Les actions sur la structure textuelle servent d'intermédiaire pour bâtir la structure mathématique sous-jacente désirée, en gardant éventuellement des informations syntaxiques telles le style ou les notations.

Un **mode inductif** est utilisé pour produire ❷ à partir de ❶. Il permet ici de généraliser la transformation de "1" en "2" par une opération d'incrémentation. Outre cette option "suite arithmétique", une option "suite géométrique" est fournie, sinon l'usager doit spécifier la formule donnant la transformation (ici remplacer "1" par le résultat du calcul de "successeur(1)"). Ce mode inductif permet d'utiliser pleinement les avantages de la manipulation directe, sans recourir à un langage de commande spécialisé pour décrire la transformation. La manipulation directe permet d'exprimer avec **simplicité et interaction** une transformation par ailleurs complexe.

application de la transformation

$$\sqrt{\sin^2(x_1)+\sin^2(x_2)} \xrightarrow{\textit{répétition}} \sqrt{\sin^2(x_1)+\sin^2(x_2)+\sin^2(x_3)}$$

Cette étape est effectuée par répétition de la transformation mémorisée puis analysée ❷. Ce mécanisme est semblable à celui utilisé dans les éditeurs de texte ou de programme lors de l'application d'une "macro", c.-à-d. d'une séquence d'opérations pré-enregistrées. La formule produite sert de validation de la transformation ❷. Un mode d'application pas à pas permet une modification directe éventuelle de la séquence, en cas d'un effet produit non désiré.

Cette transition illustre les potentialités offertes par les trois structures couplées, plus particulièrement ici de la structure mathématique, par rapport à un traitement textuel classique. Combien de fois avez-vous souffert d'un effet de bord pernitieux lors d'une substitution de chaîne de caractères ? C'est que l'éditeur utilisé ne permettait pas d'exprimer un équivalent sémantique de votre transformation, et devait se contenter d'un succédané textuel. Par exemple, une substitution textuelle de "1" par "2" donne a priori une "mauvaise"[1] transformation. Tout d'abord la mémorisation de la transformation doit être dans un mode inductif, afin que l'analyse de la transformation généralise ce changement en incrémentation. De plus, l'incrémentation désirée doit être bien localisée, sous peine de transformer aussi les carrés en cube à cette étape de répétition !

La précision recherchée par SOFTMATH permettrait d'**exprimer exactement les transformations souhaitées.** Elle est liée à l'expressivité du langage d'expression des commandes (d'usage souvent implicite) servant à exprimer les manipulations composant cette transformation. Atteindre par exemple,

---

[1] Ici comme en programmation, il n'y a pas d'erreur : il n'y a que des opérations erronées par rapport à celles que l'usager désire écrire, ou par rapport au sens qu'il désirait obtenir ! Ce biais là est incontournable, et l'ambition de SOFTMATH consiste à fournir à l'usager les moyens d'exprimer sans **biais supplémentaire** le sens qu'il désire obtenir. Par exemple, notre traitement de texte actuel vient de montrer ses limites lors du remplacement du terme "session" par "enchaînement". Le changement du genre (féminin en masculin) a occasionné de nombreuses modifications (et erreurs) lors de ces substitutions, et a requis une attention soutenue en recherche pas à pas. Cette tâche fastidieuse nous est **imposée** par le biais supplémentaire résultant de l'incapacité du système de traiter cette situation, en connaissant et en appliquant le changement de genre et les multiples subtilités qui l'accompagnent.

certains "x_2" d'une formule plus complexe (sans avoir à décrire leur position, en la décrivant conditionnellement en terme de la structure mathématique...), afin de les incrémenter. La puissance du système provient alors de sa faculté d'exploiter cette précision et d'utiliser le calcul en vue de manipulations (comme le calcul des itérés de successeur).

limite formelle de la transformation

$$\sqrt{\sin^2(x_1)+\sin^2(x_2)+\sin^2(x_3)} \xrightarrow{\textit{extraction du terme générique}} \sqrt{\sum_{k=1}^{\infty} \sin^2(x_k)}$$

Un mécanisme de généralisation est utile pour proposer le terme général d'une suite à partir de ses premiers éléments. Il peut alors être complété d'un logiciel d'extraction de la limite formelle pour une classe donnée de transformations et d'opérateurs mathématiques. Même dans un cas aussi direct que celui de l'exemple, il fait appel à des connaissances mathématiques sur les entiers (ici "successeur") ou sur le lien entre un squelette d'arbre additif et l'opérateur "sigma".

Il est souvent plus intuitif et moins lourd de raisonner d'abord sur un cas simple. Cette possibilité de limite formelle peut être utilisée pour manipuler en dimension 2 ou 3, puis généraliser en dimension quelconque. Elle constitue un prérequis à la généralisation d'une preuve toute entière de dimension 2 ou 3 à une dimension quelconque.

Cette transition illustre la puissance due à la connaissance de la transformation par rapport à la simple connaissance d'une séquence de formules. L'avantage et la faisabilité du procédé de généralisation est dû à **la connaissance de la fonction de transformation.**

Pour cela, il faut appliquer les techniques d'Intelligence Artificielle concernant l'apprentissage par observation et par explications. Le principe est simple et plus efficace que l'apprentissage à partir d'exemples lorsqu'il est utilisable. Pour rappel, l'apprentissage par explication consiste, dans le cas d'une démonstration mathématique, à généraliser chaque étape d'une preuve afin d'obtenir le théorème le plus général correspondant à cette preuve. Pour être efficace, il doit faire aussi appel à des transformations de nature différente, comme produire un autre segment de preuve lorsqu'un résultat reste vrai mais que sa preuve n'est plus applicable.

2.4.4. Expressivité des manipulations proposées grâce aux trois structures

exemples d'autres transformations

$\sqrt{\sin^2(x_1)}$ *génère* $\sqrt{\sin^2(x_1)+\sin^2(x_2)}$ *par ajout du + après l'argument de la racine*

F2 $\sqrt{\sin^2(x_1+x_2)}$ *par ajout du + au fond à droite de l'arborescence*

F3 $\sqrt{\sin^2(x_{10})+\sin^2(x_{20})}$ *F1 puis multiplication des indices par 10*

Cette étape illustre deux variations possibles de la séquence principale de l'enchaînement 2. La formule F2 illustre qu'il est possible d'agir à n'importe quel niveau de l'arborescence. Le mathématicien obtient ainsi à l'étape suivante "$\sin^2(x_1+x_2+x_3)$" car il a décrit implicitement la suite (arithmétique) des entiers naturels en ajoutant le numéro de l'étape au nombre de départ : 1.

La formule F3 est intéressante car l'application suivante dépend de la transformation : cela se produit lorsqu'un élément de référence lors de la construction est réflexivement modifié lors de celle-ci. En effet, dans la phase de duplication de "$\sin^2(x_1)$" puis incrémentation de l'indice, qu'est-ce qui est mémorisé ? L'accès au premier terme de la formule courante étape i (dite *récursion*) ou l'accès au premier terme de la formule de départ (dite *itération*) pour l'incrémenter i fois.

En effet, dans une application récursive de la transformation, les opérations de manipulation sont effectuées à partir de la formule courante ; en particulier, c'est le premier terme courant qui sera dupliqué, et non le premier terme de la formule de départ :

- Pour la **récursion**, le terme nouveau se formera à partir de "x_{10}" et sera incrémenté du numéro de l'étape - 2 - puis multiplié par 10 pour donner "x_{120}".
- Pour l'**itération**, la base de la transformation est "x_1", ce qui donnera "x_3" puis "x_{30}".

Le problème qui nous intéresse lors de ces manipulations consiste à pouvoir donner toutes les informations dès la première étape de transformation. Par exemple, la nuance entre récursion et itération peut être spécifiée lors de la manipulation directe par la sélection structurelle (récursion car action dans l'espace des structures) ou du contenu (itération) du dit premier terme. Il est néanmoins toujours possible de décrire explicitement ces transformations.

D'autres séquences de formules peuvent être générées à partir de la même formule de départ avec d'autres transformations. Certaines séquences basées sur des transformations différentes peuvent néanmoins être identiques ou coïncider localement. L'étude des transformations possibles et de leurs séquences engendrées n'est potentiellement limitée que par l'imagination de l'auteur. Leur intérêt est notamment que le mathématicien peut connaître le procédé de génération sans connaître la forme du terme général.

Cette transition illustre les potentialités offertes par une exploitation judicieuse des trois structures. Elle est conditionnée par le langage d'expression de commandes que ces structures peuvent supporter. L'usage de "macros" peut alors développer tout son potentiel. Par exemple, une transformation peut faire appel aux capacités de calcul mathématique sur la formule produite à partir d'un jeu d'écriture textuel.

2.4.5. Une sémantique des manipulations en vue de leur généralisation

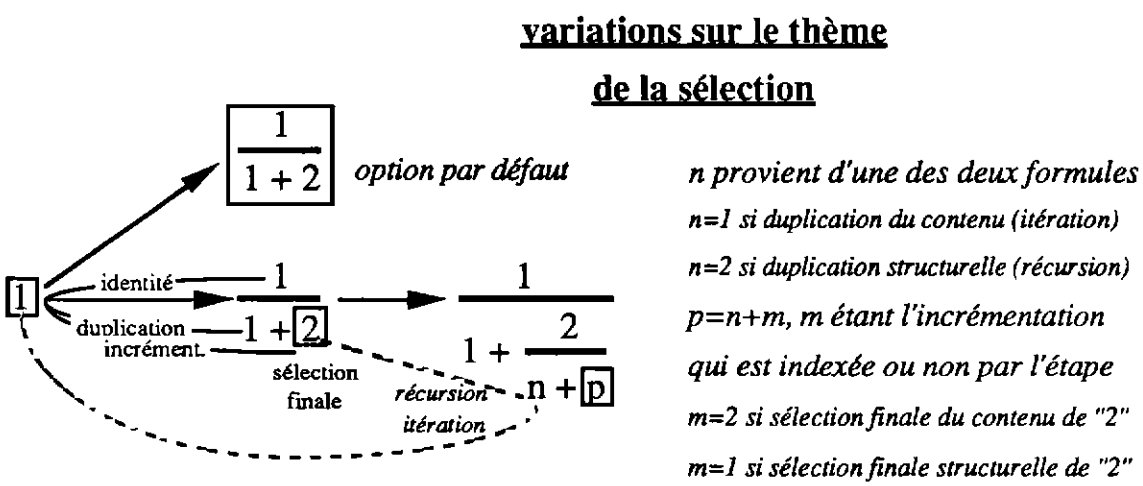


figure 3.17

L'expressivité des manipulations nous permet d'envisager **un langage de contrôle par manipulation directe.** Cette étape figure 3.17 illustre le rôle de la sélection dans l'application d'une transformation. La *sélection finale* est l'entité sélectionnée à la fin de la mémorisation de la transformation (par défaut toute la formule). Cela correspond à la première transformation, nous nous intéressons maintenant à la seconde afin de déterminer la valeur effective de n et p dans la fraction.

La transformation a consisté ici à construire la formule avec "/" et "+", en dupliquant deux fois la formule de départ. Pour le deuxième argument de "+", une incrémentation a été rajoutée. Les modes de sélection lors de la transformation vont influer sur l'application ultérieure.

Le cas de la valeur n commenté sur la figure est aussi celui de l'étape précédente. Elle vaut "1" si la duplication de "1" se réfère à l'étape initiale, ou "2" si elle se réfère à l'étape courante.

Le cas de la valeur p est plus complexe car il fait aussi intervenir la nature de la sélection finale. En effet dans l'exemple "$\sin(x_1)$", l'incrémentation est itérée à chaque application (dite *indexée*), elle pourrait ne pas l'être pour ajouter indéfiniment "$\sin(x_2)$".

Ici encore, une convention sémantique a été choisie : si la sélection finale est structurelle, les fonctions de transformation qui altèrent les parties dupliquées ne sont pas itérées (non indexée), si le contenu est sélectionné, elles sont itérées (application indexée). Dans le cas non indexé, on incrémente de 1 quelle que soit l'étape, sinon, on incrémente i fois de 1 à l'étape n°i.

Cette convention est illustrée avec les deux premiers cas ci-après : dans le deuxième cas, lorsque c'est le contenu (le sens) de la sélection finale qui est préféré, c'est "l'esprit" (incrémentation) de la transformation qui est appliqué.

$$\cfrac{1}{1+\cfrac{2}{1+\cfrac{2}{1+\boxed{2}}}}$$

itération non indexée

$$\cfrac{1}{1+\cfrac{2}{1+\cfrac{3}{1+\boxed{4}}}}$$

itération indexée

$$\cfrac{1}{1+\cfrac{2}{2+\cfrac{3}{3+\boxed{4}}}}$$

récursion non indexée

$$\cfrac{1}{1+\cfrac{2}{2+\cfrac{4}{4+\boxed{7}}}}$$

récursion indexée

Finalement nous obtenons selon les sélections effectuées dans l'exemple figure 3.17 les quatre cas ci dessus (il y en aurait huit si les 2 duplications sont découplées). L'**itération non indexée** correspond à une construction régulière, alors que l'**itération indexée** présente un intérêt fréquent dans les manipulations.

La **récursion non indexée** n'utilise que des sélections structurelles et est purement récursive. Elle correspond à une forme naturelle d'expression des constructions. Le nouveau terme serait obtenu par la même transformation à partir de la formule de départ "3". Il semble d'ailleurs normal que lorsque l'usager ne s'intéresse qu'à des structures et non à leur contenu mathématique (par le biais d'une indirection sémantique), ces transformations soient des applications récursives de l'espace de ces structures. La **récursion indexée** introduit simplement une récursion à plusieurs niveaux.

Les variations sur les autres sélections finales de l'exemple ne sont pas traitées ici. Dans le cas de la transformation $\boxed{\sin^2 x_1}$ donne $\sin^2 x_1 + \boxed{\sin^2 x_2}$, les quatre cas seraient obtenus en prenant les chiffres de droite, par ex. $\sin^2 x_1 + \sin^2 x_2 + \sin^2 x_4 + \sin^2 x_7$ pour le quatrième. L'intérêt pratique de la récursion est donc surtout anecdotique : c'est plutôt une curiosité mathématique s'apparentant à des séries récurrentes, avec une possibilité d'étude expérimentale par simulation visuelle.

Cette transition illustre les potentialités offertes par les transformations possibles. Elles mixent itération et récursion, structure et contenu mathématique. Elles rappellent les programmes destinés à la mutation de programmes. Vu leurs caractéristiques à la fois mathématiques et structurelles, elles peuvent donner lieu à de véritables jeux de codage et de combinaison de transformations.

2.5. L'USAGE DE GENERALISATIONS EN MATHEMATIQUES

2.5.1. Présentation des généralisations possibles

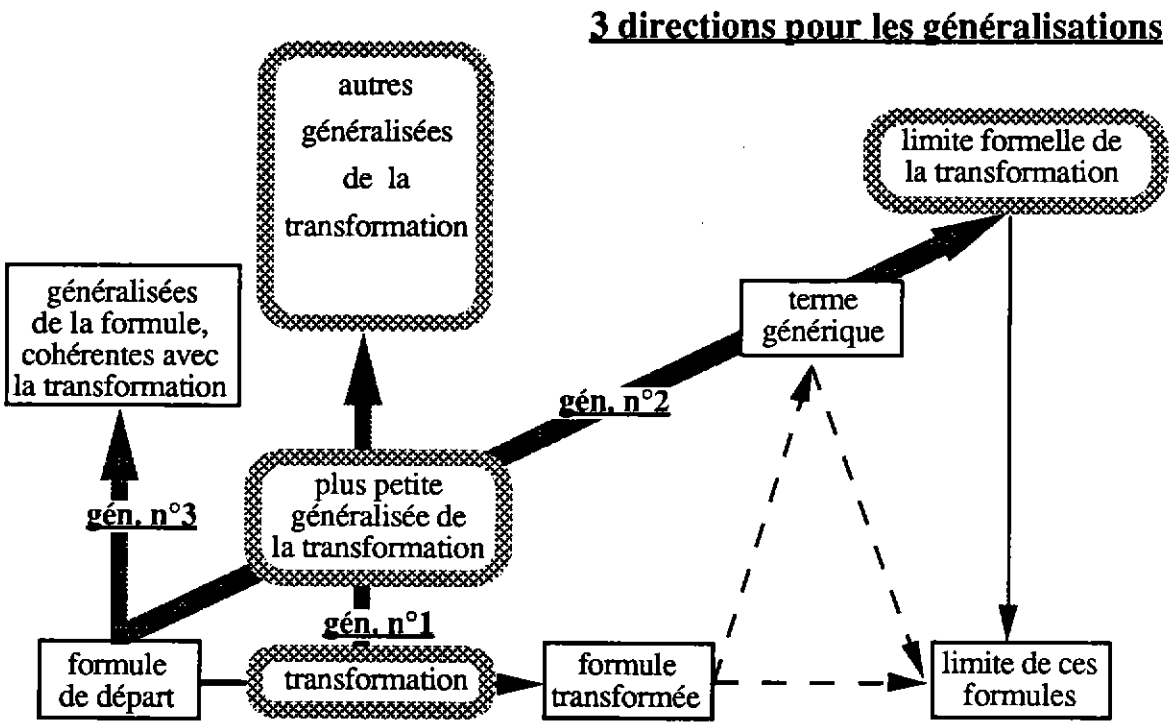


figure 3.18

Nous disposons maintenant des éléments suffisants pour introduire figure 3.18 les trois directions possibles pour les généralisations. Une ébauche de *généralisation n°1* a déjà été introduite pour généraliser la transformation de "1" en "2" par une opération d'incrémentation.

La *généralisation n°2* constitue la ligne de force de cet enchaînement. C'est elle qui permet l'application successive de la transformation sur la formule de départ, et le passage éventuel à la formule correspondant à la limite formelle.

La *généralisation n°3* est d'ordre mathématique. C'est elle qui fait passer par exemple de "$\sin(2x_1)$" à "$f(x_1)$". Elle permet de généraliser non pas la transformation, mais la formule elle s'applique.

Cette transition illustre les potentialités offertes par les manipulations mathématiques en matière d'apprentissage. Ces généralisations possibles sont effectuées **à partir de** manipulations et **pour** des manipulations. Elles diffèrent de l'usage courant de l'apprentissage en mathématiques à des fins directement calculatoires. Les généralisations n°1 et 3 font maintenant l'objet de nos investigations.

La **généralisation n°1** est principalement là pour faciliter l'expression par manipulation directe. En effet, il faut analyser une transformation effectivement réalisée à partir de la formule de départ, pour obtenir une transformation générique indépendante de l'étape (généralisation n°1). L'avantage de la manipulation directe est de ne pas se préoccuper de cela, bien que la production de la transformation générique puisse être effectuée lors de la manipulation directe. Des options sont choisies lors de la mémorisation pour connaître certains paramètres de généralisation souhaités, c.-à-d. caractériser la sémantique des manipulations dans l'optique d'une généralisation n°1.

2.5.2. Des critères de décision pour une généralisation sémantique

La **généralisation n°3** s'intéresse aux généralisations "pertinentes" d'une formule. Pour cela, elle adopte une échelle "semi-logarithmique" dont les sauts successifs correspondent à des étapes qualitatives. Ainsi, passer de "sin" à une variable fonctionnelle "f" néglige des propriétés de "sin" comme sa parité ou

sa périodicité. Ce point s'affine éventuellement si besoin par un **dialogue** avec l'usager, qui permet de refuser une généralisation pour choisir un autre chemin.

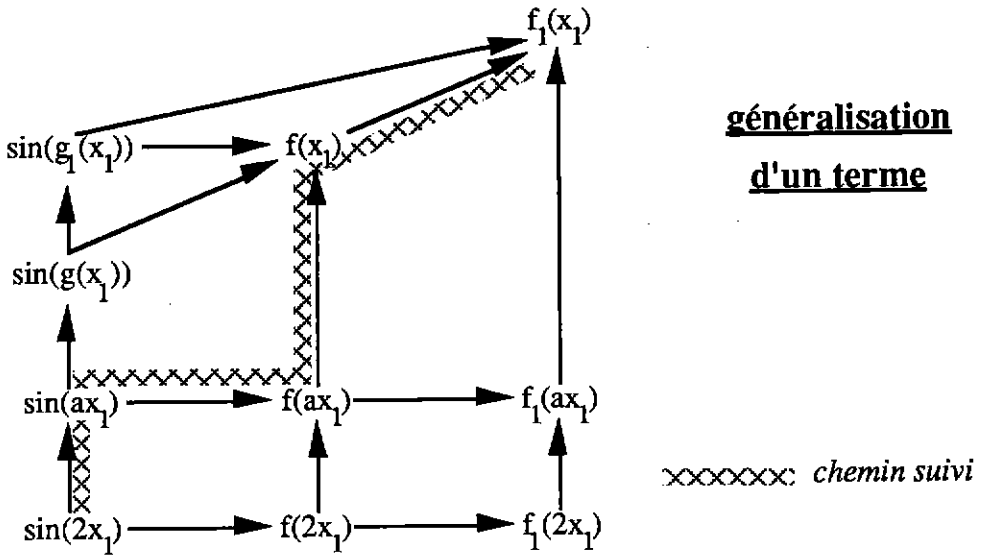


figure 3.19

La figure 3.19 présente les généralisations du terme "$\sin(2x_1)$" dans l'espace des termes. Ce graphe utilise des simplifications telles que "fog=f" ou "$f_1$og=fog$_1$=f_1". En considérant "x_1" comme intangible, les termes de la forme "$\sin(ax_1)$", avec "a" représentant n'importe quel terme constant, englobent comme cas particulier "$\sin(2x_1)$". De même, un terme fonctionnel constant "f" est plus général que "sin", mais moins général qu'un terme fonctionnel "f_1" dépendant de l'indice.

Chaque sous-terme peut être généralisé séparément ; aussi, une stratégie globale s'impose pour un **choix d'un chemin** parmi ces généralisations. La généralisation d'une formule se fait d'abord à partir de ses termes les plus profonds. Elle s'attaque au terme contextuel immédiat lorsque le niveau de généralisation atteint pour le sous-terme le dépasserait. Par exemple, généraliser "ax_1" en "$g(x_1)$" n'est pas choisi car cette généralisation dépasse le niveau de "sin". Cela revient à généraliser délibérement un sous-terme, en conservant son contexte fixé. Ultérieurement, le chemin adopte une voie médiane. La généralisation de "$f(ax_1)$" en "$f(g(x_1))$" est adoptée l'étape suivante, et simplifiée car "fog=f".

La généralisation n°3 d'une formule reste cohérente avec la transformation. Cela signifie que les manipulations mémorisées lors de la transformation sont prises en compte dans la génération : elles doivent rester applicables. Par exemple, l'indice "1" ayant été explicitement manipulé, il reste "gelé" lors des généralisations. Pour cela, la figure 16 n'introduit pas de "$f_i(x_i)$". De même, le chemin aboutirait à "$\sin(g_1(x_1))$" si le "sin" avait été gelé par son accès lors de la transformation. Ces considérations permettent de ne garder que les **éléments pertinents** à une transformation. Les généralisations n°1 et 3 tendent vers une même direction, puisqu'une généralisation trop importante de la formule nécessite une généralisation de la transformation, et que cette dernière n'a d'intérêt que pour une classe généralisée de formules.

La **sémantique** associée à une transformation produite par manipulation directe est particulière. C'est par exemple le mode récursif ou itératif de l'application d'une transformation. C'est aussi le rôle indexé ou non des opérations qui altèrent les parties de la formule de départ ou ses parties dupliquées. Le traitement de ces parties constitue le "cœur" de la transformation, et les opérations de construction de leur environnement le "décor" de la transformation. La sémantique particulière de ces parties originales ou dupliquées rappelle le cours de calculabilité de G. Huet, et sa gestion spécifique des redex lors d'une ß-réduction (il les marque afin de ne pas se perdre dans une branche infinie).

Cette sémantique spécifique permet de ne pas utiliser un langage de commande complexe pour exprimer une transformation. C'est au système de gérer la transformation effectuée (instance d'une version plus générale), ses généralisations (n°1) possibles, ainsi que les généralisations possibles du couple formule et transformation (n°1 et 3 cohérentes). Le choix des possibilités de généralisation de la formule comme celui d'une sémantique des transformations est arbitraire. Nous avons vu l'interprétation particulière des

sélections et des opérations de modification, et que l'ajout de composants extérieurs forme le "liant" de la transformation. Il reste le problème de la généralisation des fonctions d'accès.

Le problème des **fonctions d'accès** consiste à disposer d'un langage de commande suffisament riche pour mixer une requête d'accès par la structure et par les constituants. C'est un problème de routage à travers l'arbre d'une formule dont les composants ne sont pas forcément tous connus. Pourtant, il est naturel de vouloir sélectionner le plus petit terme fonctionnel le plus à gauche dans une formule, ou le terme de profondeur "n-1". La transformation de l'enchaînement 2 consistait par exemple à dupliquer le terme en "$\sin^2$" le plus à gauche, indépendamment de sa profondeur dans la formule.

2.5.3. Influence des manipulations sur les généralisations

Afin d'**anticiper les généralisations** n°1 ultérieures de la transformation (nécessaires ou facultatives), une menu d'options est automatiquement fourni au cours de chaque accès lors de la manipulation directe. L'accès par défaut y est spécifié (structurel ou du constituant), ainsi que la possibilité de raffiner l'intention de l'accès, dans la limite du langage d'expression de commandes sous-jacent. Que cherche-t-on à faire, en effet, lorsqu'il y a plusieurs occurences de variables indicées alors qu'il n'y en avait qu'une à transformer dans la formule de départ ? L'intérêt du raffinement est d'indiquer au système la sémantique de l'accès tout en bénéficiant des facilités de la manipulation directe. Cela permet par exemple, au lieu de désigner "$\sin^2 x_1$", de désigner le terme le plus à gauche de la forme "f^2" et ses arguments. Cette anticipation permet de disposer des généralisations n°1 et 3 désirées de façon automatique, en prenant comme base la structure "la plus générale", telle que les manipulations effectuées "aient un sens".

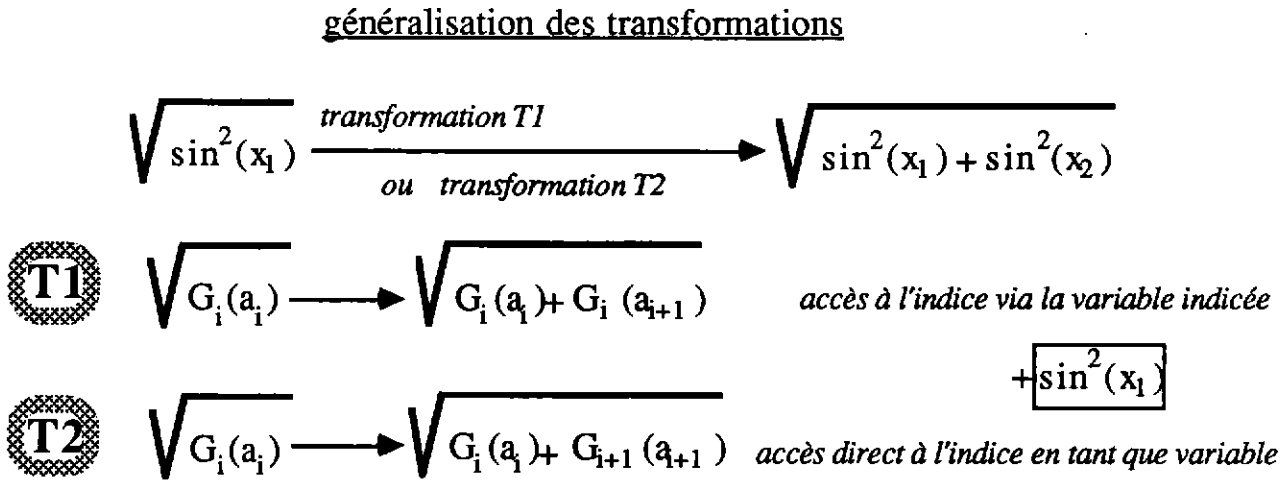


figure 3.20

L'exemple figure 3.20 présente une application de généralisation (n°1 et 3), orthogonale à celle aboutissant à la limite formelle de la formule (n°2). La généralisation concerne la transformation elle-même, vue comme une explication. L'exemple décrit l'influence d'une transformation sur sa généralisation, en reprenant la première transition de l'enchaînement 2.

L'étape critique concerne la facon d'accéder à l'indice pour le changer lorsque le deuxième terme est sélectionné. La sémantique associée sera préservée lors des généralisations. Si la sélection est structurelle comme sous-terme de "x_1", c'est-à-dire que l'indice est accédé en tant que position dans la structure "variable indicée", seuls les indices des variables indicées seront incrémentés. Cela donne la généralisation associée à la transformation T1. Si la sélection concerne la variable constituante, toutes ses occurences seront incrémentées par la transformation T2.

Ces généralisations peuvent être mémorisées et utilisées depuis une bibliothèque de transformations. Les conditions d'application d'une transformation y sont explicitées : savoir si l'appariement est syntaxique, mathématique, si les constituants ont des propriétés à respecter, etc. Les applications des généralisées des deux transformations donnent, dans le cas de la première étape :

Pour $G_i(a_i)$ instancié à $f_k(x_k) = k\, x_k^{k-1}$ on obtient :

T1 $\sqrt{k\, x_k^{k-1} + k\, x_{k+1}^{k-1}}$ T2 $\sqrt{k\, x_k^{k-1} + (k+1)\, x_{k+1}^{k}}$

Cette application concrétise les différences entre les deux transformations T1 et T2, non perceptibles sur un exemple trop réduit comme celui choisi lors de l'enchaînement 2. Les conditions d'application sont communs et portent sur la structure mathématique : il s'agit de trouver une fonction applicable sur une suite de variables et faisant intervenir leur rang dans la suite.

Cette transition sur la généralisation des transformations illustre une fois encore la **puissance** due à la connaissance de la transformation. La **richesse** d'utilisation des trois structures permet d'exprimer des intentions très proches. Les différences apparaissent lorsque la généralisation est importante.

Contrairement à la limite formelle, un choix est ici effectué parmi les généralisations possibles. Dans l'exemple, la généralisation choisie peut comporter une fonction "G" indépendante de l'indice, et non une fonction "G_i". Une présentation et un choix interactifs de généralisations ordonnées par niveau croissant sont requis.

Bien entendu, les **fonctionnalités d'accès et de généralisation des termes** peuvent être utilisés à des fins plus classiques pour décrire et localiser telle abstraction ("pattern") sur les termes en vue d'une recherche, tri, remplacement, calcul, caractérisation, etc.

2.5.4. Utilisation pratique des manipulations généralisées

Ce second enchaînement a le mérite d'illustrer que les possibilités des manipulations de formules sont aussi fécondes que les manipulations de preuves. Il montre que, similairement aux preuves, les manipulations de formules requièrent des fonctionnalités nombreuses, sont d'une complexité informatique importante et employent des techniques voisines de celles requises pour les manipulations ergonomiques de preuves. Contrairement à un a priori superficiel, les manipulations de formules sont d'un intérêt prioritaire dans l'exploitation d'un système de manipulation de preuves, car elles conditionnent l'appréciation subjective d'un mathématicien vis à vis d'un système.

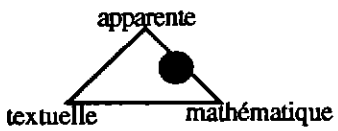
Ce second enchaînement propose par ailleurs une piste innovante pour des manipulations. De plus, son intérêt est de proposer des manipulations en vue de **faciliter la construction** d'autres manipulations selon une approche Génie Logiciel. Cela est rendu possible grâce au support des trois structures couplées, et des développements qu'ils permettent. Ce fut exploité dans l'enchaînement 1 pour présenter des opérations de transformation plus puissantes, puis dans l'enchaînement 2 afin d'**automatiser des manipulations.**

SOFTMATH cherche à répondre à la tendance exprimée en Génie Logiciel ou en Conception Assistée par Ordinateur, d'assister le cycle de développement d'un produit. Dans ce cadre, nous étudions l'apport pour les manipulations de formules, des connaissances concernant l'utilisation de formules et leurs constituants mathématiques. La connaissance des procédés de construction de formules permet d'exprimer des macro-opérations intégrant manipulations et calculs, au lieu des manipulations textuelles traditionnelles. Le tout repose sur les capacités de visualisation et de manipulation interactive, et crée une **synergie d'assistance** par une interface à **"engagement direct"** de l'usager.

Les manipulations mathématiques proposées sont sujettes à un **seuil critique** d'expressivité et d'ergonomie en dessous duquel un traitement de texte ou un traitement manuel sont préférables. La taille moyenne des formules et leurs corrélations interviennent dans le niveau de ce seuil. Plus deux expressions sont dissemblables, c.-à-d. éloignées en terme de topologie des transformations applicables, et moins il est rentable de construire la seconde en utilisant la première. Toutefois, un langage d'expression puissant permet de rapprocher leur distance.

Dans le cas d'une **expressivité** trop faible, nombre de manipulations ne peuvent être exprimées ou doivent être abusivement décomposées. Lorsque l'**ergonomie** chute, la facilité de mise en œuvre devient problématique et décourage les plus hardis. Dans les deux cas, le **rendement** du système est si faible que l'usager élimine toute prétendue assistance informatique.

Le cas de ce deuxième enchaînement est caractéristique. A quoi servirait en effet d'employer un mécanisme d'apprentissage lourd à mettre en œuvre pour l'assistance aux manipulations courantes. Deux critères peuvent relever le seuil critique : la rentabilité et la potentialité. La **rentabilité** dépend des applications répétées de la phase de mémorisation (avec 12 termes, le mathématicien gagne un facteur 10). La **potentialité** escomptée est ici la possibilité d'utiliser un mécanisme de généralisation doté d'une phase d'apprentissage obligatoire. Elle peut être aussi d'utiliser un système de calcul symbolique pour ce que le mathématicien ne pourrait pas ou ne voudrait pas faire manuellement.



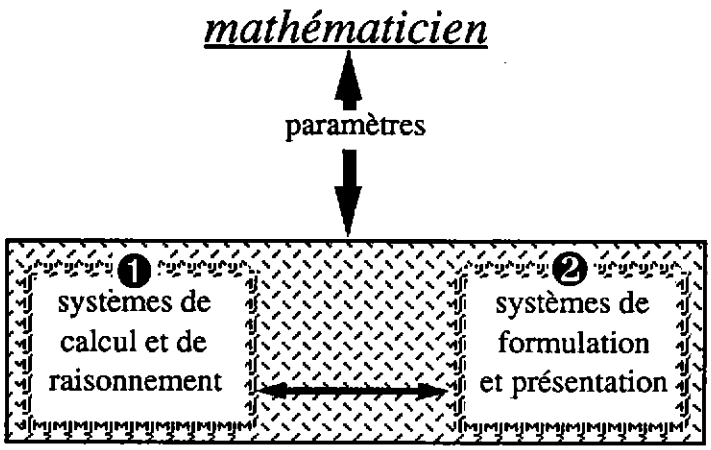
Nous avons donc présenté, dès l'introduction des trois structures, des pistes innovantes permettant un engagement direct entre le structure apparente et la structure mathématique, afin de relier la perception visuelle aux représentations mentales du mathématicien. Cela nécessite évidemment des connaissances et des mécanismes spécifiques pour permettre de telles manipulations "naturelles"[2].

[2] Nous traiterons les deux autres cotés du triangle chapitre 3 et 4.

3. MODELES INFORMATIQUES POUR UNE MISE EN ŒUVRE ERGONOMIQUE

3.1. LES FONCTIONNALITES D'UN ATELIER MATHEMATIQUE INTEGRE

3.1.1. Notre proposition d'intégration du cycle de conception



L'état des lieux a permis de **décomposer les systèmes informatiques** existants selon leurs fonctionnalités principales (rappel figure 3.1). Nous avons proposé d'intégrer ces fonctionnalités afin de factoriser les points communs de l'activité mathématique. Un tel Atelier Mathématique Intégré devrait disposer d'un minimum opérationnel : les connaissances mathématiques et les moyens de traitement du langage et de son contenu mathématique.

Parallèlement, nous avons vu que la plupart des systèmes étaient destinés à la production d'un résultat, et non destinés à faciliter son obtention. Notre proposition nécessite aussi de reconsidérer le paradigme d'automatisation en faveur d'un paradigme d'assistance au mathématicien. L'automatisation est dès lors réservée aux tâches élémentaires mécanisables, les aspects créatifs et l'essentiel du guidage de l'activité étant réservés au mathématicien. Cette problématique est plus généralement celle de l'**Assistance au Travail Scientifique**.

Dans la mesure où le mathématicien contrôle l'essentiel de l'activité, le guidage de cette activité devient un problème essentiel. Le suivi de ce guidage par un assistant Ami nécessite des connaissances nombreuses. Les systèmes actuels ne s'y intéressant pas directement, les connaissances nécessaires et les moyens de les obtenir restent à définir. En particulier, il manque les observables et les connaissances pragmatiques permettant au système d'anticiper et d'interpréter les intentions du mathématicien dans la situation en cours.

Les tâches à la charge du système sont délimitées à des **tâches élémentaires d'assistance ou de production**. Certaines tâches de production incluent l'utilisation d'heuristiques de résolution ou de schémas de démonstration du mathématicien. Il semble ainsi raisonnable d'espérer que le mathématicien puisse guider son activité par une introduction de paramètres suffisamment modérée. La contrepartie d'un tel système ouvert est alors de fournir une interaction particulièrement adaptée à la gestion des tâches élémentaires à mettre en œuvre.

Cela nous conduit figure 3.21 à introduire une troisième catégorie de systèmes nécessaires à une assistance étendue. L'introduction de **connaissances** permet en effet de faire le lien entre les résultats fournis par les systèmes de calcul et de raisonnement, et leur utilisation pour une démonstration. Le suivi de la résolution du problème permet d'assister conjointement les trois phases de la résolution interactive d'un problème :

- la phase de formulation est facilitée par la reconnaissance et la réalisation aisée des intentions de l'usager ;
- la phase de recherche est assistée par la construction de morceaux de preuve et la vérification de leur cohérence ;
- la phase de présentation bénéficie de la connaissance des modes de présentation pour faciliter le dialogue et les explications.

Ces connaissances sont essentielles pour rendre compte du cycle de production global de l'activité mathématique :

- définition du problème ;
- construction de la preuve ;
- présentation de la preuve et diffusion de la démonstration.

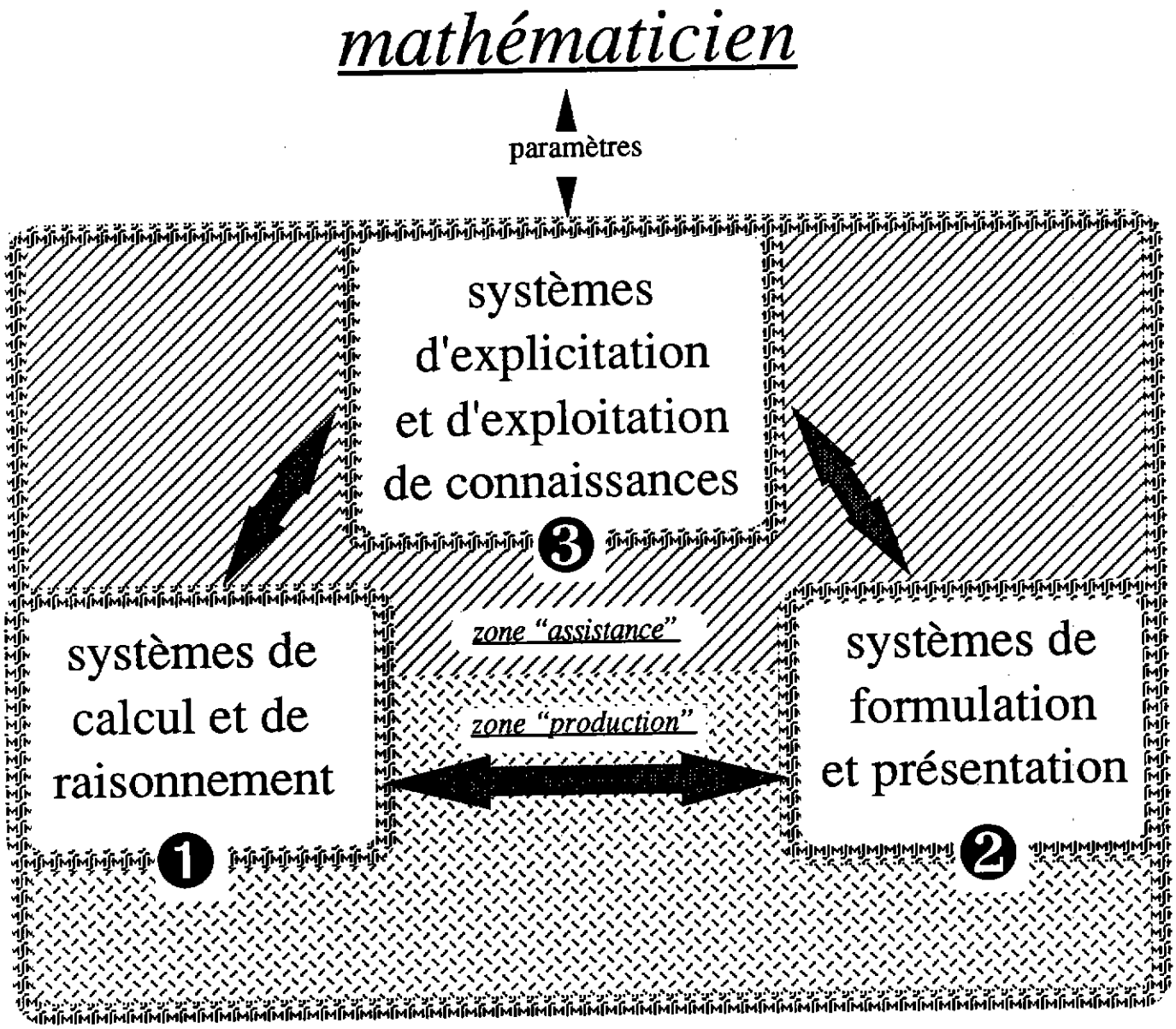


figure 3.21

Pour récapituler, notre approche de Mathématiques Assistées par Ordinateur est complémentaire de celle des systèmes existants. Elle cherche à intégrer les systèmes de calcul et de raisonnement, avec les systèmes de formulation et de présentation, en vue d'une assistance au mathématicien. Elle introduit la connaissance du langage mathématique comme lien permettant de diminuer le nombre de paramètres à fournir.

Les mots clés de l'approche choisie sont **assistance, structuration, interactivité, incrémentalité, apprentissage, langage, connaissances et raisonnement**. Elle vise à :

- transférer le cycle de conception du mathématicien vers la machine grâce à un AMI ;
- laisser la maîtrise du guidage des opérations au mathématicien ;
- assister la création d'une preuve et la production d'une démonstration associée ;
- utiliser des connaissances explicitées de la pratique du mathématicien, afin de l'assister dans les trois phases de son cycle interactif de résolution d'un problème.

3.1.2. Notre approche Intercom vers un AMI

Un Atelier Mathématique Intégré s'articule autour d'un environnement de compréhension et de manipulation d'expressions mathématiques, couplé à une base de connaissances mathématiques. Cet environnement est enrichi par des fonctions de manipulations numériques ou symboliques, dirigées, selon la philosophie adoptée dans CAMELIA [Vivet 84], par un module de raisonnement.

Cela permet d'automatiser des déductions simples, à un bon niveau d'abstraction. Les parties calcul et raisonnement, associées aux connaissances mathématiques, intégrent donc certains aspects du savoir-faire du mathématicien. Les prémisses d'une démonstration automatique "naturelle" [Bledsoe 86], sont alors réunies. Tout cela est utilisé par un Assistant mathématicien intelligent.

Les fonctionnalités d'un Atelier Mathématique Intégré peuvent se décomposer selon les deux axes de la figure 3.22 :

- L'**axe scientifique** relie les capacités de calcul et de raisonnement aux connaissances issues de l'expertise linguistique et mathématique.
- L'**axe informatique** exploite les principes d'interaction présentés en partie II, avec un environnement de simulation permettant les manipulations mathématiques et un assistant chargé de la communication avec le mathématicien.

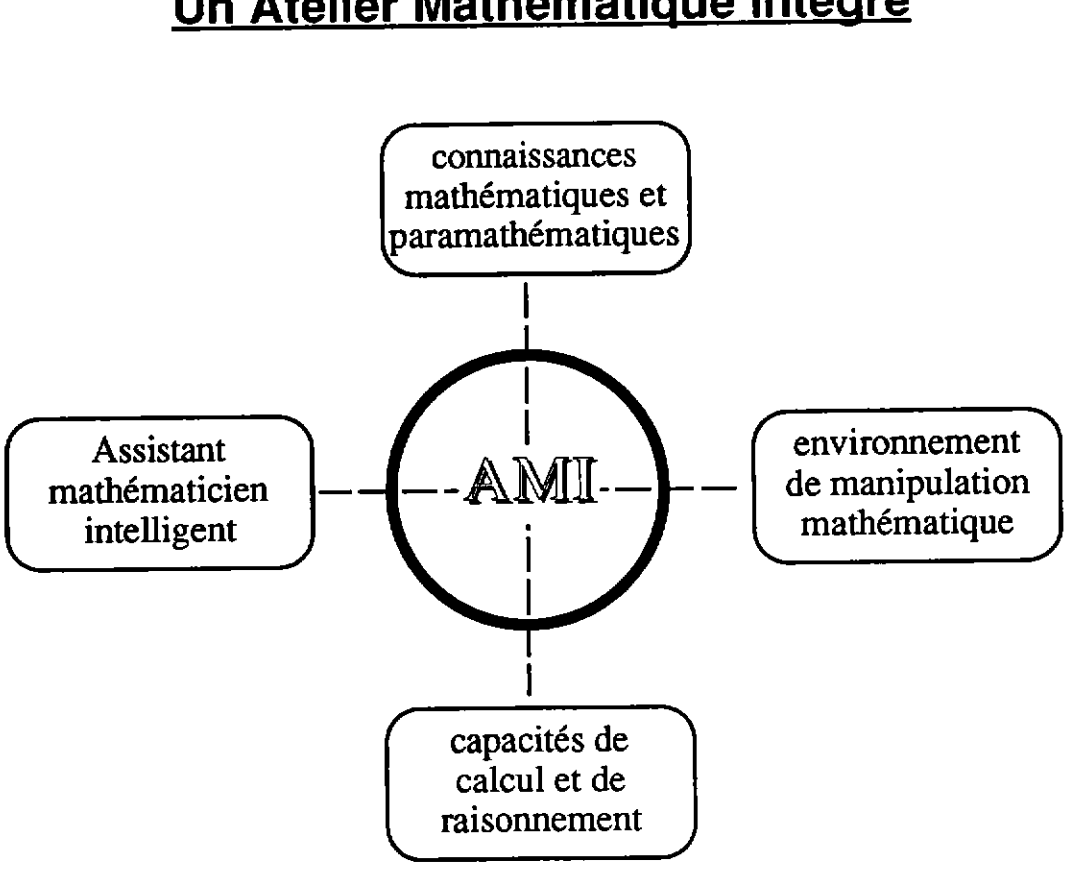
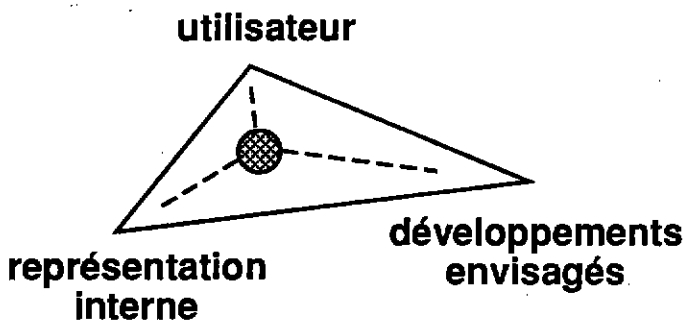


figure 3.22

Conformément à notre approche Intercom, nous proposons un développement expérimental ascendant basé sur la réalisation d'un environnement de manipulations de formules baptisé SOFTMATH. Il constituera, une fois réalisé, une plateforme d'extension vers un AMI.



Les objectifs ergonomiques - ou plutôt les impératifs ergonomiques - de SOFTMATH contribuent à en faire un projet ambitieux. Le développement de SOFTMATH doit donc aussi être planifié selon l'approche Intercom, en effectuant une redistribution des ressources au profit de l'interaction. Le noyau de développement, à l'usage des concepteurs de SOFTMATH, se répartit selon les trois axes ci dessus. La répartition des thèmes autour de ces axes figure le déroulement des chapitres successifs :

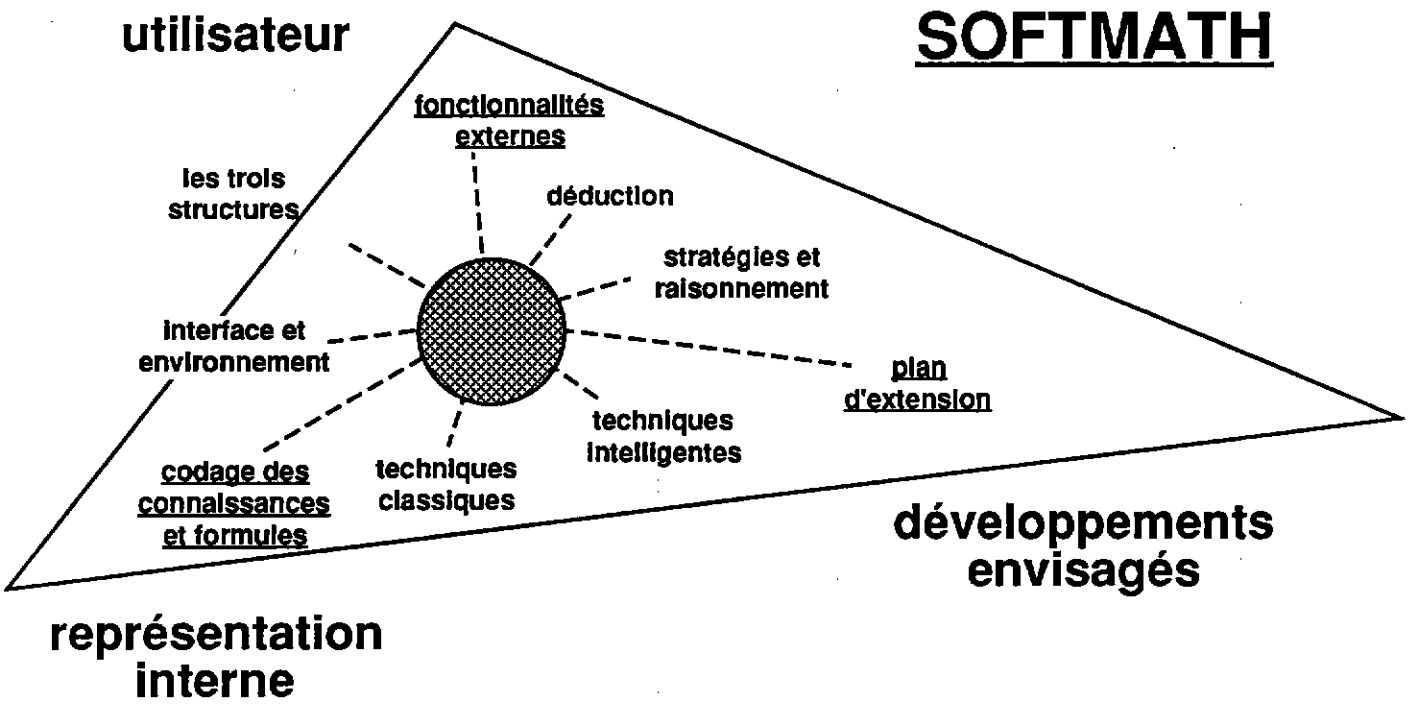


figure 3.23

Le coté gauche correspond à la **partie descriptive** de SOFTMATH telle que nous l'avons présentée dans le chapitre 2. Nous abordons ici (chapitre 3) le modèle informatique. Les chapitres 4 et 5 traitent de l'hypoténuse de la figure 3.23 et présentent respectivement le traitement des formules et les connaissances nécessaires.

Le coté supérieur du triangle traite de l'aspect raisonnement qui n'est envisagé, chapitre 6, que pour faciliter l'utilisation de formules qualifiées d'identités remarquables. Un plan d'extension parachève la **démarche de SOFTMATH** vers un AMI. Il comprendra la construction et la manipulation de preuves, ainsi que des études expérimentales pour l'extension et la validation de modules supplémentaires.

3.1.3. L'architecture structurelle d'un AMI

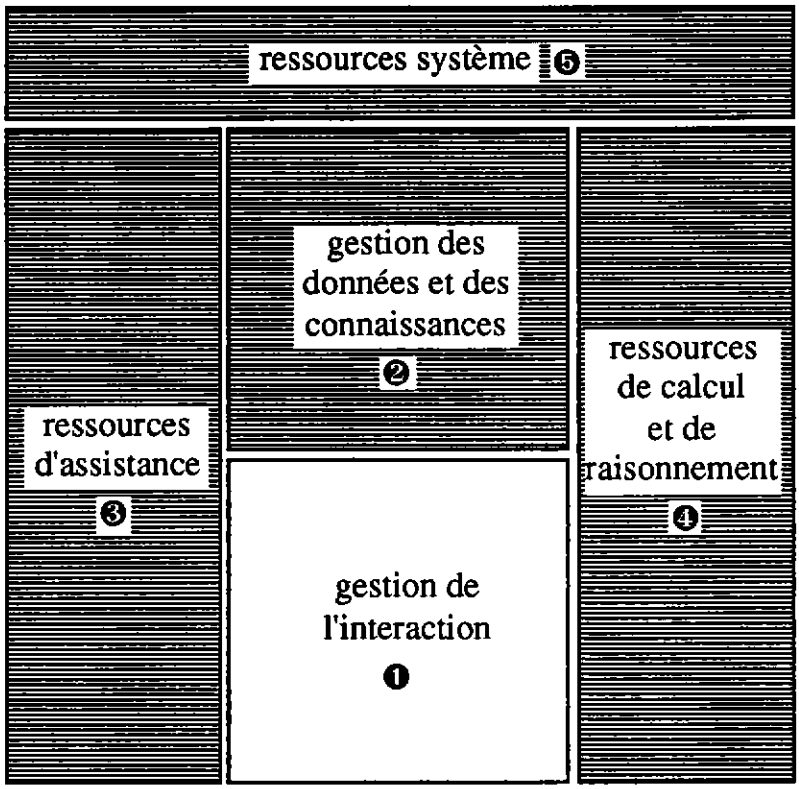


figure 3.24

Afin de réaliser l'intégration de fonctionnalités très variées en un système informatique partageant des ressources communes, nous proposons une *organisation structurelle* d'un AMI selon la figure 3.24. Nous qualifions cette organisation de structurelle car elle correspond aux contraintes informatiques générales, par opposition à une *organisation fonctionnelle* qui correspondrait aux modules dédiés à une application mathématique comme l'EIAO. Ces modules seraient obtenus par une configuration particulière du système informatique.

Nous nous intéressons particulièrement dans ce chapitre au module distingué en blanc. La gestion de l'interaction ❶ regroupe deux thèmes :

- l'interface, avec son paradigme dominant (fenêtres, "rooms", monde virtuel, etc.) et la gestion de ses abstractions spécifiques (objet graphique, son digitalisé, etc.) ;
- l'analyse et la génération syntaxique des langages (de commande, de programmation, Langage Naturel, etc.).

Ce module autorise globalement deux types de tâches, selon qu'il y a constitution du modèle de l'activité (introduction de définitions, acquisition de connaissances, etc.), ou utilisation du modèle lors de la construction de preuve.

La gestion des données et des connaissances ❷ se subdivise selon le choix de l'usager. Par exemple, certaines données achevées telle une formule ou une preuve sont qualifiées de connaissances. Les connaissances se prêtent à un usage plus général qu'une donnée, ce qui introduit une distinction quantitative et qualitative. Cette distinction rend leur acquisition intéressante, tandis que les données sont des constructions d'intérêt spécifique a priori. Données et connaissances sont informatiquement passives en ce sens qu'elles doivent être manipulées par des processus.

Les ressources d'assistance sont scindées ❸ en assistance au problème mathématique et assistance à l'utilisation du système. Elles comprennent des capacités de raisonnement spécifiques permettant, par exemple, la complétion d'une transformation de l'usager par un Ami (Assistant mathématicien intelligent). Elles permettent notamment la gestion sémantique du dialogue et des explications. Elles gèrent aussi par exemple le modèle des usagers et la documentation.

Rappelons que le **facteur principal d'assistance** provient de la modélisation détaillée de l'activité mathématique. Comme nous l'avons vu partie II, disposer de l'opération de changement de coordonnées est plus utile que de ne disposer que du changement de variable. L'assistance tient compte par exemple des connaissances sur le changement de coordonnées pour demander les paramètres manquants. Elle peut être spécifique à cette opération ou utiliser des mécanismes génériques. L'assistance dépend donc, et exploite, les connaissances disponibles.

Les ressources de calcul et de raisonnement ❹ comprennent des ressources classiques comme le calcul symbolique ou la démonstration automatique, mais comprennent aussi l'inférence dans des hiérarchies conceptuelles, la perception, l'apprentissage et l'analogie utilisées en mathématiques. Il faut en effet prévoir des mécanismes susceptibles "d'assimiler et d'accommoder" de grandes quantités de connaissances très diverses.

Les ressources systèmes ❺ sont composées d'utilitaires généraux dont l'exploitation par les modules structurels n'apparaît pas à l'usager ordinaire (la manipulation d'arbre, un moteur d'inférence, un interpréteur de langage à objet, un gestionnaire de fichiers, etc.). En particulier, la multiplicité des langages informatiques manipulés rend intéressante la présence d'un utilitaire générique de définition d'un langage à partir d'une grammaire.

Nous ne détaillerons dorénavant que quelques points spécifiques aux mathématiques. Les techniques informatiques nécessaires à la constitution d'un tel Atelier Mathématique Intégré s'inspirent de l'ingénierie des systèmes (notamment celle des environnement de programmation) et de l'**ingénierie cognitive** qui est spécialisée dans les systèmes à base de connaissances. L'évolution rapide de ces techniques ne peut que favoriser la réalisation d'un AMI.

3.1.4. Utilisateurs et conception des interfaces

Les interfaces proposées varient selon le type d'utilisateurs :

- L'*usager* mathématicien n'y voit qu'un logiciel pour effectuer des manipulations mathématiques. Il peut particulariser son environnement et introduire des connaissances mathématiques ou paramathématiques simples à définir.
- L'usager-*auteur* (ou le gestionnaire s'il existe) peut de plus inspecter des structures informatiques pour les modifier ou enrichir les connaissances à ce niveau. Ils bénéficient des outils de présentation et de dialogue en Langage Naturel basés sur ces structures informatiques.
- Le *développeur* du système l'utilise avant tout pour étendre et tester les mécanismes informatiques implantés.
- Il y a enfin des interfaces logicielles. Un *module informatique* communiquant se connecte à un certain niveau du système pour y échanger de l'information.

Nous nous intéressons ici uniquement à **l'interface destinée aux usagers.** Des ouvrages récents présentent une synthèse claire de leur conception globale [Barthet 88, Coutaz 90]. Les domaines scientifiques intervenant comme guide de conception sont :

- La *psychologie cognitive* pour l'analyse de l'activité de l'usager hors système et en présence du système.
- L'*ergonomie cognitive* qui cherche à améliorer l'interaction des diverses classes d'usager avec un système. L'ergonomie des interfaces applique ces principes à la réalisation d'interfaces *adaptées*.

L'adaptation éventuelle du système dépend des intervenants :

- Le domaine, c'est-à-dire une classe de tâches à effectuer qui se raffine selon les tâches en cours.
- "L'usager", qui se raffine en fonction d'une classe d'usagers puis d'un usager particulier.

Lorsqu'un système s'adapte finement à un domaine il y a deux possibilités :

- le domaine est réduit et il devient suffisamment spécifique et intéressant pour justifier un système spécialisé. C'est par exemple le cas d'un système expert en transformations de Fourier.
- le domaine reste suffisamment homogène pour que des choix génériques puissent être effectués et l'adaptation du système consiste à effectuer des choix supplémentaires parmi les possibilités offertes. C'est ce que nous proposons pour les tâches élémentaires en mathématiques.

Nous avons de plus, dès notre trièdre figure 1.1, proposé de scinder le domaine en langage et données. C'est-à-dire, nous avons dissocié les traitements à composante essentiellement langagière, des traitement sur les données concernant par exemple les calculs, l'utilisation des connaissances et la construction de preuves.

La difficulté majeure d'un *modèle de l'usager* est qu'il se définit comme un paramétrage de l'activité. Les actions du système peuvent notamment être adaptées à l'usager lorsque le système dispose d'une représentation détaillée des tâches élémentaires et des choix à prendre en compte. L'usager peut se définir conjointement selon plusieurs paramètres. Il y a notamment des paramètres :

- sociaux : élèves, étudiants, scientifiques, industriels, etc.
- de compétence dans le domaine ;
- d'expérience vis à vis du système : un *usager occasionnel* se contente d'un jeu de fonctionnalités réduits pour mener à bien son application, tandis qu'un *usager averti* connaît les moyens d'exploiter plus complètement et plus efficacement le système. Une interface ergonomique permet une amélioration rapide de l'expérience de l'usager vis à vis du système.

Les fonctionnalités de l'interface usager ont été décrites par Norbert Kajler pour les systèmes de calcul algébrique formel [Kajler 90]. L'approche proposée par N. Kajler est innovante et particulièrement intéressante. Elle met en avant les point suivants :

① spécification à partir des besoins ;
② méthodologie privilégiant les technologies nouvelles ;
③ interfaces adaptées ;
④ fonctionalités adaptées.

Nous avons maintenant les éléments permettant de présenter point à point l'approche de N. Kajler. L'approche de N. Kajler est orientée vers la réalisation prochaine d'une interface de système de calcul formel, tandis que la notre est prospective afin de modéliser l'activité mathématique. Cette comparaison permet donc de positionner notre approche par rapport aux réalisations prochaines d'interface.

① spécification à partir des besoins :

Dans l'approche de N. Kajler, la spécification de l'interface est effectuée en fonction des besoins des usagers des systèmes de calcul formel.

Nous avons cherché à définir une interface adaptée aux besoins des mathématiciens selon l'approche Intercom. Notre étude du Langage Mathématique, nos propositions d'assistance à la construction de preuves et nos propositions de manipulation de formules constituent une spécification générale.

② méthodologie privilégiant les technologies nouvelles :

N. Kajler parle d'une :

> « méthodologie nouvelle, utilisant les outils logiciels les plus performants du moment pour générer automatiquement certains éléments de l'interface utilisateur. »

Ces outils étant d'ailleurs en cours de développement, il insiste sur le fait que :

> « les besoins relatifs aux interfaces des systèmes de calcul formel sont suffisamment complexes pour intéresser les concepteurs de ces outils et participer à leur mise au point. »

Nous partageons cette méthodologie ouverte permettant de bénéficier des progrès techniques. Nous proposons de l'appliquer pour un AMI tout entier et d'instaurer une problématique interdisciplinaire. De plus, les besoins des mathématiques sont suffisamment exemplaires pour intéresser les concepteurs de systèmes cognitifs et d'outils de manipulation structurée.

③ interfaces adaptées :

Se démarquant des interfaces figées actuelles, l'approche de N. Kajler propose des interfaces adaptées à une classe d'usager et à un domaine spécifique (mécanique quantique, mathématiques appliquées aux modèles économiques, automatique, etc.).

Nous proposons que les interfaces soient aussi adaptées en fonction de la tâche en cours. Ces interfaces dont l'adaptation s'effectue sous contrôle du système sont qualifiées d'*interfaces adaptatives*. Cela permet par exemple de prendre en compte :

- le type de problème (calcul de surfaces avec l'énoncé, la surface, la courbe, son étude...) ;
- le type de raisonnement (présentation particulière pour le raisonnement par récurence) ;
- les tâches de recherche de conjecture valide, où le choix et la visualisation d'exemples et de contre-exemples est déterminant ;
- l'acquisition de connaissances, qui peut se limiter à l'introduction d'une définition manquante et de ses propriétés lors d'une session ;
- la construction graphique et la réutilisation de preuves ; etc.

④ fonctionalités adaptées :

N. Kajler note que les usagers souhaitent disposer d'opérateurs spécifiques à leur domaine, ou d'opérateurs nouveaux qui leurs sont propres.

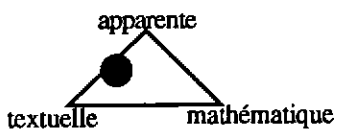
Nous postulons que cette adaptation du langage et des concepts du domaine est un facteur déterminant en mathématiques. Cela se traduit au niveau de l'interface par la définition des moyens de présentation et la prise en compte dans les options de dialogue. Ainsi, lorsqu'un nouvel opérateur mathématique comme le coefficient du binôme "$\binom{n}{p}$" est introduit, ses trois structures doivent être décrites et doivent figurer dans des menus.

Les fonctionnalités principales d'une interface de manipulation de formule, telle qu'elle est définie par N. Kajler, peuvent être regroupées en :

- manipulations générales de formules :
  telles que saisie, édition, visualisation, etc. ;
- manipulations de formules complexes :
  concerne les possibilités d'abréviation et de visualisation partielle ;
- manipulations globales au travail :
  permettent d'accéder aux objets mathématiques introduits dans l'environnement de travail ;
- utilisation de langages de dialogue :
  permet d'adapter et d'étendre les fonctionnalités mathématiques et les fonctionnalités propres au système ;
- outils d'assistance à l'usager.

3.2. GESTION DES MANIPULATIONS DE FORMULES

3.2.1. Les opérations du système sur les formules



La gestion de l'interaction que nous abordons ici concerne plutôt la structure apparente et la structure textuelle, qui sont les intervenants principaux de la saisie et de la présentation de formules par un système informatique.

Le **cycle de vie d'une formule** peut être résumé du point de vue du système en : analyser, représenter, manipuler, restituer, afficher.

Les diverses étapes qui interviennent dans la construction d'une formule sont alors :

- L'*analyse*, qui permet de constituer la représentation interne de la formule.
- La *représentation* interne de la formule, qui décrit la formule et utilise les informations connues sur les constituants de la formule.
- La *restitution*, scindée en deux parties :
  - La *décompilation* qui produit l'équivalent de la structure textuelle dans le langage d'entrée d'un module de formatage.
  - La *composition*, qui régit un ensemble de contraintes, et calcule les caractéristiques textuelles. Elle correspond au travail de restitution d'un logiciel comme EDIMATH.
- L'*affichage*, qui traduit finalement cette structure en bits visualisables sur un écran.

Une fois sa représentation interne constituée, une formule se prête à diverses *manipulations*, parfois dirigées par le mathématicien :

- La ***désignation*** d'un de ses constituants, à partir d'une des structures de la formule, est une opération primitive indispensable.
- L'*édition* d'une formule, ou d'une partie désignée, permet de la modifier, avec possibilité de préserver son sens.
- L'***assemblage*** d'une formule effectue les choix syntaxiques nécessaires pour bâtir la structure textuelle couplée à la structure mathématique. Il utilise des informations pragmatiques concernant la syntaxe du langage. Ce point est **original** et provient de notre volonté de traitement conjoint des informations syntaxiques et purement mathématiques.
- La *combinaison* de formules automatise des opérations d'édition ou de calcul sur une ou plusieurs formules. Quelques primitives mathématiques sont la substitution, le changement de variable, ou la composition d'expressions fonctionnelles.

Les **trois structures couplées** du chapitre précédent sont des abstractions qui permettent d'exprimer les informations concernant les formules. Nous abordons maintenant leurs représentations informatiques, ainsi que les aspects techniques concernant leur traitement.

La figure 3.25 suivante organise les trois structures couplées en fonction des supports de Représentation (noté "R.") de l'information. Chacune des trois structures d'une formule est associée à une représentation privilégiée. Le nom des opérations effectuées sur ces représentations est situé au dessus de chaque représentation ou transition. Ces opérations sont ordonnées en vue de la génération de la forme bidimensionnelle d'une formule.

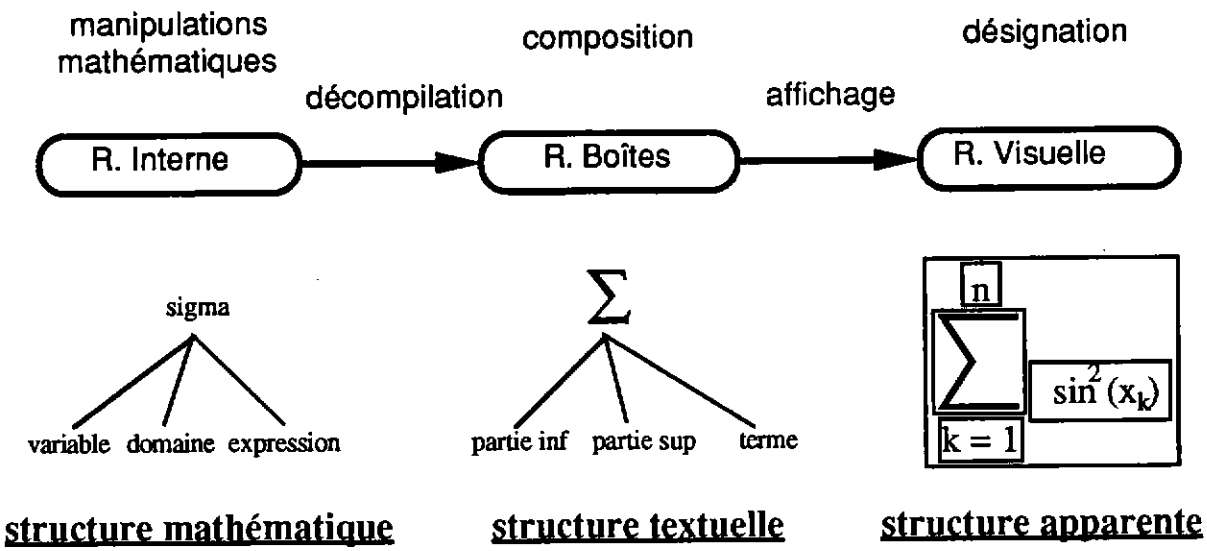


figure 3.25

La première structure utile pour des formules est la **structure textuelle**. C'est la structure utilisée par l'éditeur interactif EDIMATH [Quint 83], en vue de la production de documents mathématiques. Cette architecture permet d'automatiser la mise en forme des formules par la connaissance des opérateurs textuels. Ces opérateurs sont modélisés par le concept de *boîte* introduit par [Knuth 79] qui assure la transition entre la structure textuelle et sa composition bidimensionnelle.

La figure 3.25 représente un nœud de la hiérarchie des boîtes qui forme la *Représentation de Boîtes* de la formule. Lors de l'opération de *composition*, un "sigma" calcule la position graphique de ses opérandes et adapte sa taille à celle de ses bornes. Par ailleurs dans un terme indicé, l'indice est positionné et écrit avec une police de caractères plus petite.

L'*affichage* visualise ces calculs et mémorise les coordonnées, afin d'afficher la trace des boîtes en vue d'une *désignation* possible. La *Représentation Visuelle* correspond à la trace "bit-map" de la formule et autres pointeurs affichés. Elle constitue le support visuel de la *Représentation de l'Usager* de SOFTMATH.

Les notions utilisées par EDIMATH ne suffisent pas pour un usage mathématique. Par exemple, la chaîne de caractères :

"2f(t) dt + πr"

peut former une seule boîte dans la formule :

$$\int_0^1 2f(t)\,dt + \pi r^2$$

D'où le besoin d'une Représentation de Boîtes qui respecte la **structure mathématique**, dans un environnement d'assistance aux manipulations mathématiques. Une *Représentation Interne* spécifique à la représentation de l'**activité mathématique** est ainsi proposée (en comparaison, la "Représentation Interne" d'EDIMATH correspond approximativement à notre Représentation de Boîtes).

Pour SOFTMATH, les opérateurs de la structure mathématique dirigent la création de la partie correspondante de l'arbre de la structure textuelle en utilisant les informations d'assemblage : c'est la décompilation. Dans l'exemple de la structure mathématique figure 3.25 précédente, le nœud domaine est représenté par un intervalle défini par ses bornes. Il n'est pas directement associé à une boîte, mais les bornes "1" et "n" sont associées à deux boîtes dans la structure textuelle. Le nœud mathématique "sigma" supervise la décompilation de ses fils, afin de créer la boîte "partie inf" chez son homologue textuel : la boîte "Σ".

Une **représentation structurée** permet donc d'exprimer les concepts inhérents au niveau de structure choisi. Une fois ces concepts disponibles, les manipulations utilisant ces concepts peuvent être exprimées et automatisées. Le couplage des trois structures permet de gérer la cohérence des représentations. La traduction interne vers externe exposée permet l'interaction, car la désignation de la structure apparente permet d'établir le lien jusqu'à la structure mathématique. Toutes les opérations qui font appel à la nature mathématique des constituants, sont alors potentiellement automatisables.

Si l'édition peut s'appuyer sur les représentations présentées précédemment figure 3.25, il en va autrement de l'activité de saisie "ex nihilo". Il serait fastidieux et insuffisant de bâtir une Représentation de Boîtes comme intermédiaire vers la Représentation Interne. C'est pourquoi une syntaxe de commande spécifique permet de bénéficier de constituants mathématiques ou textuels déjà existants. Le produit généré est ainsi traité incrémentalement par un analyseur afin de produire la Représentation Interne. Les procédures de saisie doivent bien sûr être aussi "naturelles" que possible, et méritent une attention toute particulière.

3.2.2. La saisie des formules

Les capacités informatiques actuelles et futures favorisent une translation de l'effort de traduction de l'homme vers la machine, de façon à ce que l'usager puisse s'abstraire des détails d'implémentation et se concentrer sur son problème véritable.

Le transfert sur un support informatique bouleverse radicalement les aspects matériels du langage. C'est au prix de beaucoup d'efforts que les formules s'affichent comme le mathématicien les aurait écrites manuellement. Du point de vue lexical et syntaxique, le mathématicien a changé de langage : il manipule maintenant des notions telles que clavier, souris, menus, etc. Par exemple taper "s" "i" "n", cliquer sur le graphique [sin] ou sélectionner l'item "sinus" du menu de fonctions. Cette autre forme d'écriture s'apprend d'autant mieux qu'elle présente de similarités avec l'écriture papier.

Nous nous intéressons ici à la saisie de formules afin de les introduire, par opposition à l'édition qui modifie des formules existantes et à la visualisation qui est chargée de les afficher. Les principales difficultés de la saisie de formules proviennent de symboles et de positionnements très variés. Cela rend la saisie de formules fastidieuse et a donné lieu à des langages de commande difficiles à utiliser (par ex. celui du présent logiciel de rédaction).

La saisie de formules mathématique a alors donné lieu à des **logiciels spécialisés** qui construisent interactivement la forme graphique. Ces logiciels ont maintenant atteint un rendement optimal. Toutefois, ils se cantonnent à la présentation textuelle et ne peuvent accéder à la structure mathématique

des formules. Nous nous intéressons ici à une saisie des formules permettant de construire une structure mathématique compatible avec les possibilités de calcul et de raisonnement.

Nos objectifs sont :
- aboutir à la structure mathématique des formules en disposant aussi de leur structure textuelle et de leur structure apparente.
- obtenir que la construction des trois structures s'effectue facilement.

Cela peut être réalisé de deux façons :
- travailler sur la représentation interne d'un logiciel de traitement de texte afin de l'analyser pour produire la structure mathématique.
- envisager un logiciel qui intègre saisie et analyse mathématique.

La première façon a pour principal intérêt de bénéficier d'un logiciel existant. Elle est spécifique du logiciel de traitement de texte choisi et doit tenir compte de ses contraintes. Il est même plus difficile d'obtenir la structure mathématique si la structure textuelle est artificielle. Cette première façon n'est donc pas adaptée à notre approche. En revanche, les caractéristiques des saisies interactives de formules donnent des indications importantes pour la spécification d'un logiciel qui intègre saisie et analyse mathématique.

La première solution qui vient à l'esprit pour la saisie d'une structure arborescente consiste à saisir d'abord le nœud : le système qui connaît ce nœud n'a plus qu'à proposer de saisir un à un les fils dans l'ordre respectif.

En caricaturant les premiers logiciels ainsi conçus, pour saisir "a+x**2" on allait sélectionner "+" puis on tapait "a" à la place de son premier argument, "**" (il fallait que l'opérateur soit visible) puis "x" puis "2". On obtenait ainsi une fenêtre de commande avec "a+x**2" et une de visualisation, passive car elle n'admettait même pas d'entrée de commande souris, qui contenait "a+x^2". Même dans les versions plus évoluées, on ne pouvait sélectionner "+x^2" pour le copier ailleurs car ce n'est pas une structure complète.

La solution actuelle des logiciels de traitement de texte interactifs relève des mêmes principes mais ils travaillent sur la structure apparente. Les opérateurs sont visualisés et le remplissage peut s'effectuer aussi à la souris. La saisie de la structure arborescence est toujours aussi rigide, mais elle se fait "à distance" depuis la structure apparente. Par exemple la saisie de :

$$\frac{\sqrt{a_1^2+a_2^2}}{n!}$$

s'effectue en choisissant d'abord l'opérateur de fraction. Le curseur se positionne sur le numérateur qui est rempli en choisissant une racine. Le premier fils de la racine vaut ici 2 car c'est une racine carrée, etc. Les mathématiciens persévérants reconstitueront aisément la saisie complète...

Cette solution introduit des **lourdeurs de saisie** pour le mathématicien. Cela se traduit dans notre exemple précédent par l'obligation de préfixer les élévations au carré, l'addition, la fraction et la factorielle. Cette obligation est ici incompatible avec la saisie naturelle.

Cette rigidité de saisie infixée est d'autant plus inacceptable que le nombre d'opérateurs est plus important dans la structure mathématique que dans une structure textuelle d'un traitement de texte. Malgré la mise à disposition graphique de certaines de ces structures infixées, le temps de saisie est rallongé.

La solution pour permettre une saisie plus conviviale passe obligatoirement par l'exploitation d'un **mécanisme d'analyse** qui permette de produire la structure mathématique désirée à partir d'une séquence d'items lexicaux. Cette originalité a été notamment exploitée par les environnements de programmation structurés afin de permettre une saisie mixte, à partir de la structure arborescente ou à partir d'un langage de commande séquenciel à analyser. Le mathématicien dispose alors de plusieurs façons de procéder dont l'une au moins correspond à une saisie naturelle.

L'existence d'un mécanisme d'analyse peut être renforcé par des **connaissances pragmatiques** permettant une analyse plus robuste. Cela permet d'envisager un recours plus intensif à l'analyse afin d'autoriser des raccourcis de saisie informatique. Une étude lexicale approfondie des mathématiques est donc nécessaire pour déterminer un jeu de commandes simples et adaptées.

L'idéal serait une saisie manuelle aussi proche que possible du mode d'écriture usuel. Les discontinuités spatiales doivent néanmoins être explicitées par le mathématicien en plus des symboles de la formule.

Dans certaines circonstances, par ex. pour le Σ, elles peuvent être anticipées ; dans d'autres, tel un terme indicé, elles doivent être explicitées.

L'avantage d'un mécanisme d'analyse mathématique est que **la connaissance des objets mathématiques de la formule peut aider la saisie.** Par exemple, le fait de savoir que le premier "a" est un terme indicé de type "a_i" permet de retrouver que "1" est un indice dans la saisie de "a1". Il permet aussi de ne pas avoir à distinguer entre la saisie de "$a_1{}^2$" (le carré de "a_1") et celle de "a_1^2" (avec un indice double disposé l'un en bas, l'autre en haut). Certaines ambiguïtés peuvent ainsi être levées par cette connaissance pragmatique.

Les spécifications pour une saisie conviviale respectent ainsi quelques principes importants, qui peuvent être raffinés en fonction des possibilités et des expérimentations sur l'usage du système :

- L'ordre et la nature des *actions informatiques* lors d'une saisie doivent se rapprocher de la saisie papier-crayon.
- Le système fournit un *catalogue* des objets du contexte et des structures possibles, de leur saisie, de leur sens mathématique et de leurs variantes textuelles. Certaines formes visuelles sont directement disponibles à l'écran.
- La saisie est utilisée pour reconstituer la structure mathématique et la structure textuelle. Par exemple, la présence ou l'omission des bornes d'un Σ dans la saisie est mémorisé pour déterminer le style exhaustif ou allusif de la présentation textuelle du Σ.
- Certains "détails" typographiques tels les différences de taille sont exclus lors de la saisie. D'autres *similarités visuelles* comme celle entre l'indice supérieur et l'élévation à la puissance peuvent aussi être confondues lors de la saisie. Il peut ainsi y avoir une certaine forme d'*élision*.
- Le *parenthésage* sert, comme dans l'usage habituel, à déterminer les regroupements utiles à l'interprétation. L'élision du parenthésage inutile est important pour la lisibilité.
- La *juxtaposition* peut avoir un sens mathématique (multiplication scalaire). De plus, le caractère blanc est facultatif : quand il n'est pas omis il sert de séparateur. Systématiser son usage devient fastidieux, alors que l'omettre et laisser le système le rétablir pour une bonne lisibilité est agréable.

Ainsi les expressions "y=exp(x+1)", "z(t)=cosωt+isinωt" et "Im(z)=sinωt" doivent pouvoir être saisies telles quelles. Il n'est pas question de spécifier si "exp(x+1)" est le produit de "e" par "x" par la valeur de la fonction "p" au point "x+1", ou s'il s'agit de la valeur de la fonction exponentielle au point "x+1". Cela nécessite ici encore des connaissances pragmatiques intégrées dans le mécanisme d'analyse.

Une saisie conviviale est définie lors de la conception en fonction des objectifs du système. Ainsi, lors d'une expérience que nous avons encadrée pour construire un environnement de dérivation symbolique [Gatine 87], la convivialité de l'interface a été particulièrement soignée. La saisie des formules était déjà graphique et elle s'effectuait par des commandes simples qui exploitaient la finalité mathématique du système. Une session de saisie et notre commentaire sont présentés en Annexes.

La caractéristique principale de cette interface était la construction incrémentale des formules avec un langage de commandes proche de la saisie textuelle, la construction de la structure mathématique (en tolérant qu'elle soit incomplète), et l'affichage simultané de la structure visuelle. Rappelons que les systèmes de calcul formel n'offraient qu'un langage peu lisible et que la saisie "batch" devait être totale avant d'être interprétée ou rejetée.

Les choix effectués pour cette saisie conviviale ne correspondent pas à notre objectif général de modélisation de preuves et démonstrations. Notre volonté de prendre en compte les libertés d'écriture des mathématiciens cherche à concilier la richesse mathématique et textuelle du langage. Il nous faut alors laisser à l'usager la possibilité de **définir son langage**. Un logiciel permettant cela autorise aussi la définition et la prise en compte de la saisie correspondante, en exploitant des connaissances pragmatiques proposant des saisies par défaut. Notre problème principal lors de la saisie est alors la mise en œuvre d'un tel paramétrage.

La prise en compte du langage pour les Phrases et Formules permet de préfigurer la saisie mathématique de l'avenir. Il s'agit de :

La saisie vocale des formules par le mathématicien

Cette voie de recherche est particulièrement prometteuse car la saisie vocale est indépendante des inévitables commandes de positionnement bidimensionnel. De plus, elle est très rapide et extrêmement conviviale. Par exemple, un énoncé vocal de la formule :

$$\frac{\sqrt{a_1^2+a_2^2}}{n!}$$ est : racine de a1 carré plus a2 carré le tout sur factorielle n

Cette saisie englobe les problèmes de la saisie papier-crayon du mathématicien. Notons dans cet exemple :

- La possibilité de dire "racine de" ou "racine carrée de", "a1" ou "a indice 1", "sur" ou "le tout sur" pour lever l'ambiguïté éventuelle.
- Le rôle du groupement sonore lors de la lecture entre le numérateur et le reste, analogue au rôle de l'espace blanc.
- Les procédés particuliers pour préciser le numérateur de la fraction continue, permettant par exemple de repréciser certains points (la somme étant divisée par, a2 étant divisé par, etc.). La saisie est alors un véritable dialogue avec points de rattrapage.
- L'utilisation fréquente de synonymes et de procédés de paraphrases comme "divisé par" au lieu de "sur".
- La possibilité de construire les formules de façon ascendante "a1 carré plus a2 carré, je prend la racine et je divise par factorielle n". La saisie s'apparente alors à de l'édition par action et transformation d'une formule existante.

Cette saisie met bien en avant la nécessité d'un traitement du langage. Même sans viser directement une saisie vocale, ces procédés langagiers peuvent être étudiés pour assouplir la saisie des formules.

3.2.3. Gestion des formules au sein de l'interface

La gestion des interfaces informatiques est souvent effectuée actuellement par un UIMS (User Interface Management System). L'idée de base des UIMS est de séparer l'interface usager des modules destinés à l'application visée (module de manipulation symbolique). Cette séparation informatique permet d'adapter l'UIMS pour des applications différentes. Pour ce qui nous concerne, il s'agit d'applications scientifiques utilisant des variantes différentes du Langage Mathématique.

Toutefois, un UIMS a ses propres limites et ne s'applique qu'à un certain type d'applications. Nous avons vu en particulier que la saisie des formules avait nécessité le développement de logiciels spécifiques. C'est pourquoi, nous proposons un environnement de type UIMS spécialisé aux mathématiques.

La spécificité d'un système mathématique est de disposer de connaissances, sur la construction des formules mais aussi sur l'activité mathématique. Il est donc envisageable de disposer de connaissances spécifiques à la gestion des formules et de l'interface. Cette forme "d'intelligence de l'interface" est par exemple proposée pour la programmation dans [Dang 88]. Nous nous intéressons ici spécifiquement à la gestion des formules au sein de l'interface.

Nous sommes maintenant en mesure de décrire le schéma de l'environnement SOFTMATH (figure 3.26 page suivante). Lors de la boucle d'élaboration des formules, le système est amené à manipuler des objets de nature différente. Les représentations utiles sont alors divisées en deux groupes (saisie et visualisation conceptuelle) :

- Représentation.....de l'Usager (RU)
- .....du Terminal (RT)
- .....Normée (RN)
- .....d'Entrée (RE)

- Représentation.....Interne (R I)
- .....de Boîtes (RB)
- .....Graphique (RG)
- .....Visuelle (RV)

Le premier groupe correspond à la saisie jusqu'à la Représentation d'Entrée des utilitaires construisant la Représentation Interne. Le second aboutit à la visualisation adaptée au "processeur" humain en vue de la constitution de sa représentation interne mentale. Vu leur symétrie, il n'est ainsi pas étonnant que la Représentation Interne et celle de l'Usager soient complexes à appréhender.

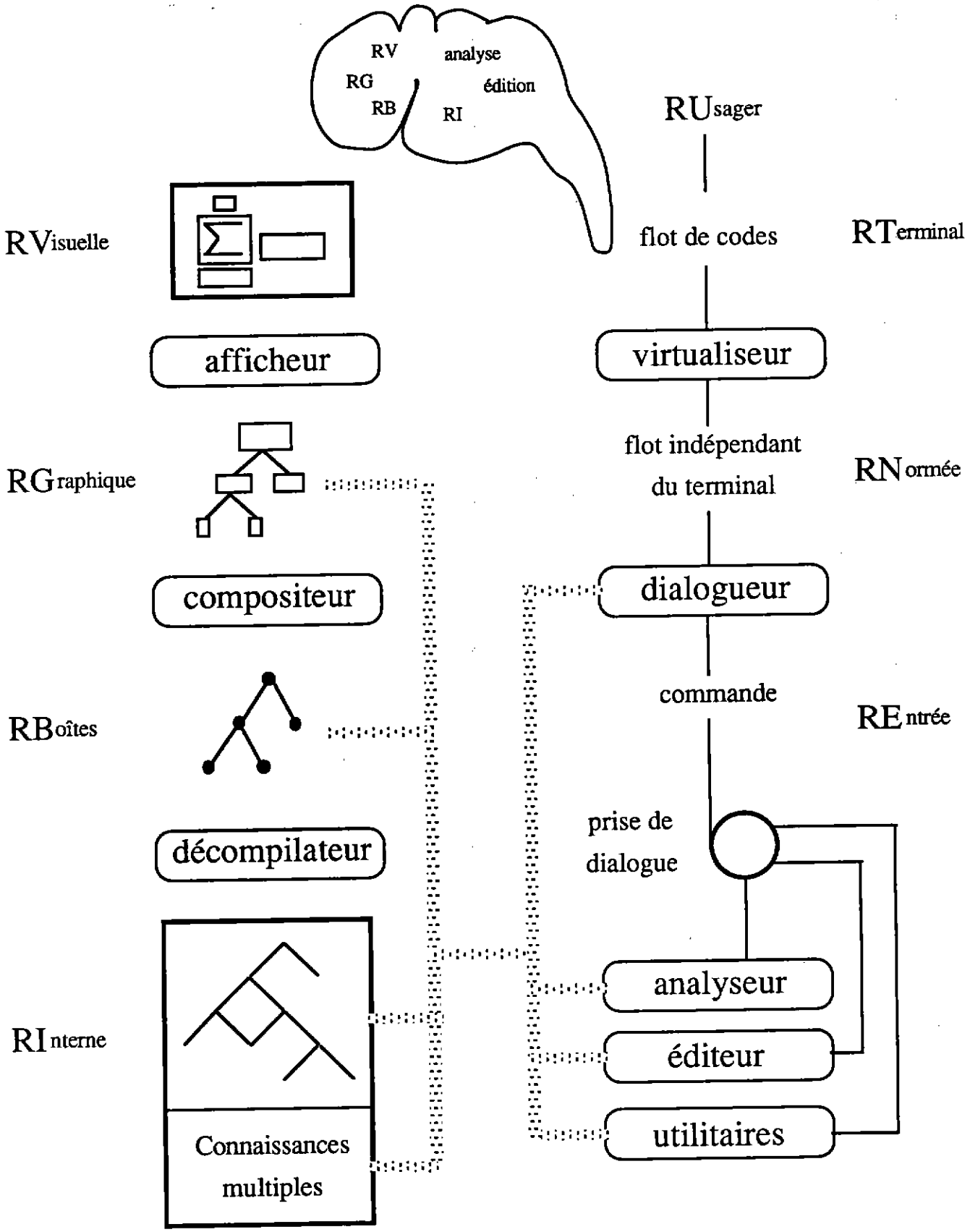


Schéma de l'environnement SOFTMATH

figure 3.26

3.2.4. Les modules de la gestion des formules

Le principe des manipulations structurées consiste à tout ramener à une Représentation Interne (RI). Toute action de l'usager est traduite en une action sur la Représentation Interne. C'est à partir de la Représentation Interne que la rétroaction est Visualisée (RV).

La difficulté consiste à coordonner ces deux flots et à proposer des concepts appropriés à la Représentation de l'Usager (RU). C'est la notion d'interface à **"engagement direct"** [Norman 87]. Plus précisément, les concepts utilisés par l'usager pour formuler sa tâche sont disponibles et sans biais ; d'autre part les objets sur lesquels des actions sont effectuées sont les mêmes que ceux qui matérialisent les résultats.

Le flot de codes du Terminal (RT) est filtré par le *virtualiseur* pour être exprimé dans une Norme indépendante du terminal (RN). Le *dialogueur* l'analyse et le transmet à l'outil concerné dans la Représentation d'Entrée (RE) lui correspondant. Une commande de la Représentation Externe spécifie l'action de l'outil sur la Représentation Interne. Ainsi le virtualiseur fournit les coordonnées de la pression du bouton de la souris. Le dialogueur les transforme en commande qu'il transmet à l'outil.

S'il y a lieu, la rétroaction (légèrement simplifiée lors de notre présentation préliminaire des trois structures figure 3.25) est déclenchée. Le *décompilateur* produit un arbre de Boîtes (RB). Le concept de **boîte** [Knuth 79] permet d'exprimer l'ensemble des Représentations Visuelles possibles indépendamment des terminaux et de la taille de la fenêtre. Une boîte "expression de termes additifs", d'arité variable, peut ainsi s'afficher sur une ligne, se couper entre deux arguments pour occuper plusieurs lignes, ou mettre chacun de ses arguments sur une ligne différente. Le *compositeur* choisit une possibilité en fonction de paramètres comme les dimensions allouées. Il calcule alors les positions respectives des boîtes instanciées. Cette Représentation Graphique (RG) est enfin traduite en Représentation Visuelle par l'*afficheur*.

Ce second groupe de représentations (RV, RG, RB et RI) sert de support à la Représentation de l'Usager. Leur correspondance doit donc être assurée dans les deux sens. En particulier une **désignation** en Représentation Visuelle peut permettre la sélection de la partie de la Représentation Interne associée. En plus de ces manipulations implicites, les représentations peuvent être visualisées pour obtenir des **points de vue** différents sur une formule.

Les outils disponibles dans l'atelier disposent chacun d'un protocole de communication avec le dialogueur via la *prise de dialogue*. La diversité et l'intégration des utilitaires font la richesse de l'atelier. On peut imaginer un générateur de code pour archivage, un présentateur d'explication ou de démonstration, l'accès à un logiciel de traitement de texte, de courrier électronique, un logiciel de calcul formel ou d'enseignement... Leur puissance dépendra des connaissances disponibles dans la Représentation Interne.

Chaque outil, en effet, a accès à la Représentation Graphique et à la Représentation de Boîtes pour faciliter des manipulations textuelles, et à la Représentation Interne pour toutes les connaissances concernant le Langage Mathématique. Cette potentialité est limitée par l'existence ou la complexité des techniques informatiques capables **d'exprimer et d'exploiter ces connaissances**.

Un compromis est à trouver entre une modélisation (et une représentation) déclarative des connaissances, et le développement d'algorithmes spécialisés pour les cas difficiles. C'est le problème qui se pose en particulier pour l'analyse et l'édition du langage mathématique usuel. Il est éludé par EDIMATH puisque sa Représentation Interne ne comporte que des informations relatives aux manipulations textuelles, et que la déclarativité n'est pas recherchée du fait qu'il traite un ensemble d'opérateurs prédéterminés.

La Représentation Interne, en effet, est un moyen de désigner les informations traitées sur une formule, et par extension aussi l'ensemble des connaissances intervenant dans ce traitement. Nous avons choisi de traiter le **contenu, la forme et le style**, car ils sont pertinents pour les manipulations de l'usager.

Outre la structure mathématique, les informations traitées sur une formule concernent aussi l'ensemble des abstractions syntaxiques. Ces dernières permettent d'aboutir à la structure textuelle de la formule dans un langage donné, sachant que leur objectif est de pouvoir décrire la structure textuelle des formalismes destinés aux mathématiques (comme la Représentation d'Entrée, un langage de programmation ou de formatage...).

La Représentation Interne englobe ainsi les deux structures sous-jacentes (mathématique et textuelle) des manipulations de formules. La structure textuelle d'une formule est atteinte et manipulée grâce aux boîtes

de la Représentation Graphique associée à la Représentation Visuelle courante. La structure mathématique n'a pas de correspondant visuel immédiat.

Ce modèle illustre les réponses proposées par SOFTMATH au **goulot d'étranglement** de l'assistance informatique au mathématicien : la création et la transformation de formules, et la représentation et la mise en œuvre des connaissances mathématiques. Les manipulations de formules proposées précédemment reposent sur la possibilité de gérer les correspondances entre les diverses représentations introduites. Les **trois structures couplées** permettent alors la réalisation de manipulations non envisageables autrement.

3.3. UN MODELE INFORMATIQUE DE VISUALISATION STRUCTUREE ET INTERACTIVE

3.3.1. L'introduction d'abstractions mathématiques

Le transfert vers un support informatique introduit plusieurs types de langages de construction de formules :

- Un *langage de commande*, regroupant les observables informatiques produits par l'usager dans la Représentation du Terminal.
- Un *langage d'entrée*, correspondant aux données qui sont effectivement transmises aux utilitaires indépendamment de leurs paramètres de guidage. La diversité des possibilités de saisie (clavier, souris, menus, vocale, etc.) est ainsi maitrisée.
- Un *langage cible abstrait*, qui permet la construction et la manipulation de la Représentation Interne.

Par cette cascade de langages, les différentes possibilités d'expression se ramènent à un langage commun. Les langages (ou les formes) de saisie diversifiées nécessitent pour bénéficier pleinement de cette souplesse de choix d'une interpénétration des langages de commande permettant des équivalences jusqu'à une granularité fine des constituants mathématiques.

Nous nous intéressons ici aux abstractions de nature mathématique dont l'introduction est pertinente pour un traitement informatique. L'introduction d'**abstractions** comme le concept de boîte ou celle de terminal virtuel dans les interfaces interactives a permis de fournir des concepts mieux adaptés aux applications [Coutaz 86]. Pour notre architecture mathématique, nous avons introduit les notions de décompilation, de composition et d'affichage.

Avant la décompilation, le traitement est indépendant des contraintes informatiques matérielles. La décompilation produit une arborescence de boîtes chargée de gérer les contraintes de présentation. Le problème informatique est alors équivalent à celui d'un traitement de texte comme EDIMATH (cf. aussi les "pretty-printers" des environnements de manipulation de langages).

Le **concept de boîte** permet d'implanter une composition paramétrée, et de diriger l'affichage. Par exemple, selon la longueur des termes A, B et C relativement à la largeur disponible, une addition peut être représentée par :

```
A + B + C
```

ou par :

```
A + B
+ C
```

ou sinon par :

```
  A
+ B
+ C
```

Lorsque les contraintes de lisibilité le suggèrent, ou lorsque la formule déborde par rapport à la taille disponible, ce concept de boîte permet aussi de supporter les mécanismes informatiques permettant l'holophraste, c.-à-d. l'omission de certaines parties ou leur remplacement par une abréviation.

Nous avons ainsi introduit les abstractions nécessaires à une visualisation conceptuelle des mathématiques. De telles abstractions restent à définir en saisie.

Le principal écueil que nous avons rencontré dans les systèmes informatiques est la multiplicité de polices de caractères nécessaire à l'usage des mathématiques. Le risque d'ambiguïté est à bannir lors du choix des polices. Vu les positionnement variés utilisés en mathématiques, le critère important est la forme et non le code d'un caractère. Un "*" reste une astérisque indépendamment de sa taille et de son positionnement. Le symbole de multiplication pourra être obtenu par son code direct ou par une astérisque convenablement positionnée. Des objets informatiques différents ont ainsi des caractéristiques visuelles identiques ou pouvant prêter à confusion.

Pour éviter des collisions visuelles inattendues, nous proposons d'introduire et gérer l'abstraction de **symbole mathématique** indépendamment de la police d'un caractère ou d'un positionnement non contrôlé. Cette abstraction de symbole mathématique s'étend aussi à des chaînes de caractères lexicalement correctes en mathématiques. Cette abstraction a une sémantique empirique car elle correspond à une entité humainement et visuellement reconnaissable.

La notion de symbole mathématique permet de regrouper tous les caractères ou chaînes de caractères mathématiquement équivalents. Cette abstraction informatique regroupant plusieurs codages de l'entité mathématique correspondante. Plusieurs chaines de caractères sont possibles (tg, Tg, TG, tan, Tan, TAN), plusieurs polices sont acceptables, une commande d'un langage spécifique donné peut tenir lieu de symbole (dont la nature de la construction est masquée).

De même, il est utile d'introduire pour la saisie les abstractions de ligne et de taille mathématique. Une **ligne mathématique** correspond aux structures présentes à un niveau horizontal donné de la lecture d'un texte mathématique. Elle regroupe dans "$\sin^2(x)+a_i$" des entités lexicales des mathématiques comme "sin", "( )", "x", "+" et "a".

L'abstraction de **taille mathématique** est le corollaire indispensable à celle de symbole. Elle permet de mettre en correspondance des polices de taille informatique différente pour pallier l'inadaptation mathématique de leur définition. Ce manque de cohérence est dû à l'usage de polices multiples (gothiques, grecque, romaine...) au sein d'une même formule.

La notion de chaine de caractère est alors remplacée par celle de symbole mathématique. Les entités terminales sont alors des symboles. Ces symboles possèdent des attributs tels que la taille, le niveau de la ligne, la visibilité, le style (gras, italique...). Ces attributs sont manipulés dynamiquement pour mettre en évidence des propriétés (les symboles de fonction sont affichés en gras, les ligne de niveau secondaire ont une taille inférieure, etc.). Cela permet de ne présenter que certains nœuds de l'arborescence en fonction de l'information à mettre en valeur et matérialiser ainsi la notion de **point de vue**.

3.3.2. Les langages à objets

Nous avons jusqu'à présent présenté les aspects conceptuels propres à la réalisation d'une interface mathématique. Cette section est consacrée à la production du code permettant la visualisation des formules.

Avant d'étoffer nos propositions précédentes, nous avons implanté un module de génération bidimensionnelle des formules en langage à objets avec héritage multiple "Flavors". Ce module visualise des formules comme des vecteurs ou des sommations. Il gère la composition, le positionnement bit-map et l'affichage à partir de la donnée d'une Représentation de Boîtes.

Nous avons choisi de programmer ce module en langage à objets alors que les outils dont elle s'inspirait comme MENTOR et EDIMATH™ l'étaient en PASCAL (cf. [Deslaugiers 85]), et que les langages à objets étaient encore peu répandus. Ce module est intéressant car il illustre la clarté de la programmation à objets qui a permis, à l'origine, de coder le système innovant des machines LISP durant la décennie 70.

Nous avons depuis arrêté son développement puisque nos conclusions se sont révélées justifiées : la plupart des outils d'interface ont été réécrits en langage à objets au sein d'un environnement portable. Nous avons anticipé cette tendance pour les stations de travail courantes, et stoppé le développement d'un outil spécifique aux machines LISP.

De plus il est bientôt possible de décrire les boîtes avec une interface graphique directe au lieu d'avoir à programmer leur comportement (dans les développements du successeur de MENTOR : CENTAUR [Borras 87]). Ce projet CENTAUR a notamment permis la génération automatique d'éditeurs syntaxiques graphiques [Franchi-Zannettacci 88]. Le dernier problème reste la puissance d'expression autorisée par de tels outils malgré des usages non-standard comme ceux des mathématiques.

Nous nous contenterons d'abord de faire un bref rappel de notions relatives aux langages à objets sous un abord Génie Logiciel. Un **objet** est une entité qui possède des données internes et des méthodes pour les utiliser. Le style de programmation consiste à faire communiquer entre elles de nombreuses entités. Pour cela il est utile de pouvoir les générer à partir d'un nombre limité de moules différents. Cela est effectué à l'aide de la définition systématique d'un objet à partir d'une classe.

Une **classe** est la description des *champs* constituant les données internes des objets quelle engendre, ainsi que des *méthodes* dont ils disposeront. Une objet n'est alors qu'une instanciation des champs ainsi décrits. La difficulté de la programmation consiste alors à bien définir les classes, c.-à-d. les concepts manipulés.

Le programmeur est aidé pour cela par une **structure hiérarchique** entre les classes. Un mécanisme d'héritage permet de transmettre les champs et les méthodes d'une classe à sa fille. Lorsque plusieurs classes peuvent transmettre leurs propriétés simultanément, l'héritage est dit *multiple*. La définition d'une classe sans partage hiérarchique serait équivalente.

L'**héritage multiple** est plus riche car il permet de structurer les concepts en treillis. La difficulté principale consiste donc à modéliser la connaissance en un treillis de concepts adéquat. Bien que les classes ne soient pas typées, une partition conceptuelle permet de mieux organiser les connaissances. Il est utile pour cela de décrire séparément des *fonctionnalités essentielles* avant de les assembler pour former un *concept de base*.

Par exemple une automobile peut être vue comme ayant deux fonctionnalités essentielles : la partie moteur et la partie carrosserie, chacune étant elle-même décomposable. A partir de ces descriptions, le programmeur peut ajouter des *spécificités* pour obtenir une finalité sportive ou utilitaire par exemple. A cette occasion, des propriétés trop générales fournies par défaut peuvent être modifiées ou redéfinies.

Cette structuration des classes par fonctionnalités et spécificités a pour principal intérêt de définir plusieurs classes semblables en **partageant du code de façon claire** à un niveau conceptuellement élevé. L'étape suivante consiste à masquer le code du langage à objets (comme celui d'un assembleur...).

3.3.3. Représentation du concept de boîte

Le concept de boîte est défini comme une classe du langage à objets "flavors". Il constitue le maillon d'une arborescence décrivant une fenêtre d'un terminal informatique. Le paradigme objet présenté ci-dessus permet de bien structurer les diverses fonctionnalités des boîtes. Les fonctionnalités essentielles sont :

- les **caractéristiques physiques** : dimensions totales, position des points d'alignement;
- les **caractéristiques arborescentes** permettant le lien avec les boîtes filles : liste des filles et de leurs positions relatives;
- la faculté de **composition** qui instancie les dimensions de l'arborescence toute entière;
- la faculté d'**affichage**, qui permet de visualiser cette instanciation.

Nous définissons alors une boîte comme une entité possédant ces fonctionnalités essentielles (figure3.27). A cela, des spécificités sont ajoutées pour répondre à des finalités plus spécifiques. Un treillis hiérarchique est ainsi constitué.

Parmi les structures rencontrées en mathématiques, on distingue les chaines de caractères, les assemblages horizontaux ou verticaux de constituants et des assemblages spécifiques comme la sommation ou l'indexation. Pour chacune de ces structures, des spécificités sont définies et se superposent aux propriétés essentielles des boîtes. Ainsi, c'est grâce à sa spécificité que la boîte effectuant l'assemblage horizontal peut aligner ses filles.

Le cas de la **boite sommation** est représentatif du rôle des spécificités. L'explication de ce cas correspond à l'interprétation directe du code "flavors" présenté en Annexes. La boîte sommation comprend un symbole graphique sigma, des bornes inférieures et supérieures optionnelles, et le terme principal de la sommation. Nous les désignons respectivement par "sigma", "inf", "sup" et "terme".

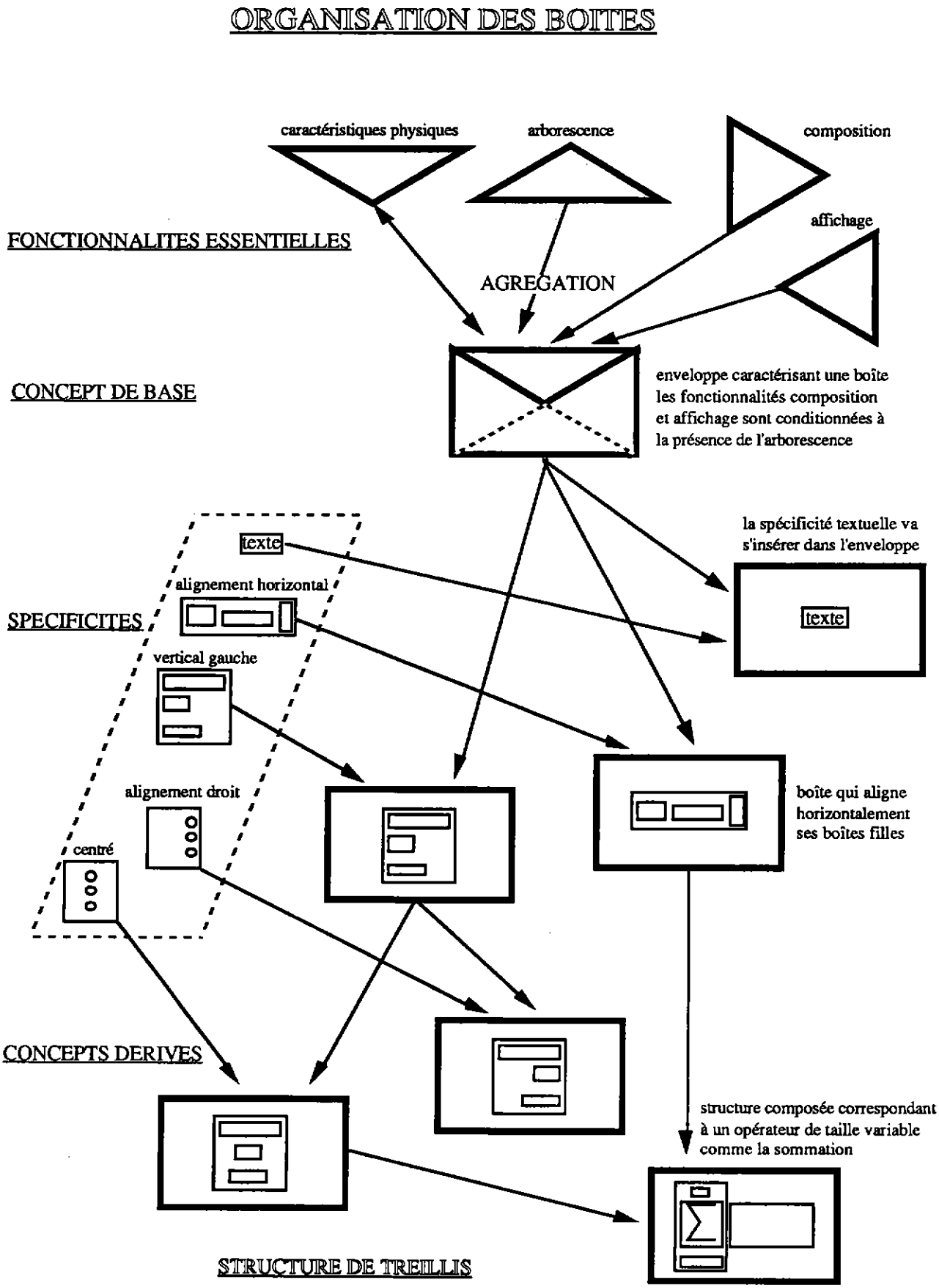

ORGANISATION DES BOITES
caractéristiques physiques
arborescence
composition
affichage
FONCTIONNALITES ESSENTIELLES
AGREGATION
CONCEPT DE BASE
enveloppe caractérisant une boîte
les fonctionnalités composition
et affichage sont conditionnées à
la présence de l'arborescence
la spécificité textuelle va
s'insérer dans l'enveloppe
texte
alignement horizontal
SPECIFICITES
texte
vertical gauche
alignement droit
centré
boîte qui aligne
horizontalement
ses boîtes filles
CONCEPTS DERIVES
structure composée correspondant
à un opérateur de taille variable
comme la sommation
STRUCTURE DE TREILLIS

figure 3.27

Le choix du contenu des bornes et du terme principal est effectué lors de l'étape antérieure : la phase d'assemblage. Elle prend en compte les caractéristiques mathématiques (sens, règles de notation...) et la forme syntaxique utilisée en saisie pour créer l'arborescence de boîtes disponible lors de cette phase de composition.

Cette arborescence étant issue des caractéristiques mathématiques, elle est constituée de la boîte mère et de ses trois filles : inf, sup et terme. Toutefois, l'arborescence utile à ce niveau devrait inclure le sigma. Pour cela, la boîte sommation va greffer une fille supplémentaire sigma comme un champ spécifique correspondant à une donnée interne. La boîte sommation gère le sigma comme l'une de ses propres filles, mais cette différenciation n'est pas perceptible au niveau du sigma. La gestion de ces informations sur la boîte sigma est l'une des spécificités de la sommation, puisque le sigma subit un traitement similaire à celui de ses demi-sœurs présentes dans l'arborescence.

La **composition** de la sommation prend ainsi en compte le sigma en plus de ses trois demi-sœurs. Ces quatre constituants une fois calculés, la sommation procède en deux étapes : la composition verticale centrée du groupe sigma inf et sup, suivie d'une composition horizontale avec terme. Le souci d'interdépendance minimale du code permet d'utiliser de façon muette les boîtes spécialisées au travail correspondant à chacune de ces deux étapes. La composition de sigma est aussi intéréssante en soi car elle dépend de caractéristiques contextuelles.

La **boîte sigma** est elle-même constituée d'un symbole graphique de dimension variable. Le sigma est adaptable en largeur et en hauteur en fonction des dimensions d'inf, terme et sup. Cette fonction d'adaptation peut être arbitrairement choisie depuis le cas discret du choix d'une taille appropriée de police de caractère jusqu'à une variation continue respectant les critères d'esthétique du sigma tout en s'adaptant aux dimensions des trois boîtes sœurs. Cette fonction constitue le cœur de la spécificité de la composition de la boîte sigma. Sa mère, la boîte sommation, se charge de lui envoyer les informations pertinentes à ce calcul.

L'**affichage** de la boîte sommation tient aussi compte de la présence du sigma en plus des autres filles. L'affichage par défaut ne prenant en compte que l'arborescence, celui de la sommation est pouvu d'une spécificité qui affiche le sigma avant d'effectuer la procédure standard. Du point de vue du sigma, cet artifice est transparent et son attitude est similaire à celle de ses demi-sœurs. Un test de cohérence est ajouté à l'instanciation de la sommation pour vérifier le nombre de ses filles.

Le code du sigma et de la sommation, présenté en Annexes, est le reflet direct de la description ci-dessus. Selon le paradigme des types abstraits algébriques, seules les procédures internes à une boîte connaissent la représentation concrète adoptée. Les appels aux autres boîtes se font à l'aide de méthodes d'accès indépendantes de la représentation adoptée. Cela permet d'effectuer automatiquement des tests de cohérence ou des actions associées à l'information demandée. Cette approche permet une **écriture incrémentale et modulaire du code**. Sa limite principale est de laisser encore le programmeur s'impliquer dans le code et non simplement à un niveau de description plus conceptuel.

3.3.4. L'extension du code

Le code est réutilisé ou étendu si la **nouvelle fonctionalité** fait intervenir des modifications fines. Par exemple, dans le codage initial, seul un alignement vertical centré est implémenté. Le centrage gauche ou droite peut apparaître utile ultérieurement. Le centrage fait partie de la composition spécifique aux boîtes verticales centrées déjà définies. On veut donc écrire trois fonctionnalités très proches, d'où l'idée de conserver les points communs en isolant les différences.

La structure adoptée permet une reconfiguration minimale du code. Une solution consiste à scinder les spécificités en deux phases : l'une correspondant à un alignement vertical, l'autre au mode de centrage. Le treillis des boîtes s'est ainsi enrichi par cette simple définition. L'adéquation du code se fait par déplacement des méthodes originales vers la première phase d'alignement vertical. La ligne de code correspondant au mode de centrage est réécrite en faisant appel au centrage spécifique défini dans la deuxième phase de la scission. Cette phase contient uniquement la méthode adaptée au mode de centrage désiré. L'alignement vertical centré est reconstitué par assemblage des deux constituants avec un comportement en tout point identique à celui du code original. Cette modification est restée locale et n'a nullement affecté le code faisant appel à ces fonctionnalités.

Cette extension des fonctionnalités a un effet différent selon la position occupée dans le treillis hiérarchique. Il est par exemple possible d'étendre la notion de boîte en rajoutant la **désignation**. Cela s'effectue tout simplement par ajout d'une méthode semblable à la composition ou à l'affichage dans la définition essentielle des boîtes.

L'algorithme de base de la désignation consiste à partir de la racine et identifier récursivement la fille candidate. Lorsque la boîte est terminale, la solution est trouvée. Sinon la boîte candidate est celle dont la position est la plus proche du point désigné. Les coordonnées étant relatives à la position de la mère, les nouvelles sont calculées et l'algorithme appelé récursivement.

Toutefois certaines boîtes ne se satisfont pas de cela, et chacune d'elle peut être dotée d'une méthode adaptée au traitement de la désignation. C'est le cas si les boîtes filles se superposent, si l'on veut désigner une boîte non terminale par sa position cartésienne, ou dans le cas de la boîte sommation. Par exemple, le sigma ne faisant pas partie de l'arborescence, sa désignation requiert un mécanisme spécial de traitement au niveau de la sommation.

Des mécanismes complémentaires de désignation sont implantables, basés sur la structure arborescente. Deux méthodes simples d'exploration sont alors le déplacement à profondeur constante par rapport à un ancêtre donné ou à hauteur constante par rapport aux terminaux. Des traitements spécifiques peuvent ici encore être introduits permettant de masquer certaines structures et considèrer un terme indicé comme terminal pour cet algorithme par exemple. La recherche d'algorithmes de déplacement structurel prenant en compte la structure particulière de l'arborescence est prometteuse car elle permet d'utiliser des connaissances mathématiques afin d'obtenir des modes de déplacement naturels.

La notion d'abstraction pose toutefois problème lors de la **définition concrète d'un nouvel élément** d'une abstraction donnée. C'est le cas par exemple de la définition d'une nouvelle structure d'assemblage, d'une nouvelle boîte pour la composition ou la création d'un nouveau symbole. Leur définition concrète requiert de la spécifier à l'aide des éléments faits pour être masqués par l'abstraction. Par exemple la définition d'une boîte union composée d'un symbole de taille variable "$\cup$" et d'une structure similaire à celle de la sommation, va impliquer une connaissance de la structure des boîtes.

Comme toujours en génie logiciel, il y a plusieurs solutions complémentaires :

- Prévoir à la conception des spécifications aussi adaptées que possible. En effet, la généralisation d'un opérateur binaire à un nombre quelconque d'opérandes est une opération courante en mathématique. En ce sens la structure utilisée pour la sommation n'est qu'une instance d'une structure qui aurait pu être envisagée lors de la spécification.

- Fournir des utilitaires pour personnaliser ou étendre l'environnement. Ce coût logiciel plus élevé se justifie pour un système très ouvert. Un logiciel d'assistance à la création peut être écrit pour les principaux types d'extension. Il requiert un travail important pour produire du code à partir d'abstractions adaptées à l'interface. Ici ce logiciel doit permettre de spécifier les conditions de composition de l'union ainsi que les contraintes à respecter. Un langage d'expression possible est formé de choix et de combinaisons de caractéristiques prédéfinies, permettant un apprentissage et une définition par l'exemple.

- Produire une nouvelle version du code. Ce choix est souvent le meilleur en phase de développement. Il reste possible en phase opérationnelle à condition de disposer d'un code clair, modulaire et bien documenté. L'interaction entre conception et utilisation est inévitable et cette adaptation reste la plus économique pour éviter des spécifications trop exhaustives voire impossibles.

Malgré sa clarté de conception, cette structure hiérarchique de boîtes comporte des limitations inhérentes. Elle ne permet que des propagations locales et un mode de décision décentralisé. Des algorithmes sophistiqués d'optimisation globale (tels que résolution de contraintes) sont peu adaptés à ce traitement. Il faut alors tenir compte d'un contexte général à l'arborescence et se servir de cette dernière comme support d'information. La décision prise globalement est alors propagée dans l'arborescence. Même si la complexité n'est pas optimale, ce procédé est toujours réalisable. Le traitement décentralisé par objets présente toutefois suffisament d'avantages pout s'avérer indispensable en phase de prototypage et développement.

Nous avons ainsi brossé dans ce chapitre un panorama détaillé de niveaux informatiques permettant la conception d'un **système ergonomique spécialisé en mathématiques**. Le chapitre suivant détaille l'analyse et la Représentation Interne des formules dans un tel Atelier Mathématique Intégré.

4. LE TRAITEMENT DU LANGAGE MATHEMATIQUE

4.1. LA PRISE EN COMPTE DU LANGAGE MATHEMATIQUE

Notre étude de l'activité mathématique a montré la nécessité de la prise en compte du langage lors de la constitution de modèles mathématiques. Nous avons vu en effet que le Langage Mathématique est une composante essentielle de la description de problèmes, l'introduction de définitions, la production de démonstrations, la génération d'explications, etc. Nous abordons ici le problème du ***traitement*** du Langage Mathématique.

Une partition, classique en linguistique, de l'activité de traitement du langage est : **l'analyse, la représentation et la génération.** La représentation est dépendante de l'utilisation qui en est faite. Vu l'importance de la forme et du sens en mathématiques, nous avons choisi une représentation qui combine les informations syntaxiques et mathématiques.

Les objets mathématiques introduits et décrits par le langage ayant des fonctions différentes, nous proposons de différencier l'usage du langage selon la nature de ces objets. Les aspects langagiers sont alors associés aux aspects mathématiques dans la modélisation de l'activité mathématique. Cette modélisation détermine notamment le "sens" que nous sommes amenés à donner à un énoncé du langage. Elle permet en particulier la prise en compte d'énoncés successifs d'un discours.

Indépendamment de cette représentation du sens, le traitement du Langage Mathématique présente des similarités syntaxiques selon le type de langage : Phrases, Formules, Schémas, Croquis. Les techniques informatiques de traitement des langages de Formules s'apparentent à celles des langages de programmation, tandis que celles relatives au traitement des langages de Phrases s'apparentent à celles du Langage Naturel.

Pour ce qui concerne les langages de Phrases, l'ouvrage de Gérard Sabah [Sabah 89], pourtant très complet et profond, ne cite pas le Langage Mathématique. Nous considérons qu'il s'agit d'une lacune provenant essentiellement de l'immaturité de traitement du langage dans les applications mathématiques, et de l'absence d'une linguistique des mathématiques. Rappelons que la rigueur des concepts et de l'argumentation permet l'instauration d'un dialogue intelligent[1] et facilite la réalisation d'applications mathématiques non restreintes à un micro-domaine opératif.

L'utilisation de techniques de traitement du Langage Naturel pour les mathématiques nous semble donc prometteuse pour les disciplines scientifiques concernées. Toutefois, un traitement du langage non superficiel nécessite un **système spécialisé.** Nous présentons en Annexes le schéma général de traitement du langage proposé dans le cadre du projet japonais de 5ème génération. Ces techniques sont donc très lourdes et ne peuvent s'envisager que dans le cadre d'un atelier intégré...

Le Langage Mathématique présente une spécificité qui le distingue notamment du Langage Naturel et des langages de programmation :

Le langage utilisé est défini par le mathématicien, pour son problème.

Même si quelques touches sont parfois ajoutées au modèle mathématique sous-jacent, l'essentiel de ces définitions porte sur des aspects moins profonds : elles décrivent les objets construits à partir de ce modèle et définissent les notations employées pour cela.

Dans le problème de la surface de la parabole (partie I), l'action du mathématicien n'était pas de décrire ce que sont un foyer ou un pôle, mais bien de les désigner dans le problème et de définir leur relation effective. La définition du langage n'apportait pas d'extension aux structures conceptuelles, mais au contraire définissait leur utilisation et les moyens langagiers nécessaires à cela. Rappelons néanmoins que ce modèle mathématique conceptuel était "prêt à l'emploi" et prévoyait les moyens langagiers nécessaires à la désignation et à la manipulation de ses composants.

[1] La seule référence à des exemples mathématiques dans [Sabah 89] provient d'ailleurs de l'exploitation des travaux de Jean Véronis [Véronis 89] sur le traitement de l'erreur dans le dialogue, car J. Véronis les a appliqués aux énoncés de géométrie élémentaire pour l'EIAO.

L'exploitation du langage dans la phase d'utilisation se fait donc essentiellement par une définition des aspects de surface du langage relativement à un modèle conceptuel stable en première approximation. La définition du langage est alors essentiellement de nature terminologique (mots, syntagmes et notations).

L'ampleur de cette définition du langage utilisé introduit une **variabilité notable** des caractéristiques de surface du langage. Ce facteur constitue la spécificité du Langage Mathématique par rapport aux autres langages à traiter. Cette variabilité rend souvent caduque les solutions proposées pour des langages de programmation définis a priori.

L'utilisation du Langage Mathématique s'effectue alors par l'adaptation préliminaire du langage aux besoins du mathématicien. Cette adaptation est traditionnellement résumée par une liste ou un index de symboles et de notations.

Nous traiterons uniquement ici la spécificité du Langage Mathématique qui influe sur son traitement : **l'adaptation syntaxique du langage** par des définitions introduites par le mathématicien, et les conséquences de ce procédé sur la nature et les outils de traitement du langage. Cette spécificité se décompose en deux points :

- la mise à disposition d'un langage[2] de description permettant l'introduction de ces définitions par le mathématicien.
- la prise en compte de ces définitions dans le traitement du Langage Mathématique.

Il y a alors production d'un mécanisme global à partir de définitions locales. Cette forme de synthèse est, par essence, très difficile dès que les structures manipulées ne sont pas régulières.

L'adaptation du langage est nécessaire dans toutes les disciplines scientifiques. Les exemples proposés en Annexes concernent les mathématiques, l'informatique, la physique et la biologie. La diversité des choix possibles est illustrée pour les mathématiques avec des définitions issues de l'analyse, de l'algèbre, de la théorie des nombres et de la théorie des graphes.

Le besoin de ces définitions affecte tout autant les traités que les articles. Dans les traités, les choix se veulent homogènes bien que le sens d'un symbole puisse être redéfini ou particularisé par chapitre ou par section. Un article a, quant à lui, besoin de préciser rapidement de nombreux choix langagiers afin de pouvoir travailler le contenu.

L'article de la visualisation de matrices (en Annexes) est caractéristique car la première page est directement consacrée aux notations du problème. Elle précise les nombreuses subtilités de notation employées ainsi que le sens qui leur est attribué. Il s'avère que l'utilisation de cette table est indispensable pour l'interprétation de langage. Sa prise en compte est aussi indispensable pour la vérification des résultats ou pour une extension de ce travail.

Une telle table peut être lue a deux niveaux :

- au niveau syntaxique, pour l'analyse et la génération de certaines structures syntaxiques ;
- au niveau sémantique, pour l'interprétation de ces structures syntaxiques dans le problème.

Au niveau sémantique, la représentation de cette table doit être établie pour tout travail sur le problème. Lorqu'elle a déjà été construite, la transmission de cette table et la constitution de sa représentation sont indispensables dans un AMI. Par contre, sa construction initiale se fait par récupération et adaptation à partir des structures conceptuelles liées au problème. Sa construction se fait alors progressivement en fonction des besoins de formulation et de résolution du problème, comme cela a été étudié pour la surface de la parabole.

La quantité de notations habituellement nécessaires est telle, qu'une mauvaise familiarité des notations constitue un handicap mnémonique majeur. Le choix des symboles et de leurs similarités en fonction de leur rôle sémantique nécessite donc une attention particulière. L'usage pragmatique du choix des notations permet aux mathématiciens de retrouver aisément leur interprétation et leur rôle dans le problème.

Une assistance qui faciliterait la définition de ces listes de définitions est souhaitable. Elle exploiterait les structures conceptuelles du problème conjointement à une expertise pragmatique de définition du langage

---

[2] Dans ce chapitre, le mot "langage" prend un sens plus informatique que l'on retrouve dans "théorie des langages". Les difficultés qui nous concernent résident principalement dans le choix adéquat du langage et de son procédé de définition.

et de génération de notations. Nous n'avons pas toutefois réalisé de Système-Expert spécialisé dans le nommage, ou permettant au mathématicien de définir aisément des règles heuristiques pour faciliter le processus. Comme une automatisation totale engendrerait des choix criticables, il est bon que le mathématicien puisse guider certains choix, voire que le système propose de réviser globalement certains choix en cours de session.

Nous nous intéressons dans la suite de ce chapitre aux aspects syntaxiques de l'adaption du langage, dans le cas particulier des langages de Formules. Nous avons néanmoins toujours pris en compte la compatibilité avec les langages de Phrases.

4.2. L'ANALYSE DES FORMULES

4.2.1. Présentation de l'analyse

Le reste de ce chapitre décrit la gestion des structures des formules, incluant l'analyse, la représentation et la génération de formules. Cette gestion se veut générale afin d'être **indépendante d'un domaine** mathématique particulier et de permettre un **traitement multi-langage**. Nous présentons les mécanismes de gestion à partir de la réalisation d'un d'analyseur restreint, qui nous a permis - par son expérience - d'affiner le modèle proposé.

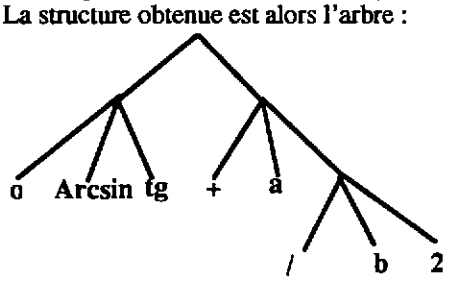
L'objectif de cette étude est de construire un système capable à partir d'une entrée au clavier d'une formule mathématique de vérifier sa correction syntaxique, de procéder à une analyse sémantique de la formule en la décomposant en objets mathématiques dont les types sont synthétisés, de déceler les incohérences sémantiques en fonction des connaissances acquises par le système et d'interroger l'utilisateur pour corriger les fautes ou ambiguïtés de saisie. Ces fonctionnalités seront dues à la qualité du mécanisme d'analyse, qualité proche de ce que l'on trouve actuellement dans certains environnements de programmation.

L'analyse consiste à **transformer un langage d'entrée en un langage cible** en exploitant le contenu informatif du langage d'entrée. Un cas particulier consiste à utiliser un langage d'entrée dit *concret* (car le codage de l'information ne met pas directement en valeur la structure de l'information), en un langage cible dit *abstrait* (où l'information visée est complétée et structurée de façon régulière). C'est le cas notamment pour l'analyse des formules.

Le langage d'entrée considéré est déjà un codage informatique provenant de la saisie des formules par le mathématicien. Sa structure est pauvre et comprend essentiellement des séquences de caractères. L'analyse consiste à reconnaître - ou non - les mots bien formés du langage d'entrée afin d'en produire une représentation abstraite. L'analyse se déroule classiquement en trois étapes, illustrées sur la séquence de caractères "Arcsinotg(a+b/2)" :

- L'analyse lexicale consiste à regrouper certains de ces caractères en symboles afin de produire un langage reconnaissable par l'analyse syntaxique.
  Nous obtenons ici la séquence de symboles : "Arcsin" "o" "tg" "(" "a" "+" "b" "/" "2" ")".

- L'analyse syntaxique consiste à déterminer une correspondance syntaxique qui obtienne une structure définie par ailleurs et élimine les séparateurs syntaxiques du langage précédent.
  La structure obtenue est alors l'arbre :



Nous avons choisi ici une structure permettant l'expression de fonctionnelles (fonction de fonctions). Un nœud non terminal s'interprète comme l'expression formée de l'application du fils gauche sur les suivants. La détermination de cette structure à partir d'ambiguïtés positionnelles entre a+(b/2) et (a+b)/2 a été levée par une **priorité** entre / et +.

- L'analyse sémantique revient à déterminer si cet arbre est mathématiquement bien formé.
  Par exemple, elle consiste ici à vérifier que si "b" est un entier, il peut être considéré comme un rationnel ou un réel pour être compatible avec "/". De même "Arcsin" et "tg" sont bien des fonctions et peuvent être compatibles avec la composition "o". Le fait que tg(a+b/2) ne soit pas forcément élément de [-1, 1] ne peut être connu avec les informations présentes. La compatibilité peut être obtenue de façon externe, par rattachement d'une **contrainte mathématique** à vérifier ou à propager.

La compatibilité sémantique ne peut être uniquement obtenue à partir des informations structurelles des formules. Il est alors nécessaire de disposer d'une représentation chargée de stocker des informations supplémentaires concernant les formules. Un étiquetage de chaque nœud de la structure est une solution : ainsi, le fait : "tg(a+b/2)∈ [-1, 1]", est rattaché au nœud le plus haut, car il n'aurait pas lieu d'être sans ce nœud d'application fonctionnelle. Cette méthode peut être étendue pour des informations de nature syntaxique. Cela permet de plus, de représenter une structure incomplète en capturant dans l'étiquetage les informations nécessaires à la reprise de l'analyse.

L'analyse présente donc trois difficultés intrinsèques, que l'on peut analyser en terme de transfert d'information :

- Lorsque le langage d'entrée ou le langage cible est difficilement reconnaissable par un processeur, l'**extraction ou la synthèse de l'information** à y coder nécessitera des méthodes complexes. Le problème consiste alors à décomposer et à coordonner les traitements.
- Lorsque le contenu informatif du langage d'entrée est inférieur à celui requis par le langage cible, le mécanisme chargé de la correspondance doit **rajouter l'information** manquante. Le fait de chercher à expliciter l'ajout de cette information amène une difficulté supplémentaire.
- Lorsqu'un traitement fin est nécessaire, il faut gérer des **structures incomplètes** et des méta-informations de description des langages.

L'analyse des formules mathématiques cumule ces difficultés.

4.2.2. Notre approche de l'analyse

L'analyse peut être considérée comme un problème de synthèse d'informations locales afin de trouver une structure globale respectant leur cohérence. Selon ce point de vue, effectuer une analyse consiste à raisonner sur un **langage de description** du langage d'entrée, du langage cible et de leur correspondance.

Par exemple, savoir que "+" est un opérateur binaire infixé permet de déterminer la structure arborescente locale au "+", ainsi que de savoir que son premier argument finit à sa gauche dans la syntaxe concrète, et que le second débute à sa droite. Des informations sur les séparateurs et la priorité entre opérateurs permettent alors de déterminer l'étendue de ces arguments dans la syntaxe concrète. Par exemple l'argument de droite continue tant qu'il existe des opérateurs infixés ou post-fixés de priorité supérieure à l'opérateur "+" (un opérateur de priorité supérieure groupe ses arguments avant). Cette notion de priorité autorise la définition de **grammaires d'opérateurs**.

Cette approche d'analyse permet de répondre au besoin de déduire et construire des structures, temporaires ou non, dès la prise en compte d'une nouvelle information. Il serait possible d'envisager simplement l'analyse sur ces principes si d'autres difficultés ne venaient s'y rajouter :

- la possibilité de ne pas savoir quel est le rôle d'un caractère sans analyse syntaxique ou sémantique (le "|" de la valeur absolue est-il fermant ou ouvrant, le "d" après une intégrale est-il la fin attendue ou fait-il partie d'une autre structure lexicale, etc.) ;
- la possibilité pour un symbole concret de correspondre à des structures différentes ("-"), d'avoir des arguments optionnels ou un nombre d'arguments variables ;
- la possibilité d'écrire certaines fonctions avec ou sans leurs arguments ;
- la possibilité d'utiliser des opérateurs implicites susceptibles d'intervenir dans toute juxtaposition (la multiplication, la concaténation de séquences, le produit externe, etc.).

Par ailleurs, la prise en compte des fortes capacités d'analyse des mathématiciens tend à diminuer l'information disponible en entrée, et profiter de l'analyse pour leur faire déduire des informations implicites telles que "a" est un nouveau symbole de type rationnel ou réel.

Nous avons construit un analyseur en PROLOG selon cette approche d'analyse. PROLOG gère le non-déterminisme de l'analyse (les choix possibles). L'analyseur utilise la technique des différence-listes pour gérer des structures dynamiques incomplètes[3] [Shapiro 86]. Cela permet de construire des arbres dynamiques à compléter par l'unification,. Cela autorise, par exemple, à anticiper sur les informations à

[3] Le lecteur averti notera le différence entre une structure dynamique, activée dans l'interpréteur PROLOG et son mécanisme d'unification, et une structure permanente accessible dans d'autres états de l'interpréteur. C'est une telle structure permanente, associée au recueil des informations qui la concernent, qui est considérée lorsque nous parlons des structures incomplètes des formules.

venir en construisant, dès l'arrivée d'une parenthèse, la parenthèse fermante correspondante. Les séparateurs d'arguments sont acceptés uniquement dans cet état d'analyse.

Cet analyseur prend en entrée la chaîne de caractères à analyser, et en paramètre des **informations sur les opérateurs** du langage cible susceptibles d'être analysés. Ces informations incluent aussi les symboles déjà connus et comprennent :

- le nom de l'opérateur dans le langage cible ;
- son type ;
- la (ou les) chaînes de caractères possibles dans le langage d'entrée.

Pour chaque couple opérateur-chaîne, les informations concernant le langage d'entrée sont :

- la priorité lexicale de groupement des caractères de l'opérateur ;
- la priorité d'application de l'opérateur à ses arguments ;
- la nature de l'associativité de l'opérateur (non associatif, assoc gauche, assoc droite) ;
- la possibilité d'utilisation de l'opérateur comme argument d'une fonctionnelle (nous la supposons toujours possible dans le langage cible) ;
- le caractère pré, inf, ou post-fixé de l'opérateur ;
- la nature du parenthésage, à moduler selon les deux points précédents (obligatoire, selon la priorité).

Cette modularité est originale et constitue le **"formalisme de description du langage d'entrée"** qu'utilise notre analyseur. Elle constitue la partie la plus lisible du code présenté en Annexes. L'analyse proprement dite est décomposée en deux étapes :

- une analyse lexicale ;
- une analyse syntaxico-sémantique.

L'analyse lexicale gère la reconnaissance de symboles et l'introduction de symboles par défaut (constantes, fonctions, variables en fonction d'une partition des lettres de l'alphabet). Des symboles par défaut sont introduits pour compléter une reconnaissance qui ne pourrait s'effectuer avec les symboles déjà connus. Une fois introduits, ils présentent la même structure d'information qu'un opérateur permanent comme sinus. En revanche, ils peuvent être oubliés lors d'une autre option d'analyse. L'analyse lexicale gère aussi les nombres et le parenthésage. Elle crée un opérateur fonctionnel de parenthésage pour regrouper les symboles entre les parenthèses : c'est déjà un début de structure arborescente.

L'analyse syntaxico-sémantique regroupe les étapes d'analyse syntaxique et sémantique afin de pouvoir éliminer rapidement des incompatibilités. Le type des constituants est notamment utilisé pour discriminer entre l'application d'une fonction à ses arguments (par juxtaposition) et les opérateurs implicites déclarés dans les informations sur les opérateurs. Cette difficulté est accrue par l'absence de parenthésage obligatoire.

La caractéristique de cette analyse est une **stratégie ascendante** majoritaire, qui construit l'arborescence à partir de la chaîne d'entrée. Cette prise en compte est prioritaire en mathématiques car les structures possibles sont trop diversifiées pour être essayées inconsidérément.

Nous avions d'abord dégagé des règles de rattachement incrémental immédiat, qui construisent des structures incomplètes partielles et les font muter lors de la prise en compte d'un nouvel item. Par exemple : "tg" "(" "a" donne l'arbre "*tangente(a)*", l'arrivée de "+" donne "*tangente(plus(a,?))*", avec le point d'interrogation représentant une branche pendante qui est complétée à l'arrivée de "b". Nous escomptions ainsi expliquer l'analyse en cas d'échec, ou proposer des complétions possibles et leurs conséquences. Nous n'avons pas implémenté cette stratégie de construction qui nécessitait un traitement spécialisé, requérant des informations et un savoir-faire non disponibles dans la description du langage d'entrée et du langage cible.

Nous avons alors choisi une méthode de construction exploitant plus directement les informations déclarées. Elle est plus systématique : elle consiste à séparer la chaîne d'entrée en segments encadrés par des séparateurs fiables, et à choisir récursivement dans chaque segment un opérateur principal qui respecte les contraintes propagées (type, priorité, parenthésage, etc.). Chaque nœud dispose alors d'une liste d'attributs décrivant les contraintes qu'il respecte à l'intention d'un contexte englobant (dans lequel il servirait d'argument), et celles que ses arguments éventuels doivent respecter.

Cela permettrait éventuellement de suspendre l'analyse en cas d'informations incomplètes, et d'accepter une reprise incrémentale lors de l'arrivée d'une nouvelle information. Cela permettrait, de plus, de **rattacher à un opérateur du langage cible les informations le concernant**, de façon à ce qu'il puisse déterminer les contraintes et superviser l'analyse de ses arguments. Nous avons simplement

validé la faisabilité sur des cas simples comme "Arcsinotg(a+b/2)". Notons que nous n'avons pas traité le problème de la validité mathématique, qui interviendrait par exemple dans $\sum_{k \bullet p \leq 0} j$.

4.2.3. L'état des lieux actuel

Cet analyseur a été écrit en 1988. Il a permis de tester et valider des idées originales comme l'analyse du sens des formules à partir de la description individuelle "déclarative" de chaque composant mathématique, la prise en compte simultanée d'informations syntaxiques et sémantiques, l'attribution d'un sens par défaut aux symboles, et la tolérance aux ambiguïtés lexicales qui surviennent lorsqu'on ne met pas forcément les parenthèses ou les espaces blancs séparateurs. Nous l'avons laissé de coté faute d'aboutir à une stratégie plus souple se basant sur une déclarativité plus générale de l'analyse et du code.

Notre approche de l'analyse présente donc comme originalité principale de :

- prendre en compte des informations locales **déclaratives** (et donc facilement extensibles) sur les opérateurs pour produire une analyse globale :
  cela permet de s'adapter à des domaines mathématiques variés et à des situations évolutives ;
- gérer des structures **incomplètes**, leur combinaison et des informations qui leur sont rattachées :
  cela autorise une analyse d'expressions incomplètes et permet de représenter explicitement dans des attributs les informations ne pouvant être décrites par une production du langage cible (partiellement validé sur des attributs comme le type ou la priorité) ;
- accepter du mathématicien un **langage souple** (opérateur implicite par juxtaposition, symboles par défaut, positions syntaxiques variées, parenthésage optionnel, etc.) :
  cela permet de prendre en compte les usages mathématiques.

Nous cherchions par ailleurs à ce que la description du langage d'entrée, du langage cible et de leur correspondance soit utilisée pour l'analyse comme pour la génération, et puisse être localisée et rattachée aux opérateurs mathématiques concernés. Les attributs permettraient de plus de regrouper, autour de ces opérateurs, l'information concernant la structure textuelle du langage d'entrée.

L'amélioration principale à apporter à notre analyseur consisterait à étendre les fonctionnalités de description des attributs. L'objectif général est que l'analyse soit plus proche de manipulations de structures incomplètes, afin de capturer facilement un état stable de l'analyseur, et permettre ainsi des manipulations incrémentales.

Un analyseur faisant des choix similaires aux nôtres a été effectué indépendamment [Bouhier 89]. Cet analyseur très intéressant a pour particularité de se rattacher à la représentation interne d'EDIMATH [Quint 83], conçu aussi à l'IMAG. Il s'insère dans une projet d'éditeur de formules mathématiques lancé en 1987 et baptisé Intermath.

Il est intéressant de noter que l'analyseur de M. Bouhier choisit des techniques très proches des nôtres (PROLOG, grammaire d'opérateurs, langage souple, etc.) bien qu'EDIMATH ait initialement été réalisé en PASCAL. Ce rapport détaillé permet d'approfondir les techniques employées : il contient notamment des grammaires lexicales, syntaxiques et sémantiques successivement employées. Il conclut sur l'intérêt de constituer un système extensible par l'utilisateur, dans la mesure du possible.

Le différence notable est que l'analyseur de M. Bouhier est quasiment opérationnel et couvre toutes les formules saisies par EDIMATH. Il est destiné à fournir une passerelle vers les systèmes de calcul formel et ses critères de traitement sont dépendants du formalisme interne choisi par EDIMATH. Notre recherche étant plus autonome, nous avons pu dégager des critères plus généraux favorisant l'extensibilité par l'utilisateur.

Il est maintenant possible de proposer un analyseur mathématique sophistiqué, adapté à un traitement interactif des formules. Les points à améliorer restent donc la sophistication des moyens de description des informations (déclarations simples et riches) et la facilité d'utilisation de l'analyseur (structures incomplètes).

4.2.4. Les besoins spécifiques aux mathématiques

Les choix précédemment effectués suggèrent les besoins de traitement spécifiques aux mathématiques. Chaque point induit de fortes contraintes sur les mécanismes informatiques à concevoir. Les mots clés sont :

- **famille de langages :**
  la variabilité des notations et des conventions possibles revient à pouvoir traiter une famille de langages. Il faut donc travailler à partir des descriptions de ces langages (méta-outils), et non concevoir directement les outils pour un langage.
- **description des langages :**
  la description d'un langage doit servir aux outils amenés à le manipuler. La description des langages utilisés doit s'apparenter aux procédés et aux concepts de description du mathématicien. De plus, elle doit aussi permettre la description des structures incomplètes utilisées par les outils dans les manipulations des langages. Un test de bonne description consiste à savoir indiquer comment compléter une structure incomplète.
- **incrémentalité :**
  c'est une exploitation des structures incomplètes. Elle évite d'attendre une structure complète pour être en mesure d'effectuer un traitement et produire un résultat. Elle permet notamment de démarrer une analyse n'importe où, et analyser par exemple le nuplet des arguments d'une fonction. Dans une autre perspective, elle permet d'étendre le langage par des modifications locales.
- **expertise d'étude d'un langage concret :**
  une telle expertise consiste à pouvoir exploiter la description d'un langage pour son analyse et sa génération. Elle permet de déterminer des séparateurs syntaxiques fiables, d'isoler des ilots de confiance dans le traitement, et de détecter et prendre en compte les collisions lexico-syntaxiques à partir de la description du langage. Elle consiste par exemple à trouver quand et comment il est possible de différencier les diverses syntaxes du Σ. Elle consiste aussi à décrire où la chaîne de caractères ";" (voire "s", ou "pi", ou "!", etc.) est susceptible d'intervenir. Le recours à une telle expertise d'étude des langages est **inexploré à notre connaissance.**

Il est important de souligner que même la prise en compte individuelle de ces points reste essentiellement un problème ouvert. Il est intéressant de les aborder pour les mathématiques car cela correspond à un besoin à satisfaire, bien qu'il soit possible dans un premier temps de remédier à ces problèmes par des moyens ad hoc. De plus, ces développements intéresseront utilement des langages d'un traitement plus simple.

L'aspect multi-langage est déjà abordé dans les environnement de programmation génériques. La différence essentielle avec les mathématiques est que les langages traités pour la programmation sont peu nombreux et connus : ils peuvent être définis globalement un à un par un informaticien-expert. De même, les correspondances sont souvent des transformations codées individuellement. Le même principe doit toutefois être repris, puis assoupli et spécialisé aux mathématiques :

Un traitement général du langage utilise la description

du langage d'entrée, du langage cible et de leur correspondance.

Les techniques développées pour les langages informatiques présentent souvent des insuffisances : limitation de l'expressivité du langage décrit, perte d'informations sur le langage concret, utilisation implicite de connaissances, manque d'incrémentalité pour l'analyse ou l'extension d'un langage, algorithmes développés pour des cas plus simples, etc. Néanmoins, la description et le traitement informatique des langages a fait, et continue à faire, d'énormes progrès.

Il est important de noter qu'un système comme Mathematica (et les autres "grands" systèmes de calcul formel) ne peuvent analyser que des expressions complètes. Malgré la simplicité de l'expression informatique de son langage, il ne peut proposer une complétion, encore moins une correction. Souvent d'ailleurs, une saisie descendante guidée par la description du langage n'est pas possible. Les objectifs d'**analyse incrémentale d'expressions incomplètes** restent donc étrangers aux préoccupations des systèmes mathématiques habituels.

La description des langages peut alors s'effectuer par le biais de grammaires qui codent en fait un système de production des expressions bien formées. Toutefois, il est nécessaire de bien distinguer la structure du langage, de la grammaire qui a servi à sa description. Plus que les grammaires à attributs, les

grammaires logiques [Cohen 87, Pereira 80] basées sur PROLOG présentent un bon compromis grâce à leur expressivité [Deransart 85]. Toutefois elles sont basées sur une stratégie descendante. Or la diversité des générations possibles en Langage Mathématique favorise la propagation de contraintes depuis la formule. Un procédé ascendant est développé dans [Matsumoto 86] pour le langage naturel afin d'utiliser l'apport d'un **dictionnaire** (c.-à-d. description des opérateurs). Ce procédé utilise la grammaire logique et le dictionnaire comme description du langage (niveau méta), et il l'interprète (niveau méta-méta) d'après un algorithme de recherche avec retour arrière. Le tout est codé en PROLOG.

La tendance actuelle s'oriente vers des ateliers spécialisés améliorant les procédés de description des langages. Nous pensons que cet axe de recherche est prometteur en mathématiques. Cet axe s'apparente à l'étude et à l'utilisation de grammaires spécialisées pour le traitement du Langage Naturel. La mise au point d'une **description de langages pour les mathématiques** à partir de concepts accessibles par le mathématicien reste à définir.

Un objectif possible est l'introduction d'une nouvelle structure syntaxique à partir d'une formule. Par exemple, le mathématicien saisirait une nouvelle structure textuelle :

$$\sum_{i \in I} a_i$$

et indiquerait comment bâtir la structure mathématique : l'argument gauche de "$\in$" est sélectionné pour correspondre à la variable de la structure mathématique, l'argument droit au domaine, et le terme a_i pour correspondre à l'expression principale.

Les emplacements manipulés seraient alors généralisés : i doit rester une variable, I ne correspond pas seulement à un symbole d'ensemble mais à toute expression ensembliste, et a_i correspond à une expression mathématique. Le mathématicien pourrait ainsi développer - et consulter - sa propre grammaire, et la décrire à partir d'exemples génériques. Il est nécessaire pour cela de pouvoir engendrer l'analyse à partir de la description du langage, ainsi que nous l'avons esquissée.

La structure mathématique serait plus complexe à indiquer pour :

$$\sum_{i \mid n} a_i$$

car le domaine serait {d$\in$N, d|n}. Il faudrait récupérer la borne inférieure "i|n" pour la considérer comme la relation principale d'un ensemble défini en compréhension sur le domaine de travail par défaut.

Quelles que soient les solutions proposées, la description des nouvelles définitions ou notations introduites en mathématiques reste un problème prioritaire pour les systèmes mathématiques ouverts. Il inclut notamment la description grammaticale des expressions mathématiques, description capable d'être utilisée en analyse et en génération. Ce problème se retrouve en Langage Naturel pour la constitution de dictionnaires.

4.3. LA GESTION INTERNE DES FORMULES

4.3.1. Une organisation mathématique "à objets"

Cette section concerne la gestion des données et des connaissances utiles aux formules. Nous nous intéressons ici uniquement à la représentation des formules : l'exploitation mathématique de ces formules est considérée comme indépendante.

Les formules étant des relations mathématiques utilisées dans une théorie, nous distinguons trois sortes d'objets intervenant pour la **modélisation de la structure mathématique des formules**[4] :

[4] Cela est notamment visible dans l'exemple "Arcsinotg(a+b/2)" précédent, où le seul nœud non terminal possible est l'application de fonctions (la ß- réduction du λ-calcul).

- le *squelette des références*, qui représente la relation par les liaisons entre des constituants terminaux.
- les *composants des formules*, c.-à-d. les constituants terminaux comme "tg", "o", "a" ou "2" intervenant dans la relation ;
- les *concepts mathématiques* de la théorie nécessaire à l'exploitation de ces relations, c.-à-d. les caractéristiques de la fonction tangente, des fonctions trigonométriques, des entiers ou des constantes.

Il y a donc, comme le besoin en a été aussi éprouvé lors de l'analyse, **des informations de nature structurelle, et des informations de nature descriptive**. Cette dualité est fondamentale pour représenter la gradation entre informations explicites à un emplacement donné, informations déclarées dans le texte, et informations implicites concernant la théorie mathématique. Les informations de nature descriptive concernent les aspects mathématiques, mais aussi les aspects textuels et des informations méta-descriptives.

Nous avons relevé dès [Moulis 87] que la structure conceptuelle des mathématiques se prête à une organisation et à une gestion "à objets". Un concept mathématique est alors représenté par une classe d'un langage à objets spécialisé pour la représentation des connaissances. Les composants des formules sont des instances de ces classes. Ils bénéficient ainsi d'une description générale commune à toutes les utilisations mathématiques envisagées dans la modélisation.

L'**organisation structurelle** des concepts mathématiques, correspond à l'agencement des classes. Par exemple, la fonction "sinus" a des propriétés propres, et d'autres plus générales. Elles sont partagées par des fonctions impaires, trigonométriques, ou plus généralement par les fonctions réelles. Ces caractéristiques induisent la structure des classes. Ainsi, la formule "$\sin^2 x_k(t)$" bénéficie des propriétés relatives aux fonctions trigonométriques (ou propres aux fonctions sinus et cosinus), et la substitution de "$x_k(t)$" en "$2k\pi+t$" est suivie d'une simplification.

Ce principe d'organisation s'applique de même pour les aspects textuels. Par exemple, les procédures de choix d'assemblage sont l'une des facettes caractérisant les objets mathématiques. La classe des fonctions possède alors des caractéristiques d'assemblage, et délègue à ses classes plus spécifiques, comme la fonction d'indexation, la faculté de modifier ces caractéristiques. La description des langages rattache alors aux concepts les informations qui les concernent.

Ainsi, les composants utilisés dans une formule sont des objets obtenus par instanciation d'une classe. Les classes permettent alors la description des informations rattachées à chaque nœud, ainsi que leur traitement. Contrairement aux manipulateurs symboliques, cette représentation est composée d'objets informatiques, et non de passifs identificateurs. Elle est utilisée pour des manipulations généralisées : calculs bien sûr, mais aussi restitution, désignation, édition, etc.

Cette **organisation des connaissances "à objets"** correspond bien à la granularité des mathématiques. Elles partent, en effet, d'entités élémentaires, et agissent par construction d'assemblages, d'assemblages d'assemblages... selon la description du langage réel des mathématiques présentée par [Godement 66]. Le sens et les propriétés d'une formule sont dérivés de celui de ses composants par une fonction d'assemblage. Dans le cas d'une intégrale, il suffit de connaître sa fonction, ses bornes, et la mesure. On arrive ainsi, en codant des objets terminaux et les opérations d'assemblage, à reconstituer les propriétés des formules composées. Les propriétés d'une formule sont ainsi une émanation des composants qui la constituent.

Nous proposons donc de regrouper tant que possible les informations et les traitement autour des concepts mathématiques. Certaines propriétés des formules sont entièrement prises en charges par les concepts : c'est le cas de l'assemblage, où les besoins de centralisation se réduisent à la présence de variables d'état. Parfois, des opérations plus uniformes ou relatives à de nombreux concepts, entraînent une centralisation. Un utilitaire est alors chargé de recueillir et traiter les informations. C'est par exemple le cas de l'analyse, pendant laquelle les concepts ne sont pas encore identifiés.

4.3.2. Organisation de la gestion interne

Les données internes nécessaires aux manipulations de formules sont ici scindées en fonction de leur persistance :

- base de connaissances pour les informations générales de la théorie.
- bases de faits pour les informations spécifiques à des sessions de résolution d'un problème.

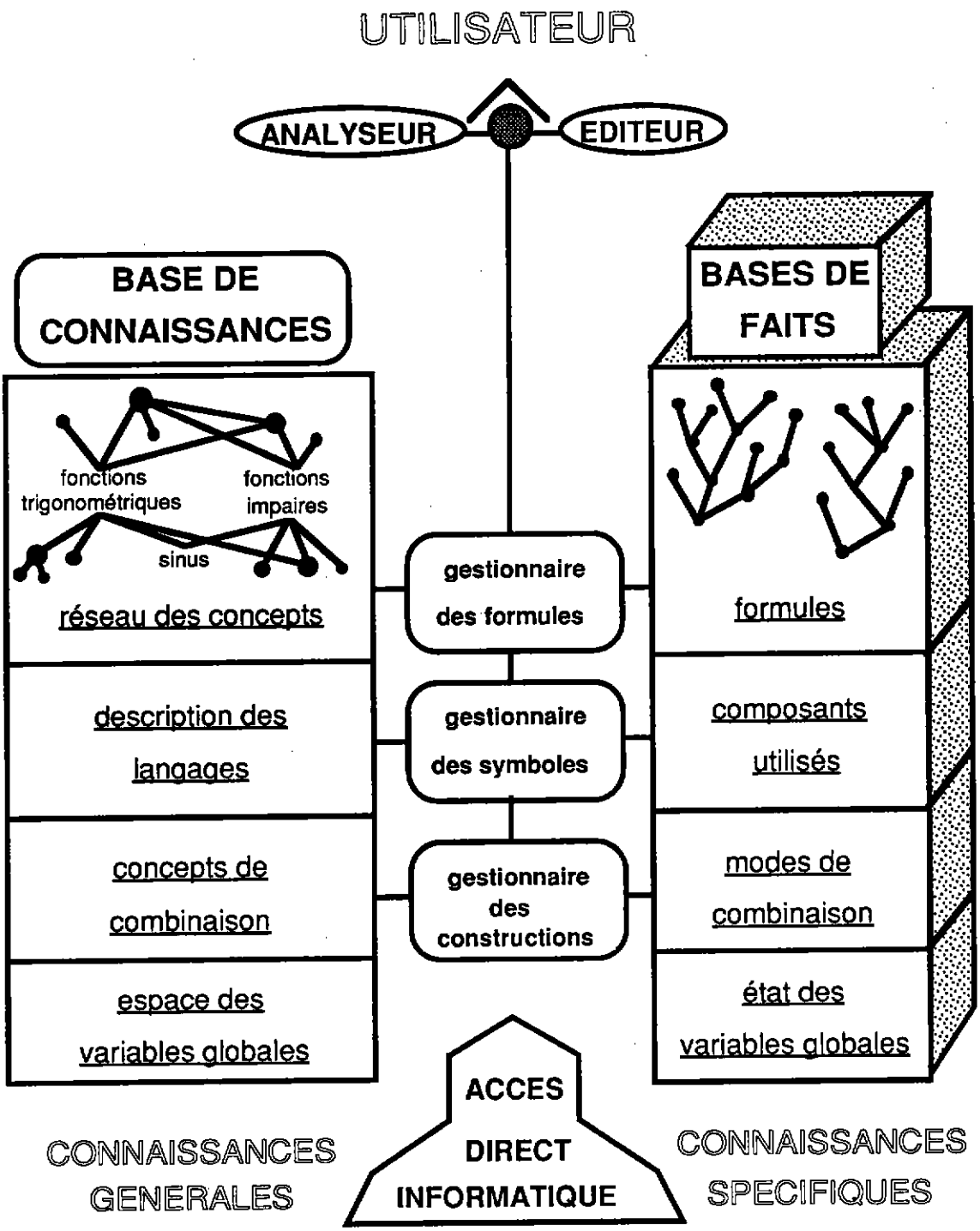


figure 3.28

Le schéma figure 3.28 détaille cette partition et présente les principaux utilitaires :

- L'espace des **variables globales** caractérise les choix possibles et leurs valeurs par défaut. Elles concernent des aspects syntaxiques, sémantiques ou simplement informatiques. Leur état forme l'environnement de validité du contenu de la base de faits.
- Le **gestionnaire des constructions** s'occupe des manipulations de formules en exploitant les concepts de combinaison : ajout, édition, substitution, application d'un opérateur, etc. Il mémorise les manipulations effectuées grâce aux modes de combinaison choisis, gère l'historique des manipulations, regroupe les opérations de construction de preuves, mémorise les transformations réutilisables, etc.
- Le **gestionnaire des symboles** utilise la description des langages pour effectuer, par exemple, la génération d'un ou de cent nouveaux symboles de variables en respectant la compatibilité pragmatique du choix des symboles. Il gère les abstractions lexicales, détecte les conflits, gère la cohérence des structures textuelles choisies, etc.
- Le **gestionnaire des formules** fait le lien entre le squelette des références, les composants et le réseau de concepts. La description d'un composant permet par exemple d'aller chercher ses propriétés ou de vérifier que la formule est mathématiquement bien formée. Elle permet aussi d'accéder aux emplacements où il intervient, dans une formule ou dans un segment de preuve.

Bien qu'un accès informatique direct soit prévu pour l'édition de code, l'usager communique avec le système avec une interface mathématique centrée autour de l'analyseur et de l'éditeur. Ce sont eux, en effet, qui assurent les fonctions de compréhension et de manipulation. D'autres utilitaires leur sont subordonnés, comme un gestionnaire de messages ou un utilitaire de désignation.

4.3.3. Constitution de la représentation interne des formules

La forme du Langage Mathématique joue un rôle important dans la compréhension de son contenu. Comme les formes possibles ne sont pas normalisées, la Représentation Interne du langage doit donc coder ces **informations textuelles**, afin de les restituer. La Représentation Interne des formules consiste à représenter la structure textuelle et la structure mathématique des formules.

L'organisation que nous proposons pour les formules présente des similarités avec les travaux de compréhension du Langage Naturel dans la lignée de [Lehnert 83]. Elle cherche à respecter les deux axiomes fondamentaux de la théorie des dépendances conceptuelles [Schank 77], concernant la représentation du sens d'une phrase :

- Deux phrases ayant un sens identique, indépendamment du langage, ont une même représentation ;
- Toute l'information implicite d'une phrase doit être explicitée dans la représentation de son sens.

La structure mathématique correspond à un ***Graphe Mathématique d'une Formule*** (GMF) constitué du squelette des références et des composants. La structure textuelle se prêterait aussi à une telle représentation avec un Graphe Textuel d'une Formule. Toutefois, la proximité de ces deux graphes et le couplage nécessaire à leur exploitation nous fait préférer la représentation de ce graphe textuel sous forme d'une décoration du graphe mathématique, appelée ***Décoration Textuelle d'une Formule*** (DTF).

Cette option de décoration nous permet de relier les informations textuelles d'une formule jusqu'aux concepts mathématiques correspondants. Conformément à notre approche à objets, nous pouvons ainsi **regrouper les informations textuelles générales autour des concepts**. Par exemple, la sommation regroupe les structures textuelles possibles d'un Σ. La Décoration Textuelle d'une Formule peut alors se contenter aussi d'une référence vers cette description. Elle désigne par exemple l'un des assemblages présentés ci-après :

$$x_1 + x_2 + \ldots + x_n \qquad \sum_{k \in \{1,\ldots,n\}} \qquad \sum_{1 \le k \le n} \qquad \sum_{k=1}^{n} \qquad \sum_{k} \qquad \sum$$

Ainsi, le Graphe Mathématique d'une Formule matérialise la relation mathématique et permet l'accès aux informations pertinentes pour le traitement mathématique. La Décoration Textuelle d'une Formule regroupe les informations textuellement pertinentes et préfigure l'apparence visuelle. Il est aussi possible de rajouter d'autres décorations au GMF : par exemple pour les nœuds du squelette de référence susceptibles d'être complétés, les décorer par une interface synthétisant les attributs utiles à l'analyseur.

Un langage mathématique "concret" a pour avantage de combiner ces deux sortes d'informations : mathématique et textuelle. Son inconvénient est que ces informations doivent être abstraites afin de pouvoir assister le travail du mathématicien. **L'acquisition et la construction informatique** de ces informations (GMF et DTF) peut alors s'effectuer selon deux approches : abstraite ou concrète.

L'approche abstraite consiste à construire directement les structures informatiques abstraites nécessaires aux traitements. Elle s'apparente moins à l'analyse qu'à la génération, puisqu'elle construit les structures informatiques nécessaires à la génération du langage concret. Il s'agirait de construire le Graphe Mathématique d'une Formule puis de déterminer les informations de la Décoration Textuelle de la Formule. D'après notre option de décoration, le choix de ces informations textuelles résulterait d'une **expertise de présentation textuelle** rattachée aux concepts définissant les opérateurs de la formule. L'homogénéité générale de ces choix locaux pourrait être coordonnée par un **style** de présentation textuelle. Cette approche est principalement intéressante lorsque les critères de choix syntaxiques sont bien maitrisés.

L'approche concrète permet de travailler à la source, avant que l'information introduite ne soit éclatée, complétée et organisée par le système. L'information soumise en entrée est analysée pour extraire le Graphe Mathématique de la Formule, ainsi que sa Décoration Textuelle. Cette approche décrit l'assemblage des symboles "$\sin^2 x_k(t)$" plutôt que sa structure mathématique (le carré du sinus...). Elle évite une construction point à point de cette structure mathématique, à l'aide de menus par exemple. Cette approche concrète est souvent préférable à l'approche abstraite, puisque plus concise et plus expressive.

La méthode usuellement préférée dans les environnements de programmation cherche à combiner ces deux approches. La saisie est alors guidée par le Graphe Mathématique de la Formule, mais pour éviter d'être trop contraignante, elle a recours à la saisie de sous-formules en langage concret. La prise en compte d'informations syntaxiques et sémantiques dans PSG [Bahlke 86] permet de filtrer les menus en retirant toutes les productions amenant à des arbres incorrects. Cette **approche mixte** intègre saisie et analyse dans un processus d'édition de formules.

La cohérence permanente des formules imposée par cette approche mixte est une entrave aux manipulations textuelles. C'est pourquoi l'éditeur doit autoriser la manipulation de **formules incorrectes**. Cette technique est utilisée dans GEPI [Canals 88] pour la programmation. Elle consiste à associer une structure apparente aux nœuds structurellement douteux, mettre la zone correspondante en relief, et différer l'analyse jusqu'à la demande de l'utilisateur.

4.3.4. La liberté de choix des structures

L'activité mathématique se définit différemment selon les besoins d'exploitation des résultats. Nous avons vu partie I dans l'exemple des groupes, que l'égalité de l'inverse à droite et à gauche se formule différemment selon l'exploitation qui en est faite. De même, la distinction mathématique entre x(yz) et (xy)z n'est pas toujours intéressante[5]. Elle est pertinente lorsque le mathématicien cherche une analyse très fine, ou des opérations très pauvres. Dans les manipulations courantes, il est en revanche souhaitable de travailler modulo l'associativité et la commutativité.

Dans le premier cas, la structure mathématique doit distinguer ces formules, dans le second cas ce n'est pas souhaitable. Ainsi, en fonction des besoins, la structure mathématique des formules est définie différemment. Un Atelier Mathématique Intégré ne peut adopter a priori une normalisation des termes.

Le fait de traiter la commutativité et l'associativité comme négligeables pour les déductions mathématiques n'implique pas que toutes les permutations syntaxiques soient équivalentes. Nous avons vu en effet que le groupement des termes est une information importante dans la production et le suivi

---

[5] Pour n+1 termes, il y a d'ailleurs $\frac{1}{n+1}\binom{2n}{n}$ façons de les parenthéser !

d'étapes de calcul. La structure textuelle permet de faire le lien entre la structure apparente et une structure mathématique ignorant ces préoccupations. La **gestion des permutations** se fera donc grâce aux informations de la structure textuelle. Cela permet notamment de distinguer localement entre x(yz) et (xy)z lorsque leur structure mathématique est identique.

L'utilisation d'un Atelier Mathématique Intégré doit alors **définir les trois structures en fonction de l'application traitée.** La marge de manœuvre dépend de l'expressivité des langages de description disponibles. Les choix effectués dépendent du domaine d'application mathématique, de la nature automatique ou assistée de la tâche, de l'importance relative de la production sur la vérification, etc.

C'est d'ailleurs ainsi que peut s'évaluer l'utilisabilité des systèmes mathématiques : une ouverture et des facilités de description même pour des caractéristiques très fines comme les formes normales des formules ou le polymorphisme.

Le polymorphisme est intuitivement la propriété d'un objet d'appartenir à plusieurs classes. Ce lien objet-classe définit un *typage* de l'objet par sa classe. Le polymorphisme présuppose un système qui définisse les objets, les classes, et un procédé de correspondance entre les objets et les classes. Le polymorphisme est alors la propriété d'un opérateur ou d'un langage d'accepter des objets non univoques (à correspondance multiple).

Les différentes sortes de polymorphisme sont décrites dans [Cardelli 85]. Le polymorphisme a souvent un rôle très positif puisqu'il permet la généricité, c'est-à-dire l'utilisation d'un Modèle commun. Par exemple, cela est très exploité en Algèbre, où le jeu entre interprétations d'une expression facilite la résolution.

La notion d'objet peut être définie de part et d'autre de la ligne de démarcation du langage. Le langage peut définir un typage apparent en fonction de conventions syntaxiques. De même l'interprétation de certaines relations peut être définie sur plusieurs classes.

Le polymorphisme est ainsi un choix langagier qui dépend des conventions de syntaxe choisies (typage apparent), du Modèle actuel (typage intrinsèque), et des règles d'interprétation adoptées (conversion, demande d'information ou erreur). Ces choix sont différents selon le type d'application cherchée, et le mathématicien doit rester libre de les contrôler.

Une discussion de l'exemple "2 + 3,0" est intéressante pour étayer cette liberté de choix, et déterminer des paramètres de guidage de ces choix :

- — Un **typage apparent faible** est utile pour la lisibilité et la concision. Il prévaut en mathématiques classique lorsque la distinction entre les relations d'addition n'est pas intéressante. Le mathématicien écrit alors "2+3", et interprète l'addition et son résultat par le type minimal.
- — Un **typage apparent fort** permet d'indiquer des propriétés pertinentes au problème traité. Ainsi, lorsque le Modèle sous-jacent distingue les entiers des réels, il est nécessaire de trouver un procédé syntaxique pour les différencier. Ce typage apparent s'accompagne souvent d'une information implicite, "3,0" indiquant par exemple une précision absolue approximative de 10^{-1}.
- — Un **typage intrinsèque fort** peut être effectué par souci de rigueur mathématique, comme par contrainte de manipulation ou de constructibilité. Lorsqu'il y a de nombreuses sortes d'addition (comme dans Z/nZ), il faut expliciter le sens. Cela se fait au niveau du langage par une règle globale (telle des déclarations) ou par une notation plus explicite ("$+_n$").
- — Un **typage intrinsèque faible** favorise l'expressivité des mathématiques. Tout typage est en effet lié à une notion d'utilité. Lorsqu'il en a besoin, l'analyste type ses voisinages en $]x \pm 1/n[$ ou le numéricien type "selon" Z/nZ. Le typage est ainsi une organisation donnée a posteriori, et qu'il est délicat de décréter a priori.

La variabilité des choix pertinents pour un thème aussi spécifique que le typage illustre l'intérêt de pouvoir paramétrer, en fonction des applications visées, les choix de définition des structures. Nous retombons ainsi sur le problème non résolu d'une description grammaticale élégante des expressions mathématiques.

4.4. LES MECANISMES DE GENERATION

Nous envisageons un mécanisme de génération **multi-langage et avec paraphrase.** La faculté d'employer des paraphrases (plusieurs tournures syntaxiques possibles pour un "sens" identique) revient à exploiter la diversité d'expression des mathématiques. Cette finesse de modélisation n'est guère possible que dans un Atelier Mathématique Intégré.

L'aspect multi-langage répond à des critères mathématiques et informatiques :

- mathématiques, concernant tant les langages de Phrases et Formules :
  Les mathématiques étant internationales, les langages de Phrases doivent traiter différentes langues. Par ailleurs, la diversité des choix possibles dans les langages de Formules est équivalente à l'utilisation de langages différents.
- informatiques :
  l'introduction d'un système revient à multiplier les utilitaires, utilisant chacun un langage spécifique à son traitement. Pour ces utilitaires, une variation des caractéristiques langagières externes revient aussi à un changement de langage. De plus, un système informatique doit pouvoir communiquer avec d'autres systèmes qui utilisent des langages définis par ailleurs.

Voici, classé par niveau d'abstraction, un éventail de langages adaptés aux manipulations de formules :

- Le langage d'un "inspecteur" des structures informatiques de la Représentation Interne ;
- Un langage d'édition chargé de construire ces structures. C'est celui utilisé par MACSYMA, pour produire des formules "(sigma, k, 1, n , %)" ("%" réfère à la construction antérieure) ;
- Un langage informatique comme T_EX "\sum ↓ {k=1} ↑ n \sinus..." ;
- Un langage interactif "semi-structuré"comme celui d'EDIMATH, qui utilise des boîtes ;
- Un langage "étendu" acceptant les trois structures des formules ;
- Un langage de Phrases pour décrire la formule, comme la somme pour k égale 1 à n de etc.

Comme dans tout système informatique sophistiqué, il y a ainsi une véritable Tour de Babel dans un Atelier Mathématique Intégré. C'est uniquement un point de vue conceptuel externe à ces langages qui permet d'envisager une Représentation Interne partagée. Cette diversité rend indispensable le recours à des outils multi-langage dès que le système présente des fonctionnalités variées.

Pour gérer cet aspect multi-langage, les environnements de programmation génériques ont été conçus avec des **méta-outils.** Ils s'adaptent ainsi à plusieurs langages grâce à un simple changement des descriptions utilisées. La génération d'un éditeur est effectuée par couplage d'une description du langage à éditer à un noyau de manipulation d'arbres. Nous distinguons parmi ces derniers l'approche d'interprétation de DOSE [Kaiser 88], qui permet de ne pas avoir à générer un éditeur par langage. Il interprète directement la nouvelle description et permet un véritable multi-langage, en pouvant notamment étendre rapidement un langage.

Notre objectif pour les mathématiques est qu'un langage soit défini en première approximation par ses opérateurs mathématiques : ce sont eux et leurs propriétés qui créent le langage. Chaque opérateur comporte alors sa description dans les divers langages définis, et permet d'accéder aux propriétés mathématiques le concernant. La base de connaissances est alors **multi-langage.**

Pour ce qui concerne le choix de représentation des formules, le GMF (graphe mathématique des formules) se charge des abstractions mathématiques. A l'extrême, les GMF de "sin(0)", "1-1" et "0" peuvent être les mêmes si un calcul mathématique (c.-à-d. un mécanisme de simplification) leur impose une forme normale. La DTF (décoration textuelle) serait alors chargée de représenter ces trois différentes formes, afin que l'usager puisse travailler effectivement à partir de la forme affichée. La DTF consiste plus généralement à conserver l'information syntaxique pertinente.

La Décoration Textuelle des Formules regroupe les **caractéristiques liées au langage.** Elle distingue, par exemple, les caractéristiques liées à l'usage du nombre "3", selon sa base d'expression ("10" en base 3), le mode d'écriture ("III" en chiffres romains ou "three" en anglais), voire selon le typage dans le problème ($\vec{3}$ s'il s'agit d'un vecteur, 3° pour un angle, etc.). Certains formalismes informatiques ont des possibilités syntaxiques très limitées : par exemple, seule les informations concernant la DTF des symboles leur sont utiles. L'intérêt d'une représentation multi-langage ne s'en trouve pas réduit pour autant : chaque langage dispose des informations qui l'intéresse.

Nous pouvons alors caractériser la génération des informations relatives à une formule visuelle par les étapes identifiées partie II :

- un résultat interne au système issu d'une opération ;
- des actions de mise sous forme normale interne qui contrôlent les simplifications et produisent la GMF ;
- un choix mathématique de nommage et de notation déterminant la DTF candidate ;
- des contraintes informatiques agissant sur la présentation effective pour holophraster certaines DTF ou proposer des DTF mieux adaptées à un encombrement important.

Inversement, une même structure apparente $\sum_{i \in I} \sin^{i}(a_i)$ pourrait donner lieu à deux GMF différentes selon leur mode de construction. En effet la construction peut prendre "I" comme dernier argument, ou alors prendre comme dernier argument le terme principal de la sommation. Une telle distinction est par exemple pertinente en λ-calcul.

Le choix que nous proposons pour la GMF est que deux structures dont la construction est indiscernable dans la théorie en cours, ont une même représentation quelle que soit leur construction. C'est le cas du "Σ" qui peut être vu comme l'application d'un opérateur à une suite (conceptuellement agréable), ou comme application d'un autre opérateur à l'élément générique constituant le terme principal du "Σ" (utilisé en pratique). De même, il y a abstraction de la définition du domaine mathématique "I". Ce choix de GMF correspond à l'usage qui envisage une construction comme "sin(x)" en tant que valeur ou que fonction, selon son utilisation mathématique englobante.

Les mécanismes de génération multi-langages que nous proposons se décrivent à deux niveaux d'abstraction différents. Pour le mathématicien, il s'agit de définir et faire correspondre la structure textuelle avec la structure mathématique. C'est ce que nous étudions section suivante dans le cas de la sommation. Pour le concepteur du système, il s'agit d'envisager les correspondances multi-langages informatiques. La syntaxe du langage utilisé par le mathématicien étant informatiquement la plus complexe, son traitement est prioritaire sur celui des autres langages.

Nous proposons pour cela d'utiliser un **langage intermédiaire pivot** entre la Représentation Interne des formules (GMF et DTF) et les multi-langages. Il permet de considérer la DTF comme un codage et de définir la formule comme un graphe incluant aussi du "sucre syntaxique" abstrait. Par rapport aux multi-langages, il constituerait une plateforme commune permettant l'abstraction de leurs caractéristiques lexicales. Ce langage intermédiaire pivot regrouperait les fonctions de structuration textuelles communes aux mathématiques, en autorisant aussi des structures incomplètes. Il pourrait être mis en correspondance simple avec une Représentation de Boîtes.

A chaque DTF serait associée un schéma équivalent dans ce langage intermédiaire pivot, à condition que les structures proposées soient compatibles avec la syntaxe concrète du langage externe final. Cette transformation permettrait de rajouter le "sucre syntaxique" et de positionner convenablement les arguments (inversions, dédoublements...). Enfin, la transformation dans le langage externe final consisterait à associer aux séparateurs syntaxiques leur forme lexicale dans le langage. Ainsi, un couple serait "indent-de-couple-optionnel sép-nuple-gch arg1 sép-nuple-centre arg2 sép-nuple-drt" en langage intermédiaire, mais "(x,y)", "{x;y}", "\couple(x;y)", etc, selon les critères lexicaux du langage externe final. L'introduction de la technique du langage pivot est parfois utilisée pour faciliter la traduction du Langage Naturel entre langues multiples.

4.5. LE CAS DE LA SOMMATION

4.5.1. Les formes de structure textuelle

Ces dernières sections présentent successivement les connaissances langagières à mettre en œuvre pour le cas de la sommation. Nous avons effectué une étude d'exemples d'utilisation de "Σ", présentés en Annexes. Les connaissances rattachées à la sommation comprennent :

- les aspects mathématiques, que nous ne détaillerons pas :
  définition et son interprétation, cas particuliers, description synthétique (opération sur les suites, addition généralisée, etc.), opérations de manipulation (addition de deux Σ, changement de variable, etc.), exploitation pour certains types de problèmes, etc.
- les aspects langagiers qui regroupent la description des formes possibles de :
  - la structure textuelle ;
  - la structure mathématique ;
  - les fonctions de correspondance entre structures mathématique et textuelle ;

- les critères de choix de ces deux structures dans une situation d'utilisation donnée.

Les fonctions de correspondance et les critères de choix précédents constituent des connaissances du mathématicien liées à sa pragmatique d'usage du langage.

Nous avons ainsi recensé, dans l'ouvrage [Godement 66], huit **formes de structure textuelle**, dans le cas où la variable d'index est simple :

$$\sum_{i \in I} a_i \quad .....(1)$$ (p.107)

$$\sum_{i=1}^{n} a_i \quad .....(2)$$ (p.108)

$$\sum_{1 \le i \le n} a_i \quad .....(3)$$ (p.108)

$$\sum_{i} a_i \quad .....(4)$$ (p.275)

$$\sum a_i \quad .....(5)$$ (p.275)

$$\sum_{\substack{i \in I \\ P(i) \\ Q(i)}} a_i \quad .....(6)$$ (p.560)

$$\sum_{\substack{P(i) \\ Q(i)}} a_i \quad .....(7)$$ (p.607)

$$\sum_{i=0}^{\infty} a_i X^i \quad .....(8)$$ (p.573)

Notons que ces formes de structure textuelle prennent en compte des aspects syntaxiques détaillés : un éditeur classique de formules ne dispose que d'une structure textuelle, qui lui permet de générer ces formes ainsi que toutes les autres ayant un "Σ". Un éditeur chargé de reconstituer le sens mathématique devra par contre les distinguer. Par exemple, il doit savoir traiter différemment les formes n'ayant qu'une borne inférieure : (1), (3), (4), (6) et (7).

Les structures textuelles diffèrent selon l'organisation et la nature de l'information qui les caractérise. C'est pour cela que nous distinguons entre (6) et (7). Par contre, certains aspects relatifs à la composition (formatage de mise en forme, analogue au choix des césures inter ou intra-mots pour former les lignes d'un paragraphe) sont abstraits. Ainsi dans (7), la superposition "P(i)" "Q(i)" peut se juxtaposer en se séparant par une virgule, ou ne comporter qu'une seule relation.

Chaque structure textuelle correspond de plus à un ensemble de situations. Nous avons repris la définition initiale pour (2), mais le mathématicien l'étend rapidement à des bornes quelconques "p" et "q" (variables ou expressions mathématiques). De même, (3) est étendue en inégalités strictes ou pas. Par ailleurs, la répartition des pages montre que ces introductions ne se font pas a priori mais en fonction de l'émergence des besoins.

La description et l'organisation des structures introduites sont ainsi à la discrétion du mathématicien. Il convient donc, lors de l'introduction d'une structure textuelle, de prévoir une utilisation aussi générale que possible, sachant que le mathématicien sera probablement amené à étendre les structures introduites dans leur définition propre ou par introduction d'une nouvelle structure.

S'il est possible de fournir un jeu de structures textuelles satisfaisant pour les opérateurs classiques, les situations mathématiques font que l'éventualité d'une extension est toujours possible. Par exemple, nous

avons altéré ici la forme (2) de R. Godement, qui préfère : $\sum_{i=1}^{i=n} a_i$. Dans l'ouvrage [Pac 86], la variable d'index de (4) est écrite comme elle le serait dans A_i : Σ_i. Enfin, pour (6) et (7), [Graham 89] insère les prédicats dans l'expression indicée (cf. Annexes). La diversité est encore plus étendue pour les indices multiples... Par ailleurs, les connaissances pragmatiques relatives aux "Σ" varient elles aussi. Enfin, ces formes peuvent être complétées par la description des formes possibles de description d'une sommation en utilisant les "...".

Il convient donc de disposer de moyens de description rapides et aisés pour prendre en compte cette diversité.

4.5.2. Structures mathématiques pour la sommation

L'examen de ces formes de structure textuelle nous incite à introduire plusieurs structures mathématiques :

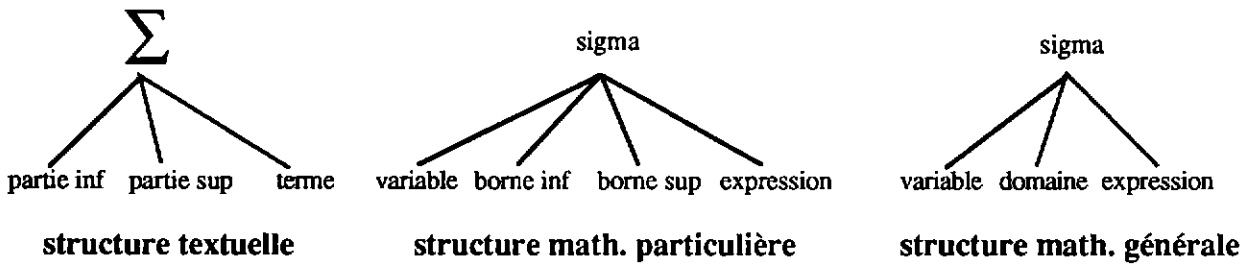


figure 3.29

Dans le cas d'une sommation générale, l'opérateur "sigma" qui intervient dans la structure mathématique a trois arguments : la variable, le domaine de variation, et l'expression indicée (figure 3.29). Le concept de sommation contient la description de cet opérateur : dans cette définition, la description de "domaine" correspond à la **description générale** définissant les expressions mathématiques de type ensembliste.

Toutefois, les formes très utilisées de la structure textuelle, (2) et (3), ne sont pas applicables à partir de cette description de la structure mathématique générale. Elles requièrent que le domaine soit un intervalle entier, et seront éventuellement généralisées aux intervalles sur des domaines munis d'une relation d'ordre. Le "domaine" est alors défini par ses bornes inférieures et supérieures.

Nous sommes face à une alternative pour prendre en compte cette situation :

- expliciter ces informations dans la structure mathématique, en imposant que pour les formes (2) et (3) le domaine soit explicitement construit comme "(intervalle-entier inf sup)" avec l'opérateur "intervalle-entier" lui-même défini comme "(intervalle ens ordre)" avec pour ensemble les entiers naturels munis de son ordre usuel.
- introduire une autre structure mathématique, l'équivalence étant gérée par l'exploitation de l'organisation conceptuelle.

Nous préférons cette deuxième solution. La première introduit explicitement de nombreuses opérations servant d'abstraction pour éviter une mise à plat de toutes les informations nécessaires. De plus, des fonctions souvent artificielles doivent être introduite pour chaque usage. Par ailleurs, cela perturbe le mathématicien qui ignore, de l'extérieur, devoir manipuler explicitement une structure aussi complexe dans sa formule.

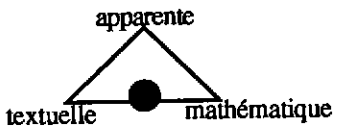
La seconde solution revient à disposer de deux structures mathématiques au choix. Le mathématicien sait que, lorsque les conditions le permettent, un intervalle est un domaine qui peut être défini par ses bornes. Cette caractéristique de l'organisation conceptuelle est utilisée pour former une nouvelle structure mathématique (particulière), qui peut être transformée en structure mathématique générale en utilisant la définition de "intervalle" comme sous-classe de "domaine".

Le passage réciproque (du général : domaine, au particulier : intervalle) exploite les conditions de définition cohérente d'un intervalle. La structure mathématique particulière ne sera utilisée que si elles sont vérifiées. C'est, ici encore, un rôle important de l'organisation conceptuelle et de l'exploitation des connaissances. La définition conceptuelle d'un intervalle peut même être exploitée plus avant,

pour calculer par exemple les bornes et les conditions de définition de l'intervalle défini dans (3) par "1<i<2n-1" au lieu de "2≤i≤2n-2 .

Cette deuxième solution explicite donc "borne inf ; borne sup" comme un point de vue définissant un "domaine". Elle permet alors de **représenter dans les formules uniquement les informations mathématiques explicites**, en laissant à l'organisation conceptuelle la charge de prendre en compte les informations orthogonales, relatives au contexte de manipulation du problème et à sa cohérence. Notons enfin que d'autres structures mathématiques peuvent être définies, pour prendre en compte par exemple la forme (8) ou les séries.

4.5.3. Les correspondances entre structures



La description des fonctions de correspondance entre structure mathématique et structure textuelle cloture ce chapitre plus spécifiquement dédié à la base du triangle. Cette description s'effectue en deux temps :

- point à point :
  une fois la structure textuelle et la structure mathématique toutes deux définies, l'usager décrit les relations entre ces deux structures. Il les complète (avec l'assistance du système) par les conditions et les moyens d'expliciter les informations implicites manquantes.
- globalement :
  les combinaisons "mathématique-textuelle" possibles sont caractérisées par les conditions pragmatiques, liées à la situation, qui permettent d'élire le "meilleur" choix.

Pour développer la description point à point, la forme de structure textuelle (4) : $\sum_i a_i$, peut être associée aux deux structures mathématiques, mais dans ces deux cas le domaine doit être explicité pour que la correspondance soit totale. Le méta-principe général est qu'une omission ne peut être effectuée que si elle est évidente. Il s'agit alors de déterminer **quand elle est évidente et comment la lever**. Ce type de connaissance pragmatique ne peut être totalement défini a priori, et doit disposer d'un langage de description simple afin d'être redéfini ou précisé ultérieurement.

Le domaine mathématique est par exemple évident lorsque :
- l'expression indicée "a_i" a un domaine mathématique évident (par ex. il a été déclaré, ou alors il n'a pas d'importance, il suffit de savoir qu'il existe) ;
- la variable d'index "i" a un domaine mathématique évident (par ex. un ensemble d'indexation a été statué dans le problème) ;
- la séquence de déduction rend le domaine évident (par ex. il est précisé lors de l'introduction du premier "Σ", puis omis lors des opérations ultérieures préservant le domaine), etc.

Dans chacun de ces cas, il faut aussi décrire comment accéder et expliciter ce "domaine évident". Nous disposons alors d'informations utiles tant en analyse qu'en génération. Pour l'analyse, ces informations suffisent à construire la structure mathématique à partir d'une structure textuelle. Le problème résiduel consiste à identifier la structure textuelle à partir des données textuelles saisies incrémentalement. Cela implique par exemple de discriminer les formes dès la saisie de la borne inférieure et d'effectuer déjà une construction incomplète de la structure mathématique. Cela peut être envisagé par une **expertise d'analyse[6] de la structure textuelle.**

La stratégie minimale, pour ce qui concerne la génération, consiste à adopter la structure textuelle présente en entrée et choisie par le mathématicien. Cela a l'avantage de la simplicité mais présente des limitations. Ainsi, cette stratégie ne convient pas lorsque l'activité mathématique génère des structures mathématiques nouvelles, ou lorsque la structure textuelle adaptée n'est pas exprimée (par facilité de saisie ou rigidité du langage d'expression). Il est donc nécessaire de pouvoir **déterminer une structure textuelle adaptée à la situation**, la structure mathématique étant supposée connue.

[6] Il s'agit ici de capacités d'analyse d'un problème, et non de l'analyse des formules pour produire une Représentation Interne. Dans certains cas, la proximité discursive nécessite des précisions pour faciliter l'interprétation du langage. Cela pourrait être un autre méta-principe à inclure.

Les informations précédentes doivent alors être complétées par des informations de nature décisionnelle permettant de choisir entre plusieurs formes candidates, voire de décider s'il est utile de déclarer globalement le domaine afin d'utiliser la forme abrégée (4). L'essentiel est toutefois déjà disponible !

De nombreuses connaissances pragmatiques concernant l'usage du "Σ" n'ont pas été abordées ici. Par exemple, lorsque le terme indicé est aussi un "Σ", on retombe sur la problématique des Σ multiples. De même, la propagation des symboles de variables d'index, a priori muettes, fait l'objet d'une gestion simple mais attentive. Enfin, des critères proviennent de l'usage mathématique, comme l'introduction de termes nuls pour clarifier l'expression de la somme [Graham 89, p 24], ou l'intérêt de la forme (3) pour faciliter l'expression des changements de variables [ibid. p23] (cf. Annexes).

La mise à disposition des structures textuelles et mathématiques permet l'acquisition ou la génération de règles "expertes" les concernant. Par exemple, la transformation de la forme (3) en la structure mathématique particulière correspondante peut s'exprimer : Si la partie sup du Σ est vide, et si la partie inf est de la forme $p \leq i \leq r$, alors la variable est i, la borne inf est p et la borne sup est r. Des descriptions similaires peuvent être utilisées pour les déductions mathématiques : Si l'expression est indépendante de la variable, alors elle s'évalue en "expression fois nombre-de-termes", avec dans ce cas nombre-de-termes égal à "borne_inf moins borne_sup plus un". Tout cela est toutefois mieux décrit avec des facilités d'expression plus "mathématiques" que nous avons proposées.

4.6. BILAN SUR LE TRAITEMENT DU LANGAGE

Nous avons vu la nature et la diversité des connaissances employées pour la sommation, et nécessaires à une prise en compte coordonnée de la forme et du contenu mathématique des formules. Il en résulte que la solution proposée n'est pas une spécification formelle de ce qui est utilisé dans la pratique mathématique. M. Bouhier note d'ailleurs, en conclusion de son rapport sur la sémantique des formules [Bouhier 89, p 16] :

> On peut toujours écrire un nombre, même très important, de règles définissant une façon de générer le sens des formules éditées par un traitement de texte, puis programmer ces règles et compiler le programme résultant. L'unique personne qui risque (et encore !) d'être satisfaite du résultat sera le créateur de la grammaire. (...)
>
> La seule solution satisfaisante consiste à créer un système extensible par l'utilisateur. C'est-à-dire un système où chaque individu aurait la possibilité de faire prendre en compte ses habitudes et ses notations.

Nous partageons cette analyse d'un chercheur qui a abordé des problèmes identiques, dans le cadre d'un système à finalité plus immédiate. Si elle se veut réaliste, une solution non restreinte doit se chercher au méta-niveau, dans la recherche de procédés de description des connaissances nécessaires.

Par ailleurs, le lecteur averti aura noté la complexité inhérente à tout système de traitement du Langage Naturel, et noté les similitudes entre notre approche des langages de Formules et les solutions concernant les langages de Phrases.

Le cas du Σ, des diverses notations et de leur traitement est symptomatique de l'activité mathématique et des connaissances requises. Un AMI doit être relativement autonome en incluant par exemple des capacités d'apprentissage lors de l'acquisition des connaissances.

Le choix critique n'est pas alors "l'acquisition des connaissances", mais plutôt la réflexion sur la **dualité entre acquisition et traitement des connaissances**. Nous avons donc, pour le traitement du langage dans le cadre d'un AMI, soulevé et étudié des problèmes importants, et esquissé des solutions paradigmatiques. Nous continuerons de la sorte cette partie III concernant la **recherche de techniques informatiques adéquates pour notre problème.**

Les solutions que nous avons proposées s'appuyent sur une organisation conceptuelle, et sur l'émergence des langages de programmation logique avec contraintes. Nous avons dégagé la nécessité :

- de langages de description des connaissances à partir d'une présentation proche du mathématicien (ergonomie de la description) ;

- de langages d'expression de la grammaire des expressions ainsi que des structures mathématique et textuelle avec leur correspondance ;
- d'un (méta-)langage informatique permettant de décrire des tâches, de gérer des contraintes, et de faire appel à des sources d'information comme l'usager ou des connaissances majoritairement déclaratives.

Les connaissances sur la sommation se sont pas forcément complètes et peuvent même être incohérentes dans certaines situations difficiles à détecter. C'est l'**avantage épistémologique d'une approche d'assistance** qui permet de rendre compte, en souplesse et par l'interaction, d'un plus grand nombre de situations ou de situations non automatisables autrement. En contrepartie, elle exige des moyens de description et de traitement sophistiqués mais automatisables.

5. ORGANISATION DES CONNAISSANCES

5.1. INTRODUCTION A LA MODELISATION DES CONNAISSANCES MATHEMATIQUES

5.1.1. La constitution de connaissances mathématiques

La constitution de connaissances mathématiques est un défi majeur pour l'informatisation de l'activité mathématique. De plus, l'organisation scientifique des connaissances est l'un des objectifs de la modélisation d'une théorie mathématique. Le problème serait-il quasiment résolu à l'aide d'une simple transcription informatique ?

Il n'en est rien ! Pour appréhender l'**ampleur du problème**, nous conseillons au lecteur de prendre les premières pages d'un traité de mathématiques et chercher à organiser les définitions, notations, théorèmes et propriétés qu'il propose. Les traités rassemblent la partie bien organisée des connaissances mathématiques. Si leur utilité ne fait pas de doute, la complexité de leur contenu nécessite un travail de recherche de longue haleine.

La partie I a mis en évidence les caractéristiques des connaissances mathématiques (ce terme inclut aussi des "métaconnaissances") :

- complexité des mécanismes d'utilisation et de constitution de ces connaissances ;
- diversité en nature, en quantité, en uniformité (cas particuliers) des connaissances utiles ;
- complexité organisationnelle : facettes nombreuses, hiérarchies et points de vue multiples, dépendances complexes incluant : agrégation, renommage, spécificités, défauts, changements de représentation, etc.

Les conditions nécessaires à l'émergence de connaissances modélisées sur un support informatique sont :

- en avoir besoin ;
- pouvoir les utiliser ;
- savoir les extraire.

Nous avons besoin de ces connaissances pour la réalisation des tâches élémentaires du mathématicien. Nous pouvons les utiliser, car reproduire l'activité mathématique est justement l'objectif de notre approche. Reste à savoir les extraire... Nous proposons de surmonter les aspects quantitatifs en les introduisant progressivement en fonction des besoins. Notre approche d'assistance permet, de plus, de limiter les connaissances nécessaires à une partie suffisamment accessible pour être modélisée. Le problème est alors essentiellement qualitatif :

comment aboutir à une modélisation adéquate ?

Nous abordons donc, dans ce chapitre, une vue générale des connaissances nécessaires à l'activité mathématique, et certains problème clés concernant leur organisation et leur utilisation. Cette étude relève alors des Sciences Cognitives pour les procédés d'investigation concernant ces connaissances, et de l'informatique pour les représentations envisagées. L'acquisition de connaissances ne se fera, à notre avis, que si elle est motivée par un besoin d'utilisation concrète d'un système informatique pour (et par) des mathématiciens.

Notre apport est, avant tout, dans les conséquences de notre approche d'assistance qui nous impose de rendre compte d'une utilisation des connaissances proche de celle du mathématicien. Nous proposons alors de :

- associer les structures mathématiques et les procédés langagiers ;
- donner un rôle privilégié aux concepts mathématiques ;
- séparer les connaissances générales des connaissances rattachables à un concept mathématique.

5.1.2. L'utilisation des connaissances

Les connaissances nécessaires à l'activité mathématique peuvent être organisées en fonction de leur rôle. Nous avons ainsi (partie I) distingué trois axes lors de la construction d'une preuve, ce qui donne six dimensions en différenciant les aspects mathématiques et les aspects langagiers (figure 3.30).

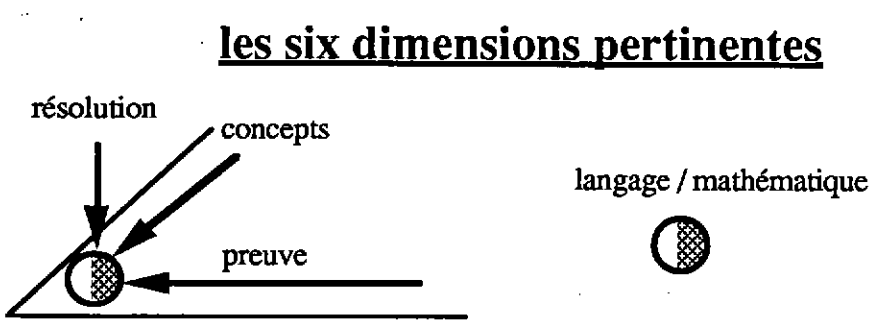


figure 3.30

Nous nous intéressons dans ce chapitre à l'axe des concepts. Cela comprend l'organisation des concepts permettant la constitution des formules, et l'organisation des propriétés mathématiques (règles, définitions, théorèmes, jugements, etc.) qui sont invoquées lors de la constitution d'une preuve. Nous étudions tant les aspects mathématiques que les aspects langagiers.

Cette modélisation des faits de base du domaine est souvent schématisée dans l'utilisation des connaissances mathématiques, vu le domaine restreint auxquels les systèmes s'intéressent. Nous ne disposons donc pas d'un bon modèle statique, en revanche, nous disposons d'un modèle dynamique permettant de produire des raisonnements sophistiqués (figure 3.31).

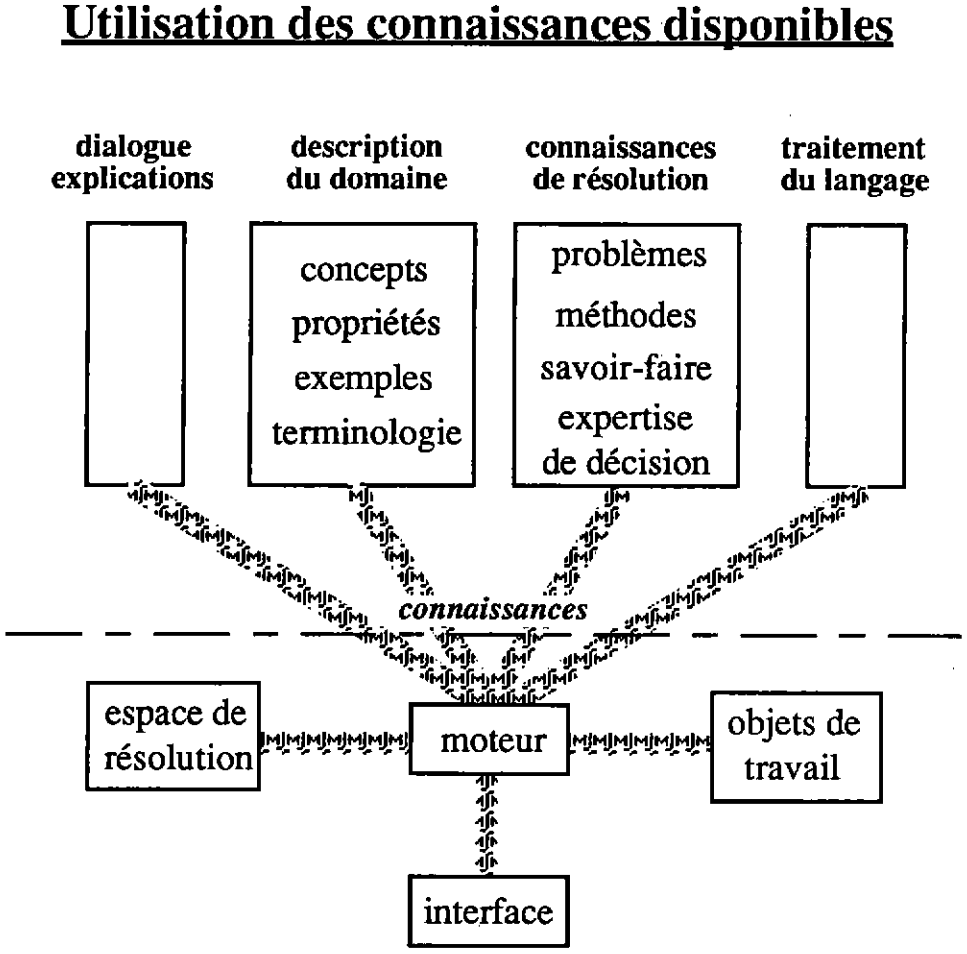


figure 3.31

Les quatre modules présentés figure 3.31 correspondent à un découpage des connaissances selon leur fonction. Selon l'application, ils sont complétés par d'autres modules de connaissances (enseignement, modèle de l'élève, assistance, etc.). Le module de résolution est classiquement composé de :

- **problèmes**, décrits et classifiés dès que le système est suffisamment vaste ;
- **méthodes** qui décrivent, souvent de façon déclarative, les moyens de résolution disponibles (généraux ou spécifiques à certains problèmes). Cela comprend le schéma de raisonnement par l'absurde, le plan de résolution d'une classe de problèmes, ou les moyens d'établir une relation de récurrence.
- **savoir-faire** (de calcul, d'analyse, d'utilisation etc.) qui regroupent classiquement les procédures que l'on ne peut pas, ou ne veut pas, représenter en tant que méthodes. Appliquer une identité, effectuer un changement de variable ou reconnaître une forme sont des exemples de savoir-faire.
- **expertise de décision** qui supervise l'utilisation des connaissances précédentes. Elle évalue les voies de recherche et guide le raisonnement par ses choix.

5.1.3. L'action du mathématicien sur les connaissances

La notion de méta-niveau correspond à un changement de représentation. Une représentation correspond à un **faisceau de points de vue statiques** sur un univers d'entités. Passer au méta-niveau consiste à considérer certain objets et relations de cette représentation statique. C'est donc d'abord une action dynamique, qui se traduit par la génération d'une autre représentation qui comporte une représentation des entités de la représentation initiale, avec éventuellement une trace réifiée de leur dynamique.

Selon l'orientation de l'analyse effectuée lors du passage au méta-niveau, l'analyste obtient :

- un méta-niveau à caractère **déductif** :
  chargé de représenter la dynamique de combinaison, d'évolution et de création d'entités de la représentation considérée. Il s'intéresse donc au raisonnement, à son contrôle et aux connaissances utiles pour effectuer des déductions respectant l'usage de la représentation initiale. Il s'analyse lui-aussi avec un méta-niveau à caractère déductif essentiel pour toute explication. Pour le mathématicien, cette hiérarchie de méta-niveaux est considérée comme intégrée à l'activité mathématique.
- un méta-niveau à caractère **descriptif** :
  chargé de fournir une description des informations utilisées dans la représentation initiale. Il modélise ainsi cette représentation. Un tel changement de représentation est obligatoirement adopté pour une transcription informatique ou une analyse conceptuelle de l'activité du mathématicien. Il est aussi adopté intuitivement par le mathématicien lors de son activité.
- un méta-niveau à caractère **langagier** :
  chargé d'utiliser un système langagier susceptible de permettre un traitement relativement autonome[1]. Des méta-niveaux partiels sont ainsi formalisés dans les mathématiques à l'aide d'un langage symbolique, qui constitue une représentation intermédiaire avec ses règles et ses conventions. Une telle représentation est dite opaque lorsqu'elle se substitue à la représentation initiale (plus intuitive), et permet d'obtenir par calcul des résultats équivalents à ceux pouvant être obtenus dans la représentation initiale. Souvent, la distinction n'est pas aussi claire et des éléments de la représentation initiale interviennent aussi dans la représentation au méta-niveau.

Le mathématicien effectue constamment des **changements d'interprétation** lors de l'utilisation. Par exemple, le mathématicien considère qu'un nombre complexe est isomorphe à une matrice et effectue un changement de point de vue (éventuellement temporaire) sur les nombres complexes. Il reste pourtant au sein d'une même représentation qu'il manipule en jouant sur les isomorphismes.

L'**abstraction** consiste à profiter d'une classification (ou d'un regroupement) afin de :

- sélectionner un point de vue intéressant ;
- analyser les regroupements selon ce point de vue ;
- effectuer une généralisation utile des aspects observés ;
- réifier cette généralisation en une nouvelle entité de la classification.

Si le regroupement initial est à un méta-niveau par rapport à la représentation, l'abstraction produit un résultat dans le méta-niveau adopté et n'est donc plus interne à une représentation. Une abstraction est effective lorsqu'une entité correspondante est produite dans la représentation (réification). L'abstraction est, pour nous, le mécanisme de réification d'une généralisation. Elle explicite les conditions sous lesquelles une généralisation est effectuée et les classifications ou les règles qui ont permis cette généralisation.

---

[1] Selon le paradigme logique, ceci correspond à l'utilisation d'une théorie de la Preuve au lieu de travailler dans la théorie des Modèles. Il s'agit ici d'un mécanisme intégré dans l'activité du mathématicien. Il est chargé de capturer et formaliser certains usages opératoires.

5.1.4. Les connaissances qui nous intéressent

La constitution de connaissances nombreuses et variées requiert une **partition et une expression claire de ces connaissances.** Une représentation déclarative des connaissances est chargée de modéliser au minimum trois niveaux d'organisation :

- les concepts (faits), qui interviennent par exemple dans la composition des formules ;
- leurs relations (règles), qui vont déterminer les manipulations de ces concepts ;
- les stratégies de raisonnement et heuristiques, qui décident quelles manipulations effectuer.

Ces trois niveaux d'organisation relèvent d'un principe général. Ils s'appliquent par exemple :

- pour la vision ensembliste des mathématiques : les faits sont des éléments tels des nombres entiers ;
- pour les déductions du mathématiciens : les faits sont des relations logiques agrémentées d'un contexte de validité, les règles déterminent le mode d'application de ces relations, et le raisonnement décide de leur application ;
- pour la description des notations du langage, les faits sont les symboles et notations disponibles, les relations décrivent les liens, parfois mathématiques, entre les caractéristiques des constituants, le raisonnement décrit comment reconstituer ces liens à partir des informations incomplètes accessibles.

L'usage montre que ces trois sortes de connaissances sont interdépendantes : les relations s'expriment avec les concepts, et les stratégies sont définies à partir des relations et des concepts. Par exemple à partir du niveau mathématique de base, le concept d'inclusion ensembliste intervient dans la relation "si $A \subset B$, alors $A \cap B = A$"[2]. Une heuristique peut alors décider d'exploiter cette relation pour montrer que "$B \cap (A \cup B) = B$".

Les liens, voire le couplage, entre la description du domaine et les connaissances de résolution sont donc plus importants que la figure 3.31 ne le laissait apparaître : ne serait-ce parce que les problèmes et leur traitement sont définis à partir des concepts, et que les concepts sont aussi introduits pour traiter une classe de problèmes.

Nous nous sommes donc intéressé prioritairement à la **description du domaine relative aux concepts mathématiques.** L'analyse de la terminologie employée en mathématiques permet de recenser divers types d'entités de nature différente, concernant :

- l'organisation de l'univers mathématique ;
- les constituants de cet univers mathématique ;
- l'expression à l'aide d'un langage de Formules ;
- l'expression à l'aide d'un langage de Phrases.

De plus, chacune des ces entités peut être abordée sous trois aspects différents :

- les principes et règles de conception ;
- les entités créées proprement dites ;
- la dynamique d'utilisation de ces entités.

Le tableau figure 3.32 illustre chacun de ces douze cas par un exemple, chacun des cas formant un domaine dont la modélisation et la mise en œuvre est complexe. Nous nous intéresserons principalement à la première ligne.

---

[2] Notons que cette relation se place déjà au niveau d'une règle de déduction du mathématicien, alors que sa variante "$A \subset B \Leftrightarrow A \cap B = B$" est une formule qui doit être exploitée par une règle d'inférence pour se retrouver au même niveau.

aspects / types	*règles de conception*	*entités concernées*	*dynamique d'utilisation*
organisation	introduire une nouvelle structure	théories, structures, représentants	manipuler un représentant d'une structure
constituants	définitions, règles de composition	zéro, sommation, struct. d'anneau	résultat, méthodes, preuves
Formules	choix et règles de notation de Σ	$\sum_{k\leq n} x_k$	substitution de x_k par k+1 ou de Σ par Π
Phrases	phrase —> groupe nom., groupe verbal	"utiliser le théorème de Bezout"	plan d'action, commentaires, démonstration

figure 3.32

5.1.5. Les causes de limitation du modèle des connaissances disponibles

Nous souhaitons aborder l'activité mathématique dans sa généralité, afin de concrétiser les idées développées en première partie de la thèse. Nous tenons à considérer toute théorie en formation ou achevée comme un point de vue particulier modélisant l'univers mathématique. Dans certaines applications la modélisation est considérée comme une référence universelle. Mais, en général, l'utilisation d'une théorie est soumise aux **limitations** :

- de la modélisation, partielle ou à compléter, comme lors d'un prototypage ou d'une extension ;
- de sa validité ou de son exactitude, comme la théorie mentale d'un élève dans un tuteur intelligent ;
- des connaissances disponibles, sachant que beaucoup peuvent être déduites mais qu'une configuration minimale est peu efficace ;
- intrinsèques aux mathématiques, comme l'indécidabilité, l'incomplétude, le caractère finitaire de la modélisation, etc.

L'adéquation entre une modélisation et la réalité des besoins est alors soumise aux limitations suivantes : partielle, incomplète, imprécise, incertaine, défauts, cohérence locale, redondances, etc. Ces limitations sont d'autant plus sensibles que la quantité de connaissances, ou l'étendue des applications, est importante. Malgré cela, le problème crucial est que la plupart des connaissances ne sont pas directement accessibles pour être modélisées.

Afin d'évaluer l'**étendue du travail de modélisation des connaissances** à effectuer, nous avons choisi de présenter un sous-ensemble de règles de conceptions relatives aux formules. Elles concernent la phase d'assemblage, qui construit une nouvelle formule à partir de formules indépendamment bien assemblées et d'un opérateur mathématique qui les coordonne (figure 3.33).

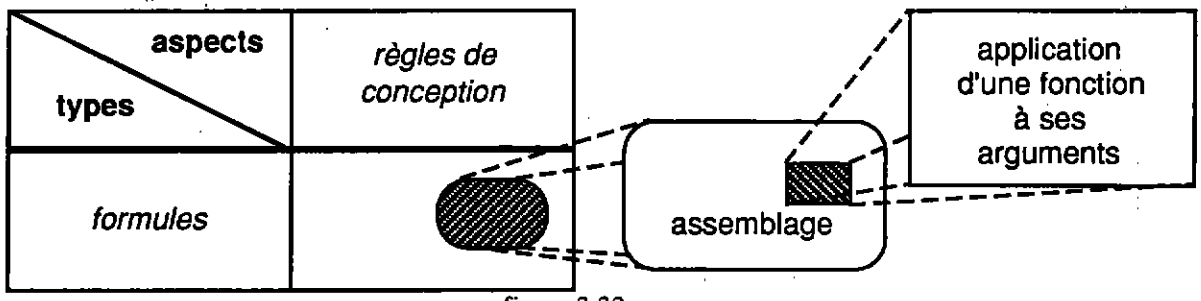


figure 3.33

L'utilisation informatique des connaissances nous oblige à les formaliser. Des difficultés surviennent alors pour expliciter les connaissances requises. Ainsi, le savoir-faire du mathématicien en ce qui concerne les règles d'assemblage, est **peu formalisé**. Au niveau de la phase d'assemblage, une formule est composée de sa sémantique et des choix syntaxiques qui interviennent dans sa forme. Il faut veiller à la cohérence des spécifications avec le résultat obtenu, car de nombreux problèmes syntaxiques peuvent introduire des dissonnances : se méfier, en particulier, des conflits (ou des incohérences) de symboles ou de notations.

Le cas des règles syntaxiques pour la structure générique d'application d'une fonction à son argument est évocateur :

- la syntaxe usuelle est trompeusement simple : "f" et "x" donnent "f(x)" ;
- si l'un des deux constituants est composé, viennent se rajouter des problèmes de parenthésage : "f+g" appliqué à "(x+y)" donne "(f+g)(x+y)" (seul le terme fonctionnel est modifié !) ;
- pour diversifier les usages des mathématiques, un formalisme "constructif" inverse l'ordre, "(x)f", tandis que le lambda-calcul préfère indifférentier les termes en choisissant "(f x)" ;
- des fonctions particulières adoptent des choix spécifiques. L'indexation a un comportement particulier, et agit sur la taille relative avec "x_k", "$x_k(t)$" et non "$(x(t))_k$". Le carré de "sin(ax+b)" va automatiquement contenir une inclusion par équivalence mathématique, et se présenter comme "$\sin^2(ax+b)$", qu'il faudra différencier de "$\sin^2 = \sin \circ \sin$"...

L'assemblage va donc développer des **notions spécifiques**, telles le choix d'un formalisme, le parenthésage, la réduction de taille, l'éclatement d'un terme composé... Les procédures de choix d'assemblage seront l'une des multiples connaissances mathématiques.

L'exemple de l'assemblage illustre la distance qui sépare l'utilisation de connaissances par l'ordinateur, de leur utilisation humaine implicite, basée sur l'expérience. Ce phénomène est compliqué par la diversité (qualitative et quantitative) des connaissances mathématiques. La constitution de connaissances mathématiques doit donc être précédée d'une **étude préparatoire approfondie des connaissances nécessaires et des problèmes soulevés par leur modélisation.**

5.2. NOS PROPOSITIONS D'ORGANISATION

5.2.1. L'organisation autour des concepts

Bien que tout système puisse subir une formalisation logique, les caractéristiques d'un langage logique permettent de critiquer la pertinence pour le mathématicien d'une telle alternative :

- "syntaxisation" :
  le mathématicien raisonne plus en terme de modèle intuitif, qu'en terme de manipulation formelle de structures ;
- atomisation :
  le mathématicien tend à regrouper des connaissances selon ses modes d'association. L'approche logique cherche au contraire à les dissocier en informations indépendantes mais atomiques. Une information est ainsi déshumanisée et peu adéquate pour un processeur humain ;
- homogénéisation :
  l'organisation des informations suit une structure universelle basée sur la prédication. L'organisation des connaissances chez l'homme dépend pourtant de sa nature et de son rôle dans les déductions. Il est notamment regettable que cela ce traduise aussi par une homogénéisation de la forme textuelle du langage ;
- "monopolarisation" :

> L'approche logique fige une interprétation et ne permet guère de jouer sur ses changements ou sur plusieurs niveaux de représentation. Notamment, l'approche logique "classique" (par opposition à la déduction naturelle) ne se prête guère à la description d'un méta-niveau à vocation déductive, puisqu'elle est basée sur l'application de quelques règles d'inférence et d'un guidage prédéfini.

De plus, un concept mathématique particulier est le point d'entrée des manipulations opératives rattachées à son usage. L'identification de sa présence peut parfois être la clé de la résolution d'un problème, ou constituer le fil directeur du procédé de résolution. C'est notamment la raison pour laquelle une mise sous forme clausale du problème perd une grande partie du pouvoir d'évocation et diminue d'autant les capacités de résolution d'un mathématicien. Un exemple détaillé dans [Bledsoe 86] pour le cas de la somme et du produit de deux limites, montre la difficulté d'automatiser de tels domaines fortement conceptuels sans ces informations.

Nous proposons alors d'organiser la modélisation des connaissances autour des concepts auxquels elles peuvent être rattachées. Dans une optique d'assistance, à moins d'une modélisation cognitive très poussée, les connaissances difficilement formulées sont du ressort du mathématicien. De même les connaissances trop spécifiques doivent être déduites. En revanche, il est bon que les **connaissances explicitement exploitées** dans les manipulations courantes soient présentes. Cela permet de suggérer ou d'assister leur usage, tout en les ayant à disposition comme argument dans la preuve en cours. Enfin, des connaissances pratiques sur l'utilisation d'un concept peuvent être déterminantes.

Cette façon de limiter les connaissances nécessaires aux connaissances facilement observables permet de contourner l'obstacle épistémologique d'une maîtrise totale de toutes les causalités. Sans même faire appel à ce qui concerne l'intuition, essayer d'appréhender le "pourquoi" de l'édifice mathématique requiert notamment la notion d'utilité d'un concept, et donc de sa provenance, de son usage et de ses limitations. Cela met en avant la dualité d'un concept, à la fois objet d'étude et outil pour étudier (cf nombre-dénombrer). Ces changements de perspectives sur l'usage d'un concept peuvent l'enrichir par un nouveau développement théorique et amener sa réévaluation.

La solution que nous proposons cherche à respecter les dépendances, non seulement quant à l'organisation conceptuelle, mais aussi quant à l'utilisation cognitive du mathématicien. Par exemple, un concept mathématique sera forcément associé aux moyens de le désigner, et aux propriétés dans lesquelles il intervient. Cette organisation permet à ce concept de **servir de point d'entrée aux multiples usages dans lesquel il est impliqué.**

Le concept de parallélogramme est un exemple illustrant la diversité des connaissances qui peuvent être rattachées[3]. Qu'obtient-on lorsqu'on permute les extrémités : rien ou un autre parallélogramme ? Quel sont ses liens avec le concept de milieu d'un segment, celui de distance ? Qu'est-ce qu'un coté, qu'une diagonale ? Que sait-on de sa position hiérarchique parmi les autres concepts ? Comment construire le parallélogramme ABCD ? Dans quels cas est-il quelconque ? Que peut-on dire si E est le symétrique de D par rapport à C, dans quel cas est-il inchangé par symétrie par rapport à une droite ? Quelles sont les propriétés possibles d'un quadruplet "(A,B,C,D)" qui ne forme pas un parallélogramme ? Est-ce qu'une rotation préserve la propriété de former un parallélogramme ? Quand est-il souhaitable d'introduire un point de façon à former un parallélogramme ? etc.

Ainsi, une description limitée du concept de groupe comprend :

---

[3] Si sa définition varie selon les programmes scolaires, ses propriétés restent intangibles : d'une part celles permettant de caractériser un parallélogramme, d'autres part les conséquences envisageables en présence d'un parallélogramme. Enfin ce concept illustre son intérêt uniquement par les manipulations qu'il procure (apprentissage, pédagogie, nécessité, diversité, représentativité ?), puisqu'il existe des procédés automatiques différents permettent de résoudre cette classe de problèmes géométriques.

description sémantique	
désignation	groupe (G,+)
concepts utilisés (hiérarchie)	ensemble, loi de composition
arguments et leur type	G de type "Ensemble", + de type G×G dans G
structure de la définition	structure d'association typée
propriétés vérifiées par définition	associativité
	existence d'un élément neutre
	existence d'un symétrique pour tout élément
équivalences (langage et math.)	l'existence d'un symétrique à droite suffit
utilisation mathématique	
théorèmes associés	l'existence entraîne l'unicité de l'élément neutre
concepts proches	l.c.i. commutative
	sous-groupe
tentatives de réfutation	pour un ensemble fini, regarder s'il existe des éléments idempotents
exemples	groupe additif des entiers rationnels
	groupe des permutations de l'ensemble E
utilisation pragmatique avancée	suppression du parenthésage de composition

5.2.2. Partition des connaissances de traitement

Une autre présentation des six dimensions pertinentes consiste à se focaliser sur les fonctions mathématiques et les fonctions de communication du langage (précédemment abrégées en mathématiques et langagières). Pour détailler la figure 1.8, les concepts se situent dans la zone mathématique élémentaire. La terminologie qui leur est associée se situe de l'autre coté de la ligne de démarcation du langage (figure 3.34).

La taxonomie mathématique des concepts indique par exemple que la fonction sinus et la fonction "f" sont des fonctions impaires et décrit leur propriétés mathématiques. La taxonomie langagière indique que le syntagme "fonction impaire" forme une catégorie particulière de "fonction" ; elle classifie et décrit la propriété "imparité" responsable de cette spécificité.

Mais ce qui fait la puissance du langage, c'est que ce "*couplage fort*" est limité aux constructions élémentaires du langage : les concepts mathématiques et leurs structures textuelles associées (ou associables).

Au delà, les structures deviennent complexes et s'organisent séparément. Il faut donc des représentations intermédiaires et une synthèse multicritères pour matérialiser leur correspondance. Nous qualifierons de "*couplage faible*", le lien ténu reliant des entités mathématiques plus profondes (telles que problème, plan de résolution, preuve, etc.) à d'autres entités langagières profondes (telles que énoncé, schéma de présentation, démonstration, etc.).

Rien d'étonnant, alors, que lorsqu'un nouveau concept émerge dans l'univers mathématique (zone apparente), les structures langagières requises pour le manipuler soient créées conjointement (couplage fort : créer-désigner). On peut empiriquement postuler qu'il existe un seuil au dessus duquel il devient "nécessaire" de nommer.

Ce seuil est fonction de la complexité de la structure à nommer et de sa fréquence d'utilisation. La complexité est relative aux entités déjà introduites, et peut provenir d'un besoin de l'univers mathématique ou être uniquement langagière (par ex., dans le cas de paramètres introduits pour faciliter la manipulation d'une expression symbolique). Cette loi s'applique aussi aux entités complexes que sont les théorèmes ou certaines formes de preuves (par récurrence, etc.). Les ressources langagières choisies pour les nommer (phrases, syntagme, nom composé, nouveau nom, notation, etc.) participent aussi à leurs relations sémantiques.

La terminologie associée à une classe est un indicateur de la fréquence de son usage. Un concept primitif ou souvent utilisé aura une étiquette qui lui est propre, marquant ainsi son indépendance. Un nœud "secondaire" sera défini relativement à d'autres classes. Son étiquette est définie par exemple à partir de celle de la classe dont elle est une spécialisation (par ex. fonction de la variable réelle périodique, continue et ne s'annulant pas). Pour le désigner, il faut utiliser des phrases d'un **langage de description**. Cela peut être, par exemple, exprimé à l'aide de connecteurs et des prédicats définissant d'autres classes.

Connaissances de Traitement du Langage Mathématique

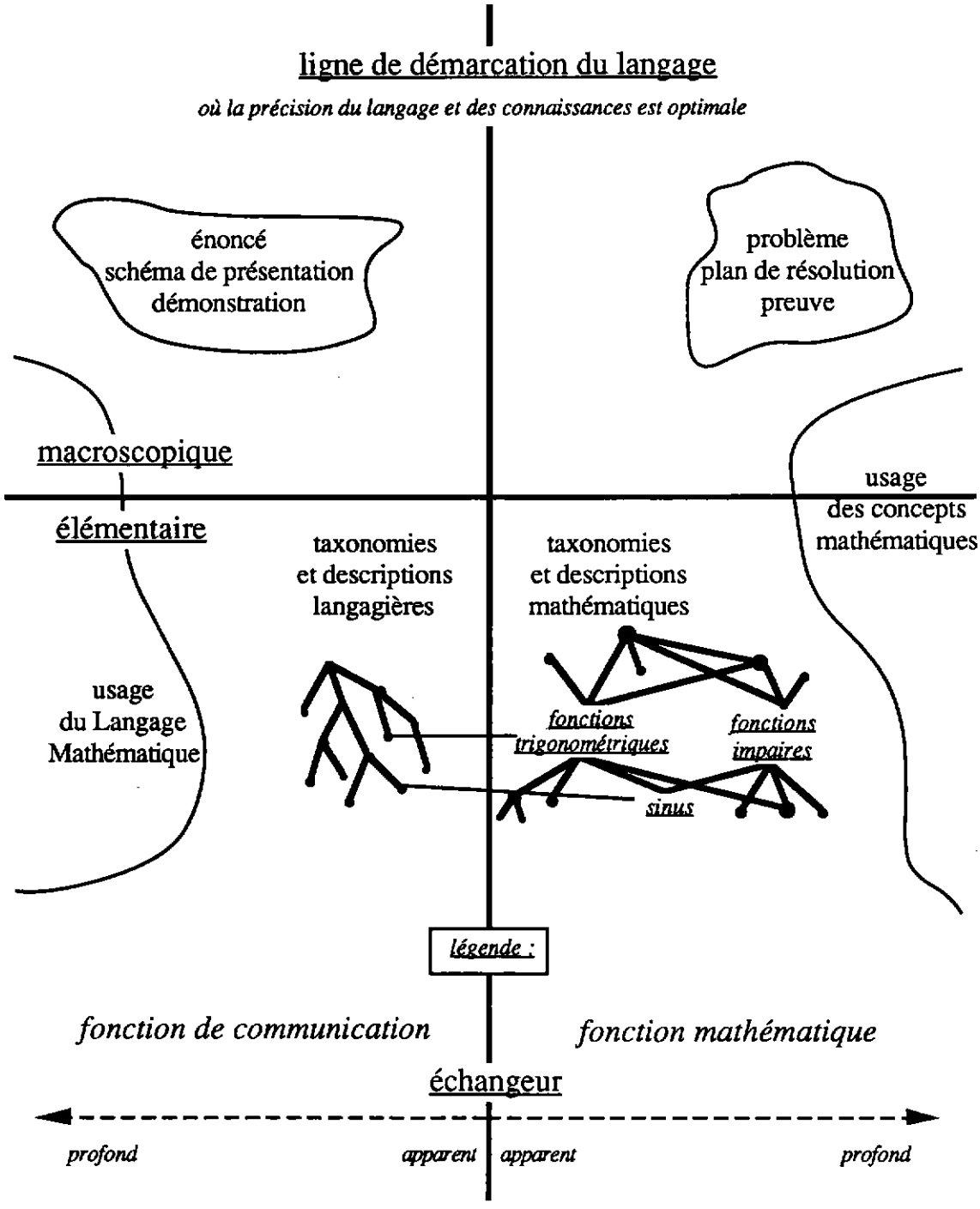


figure 3.34

La ligne de démarcation est ainsi une sorte de frontière entre le langage et la pensée. Pour reprendre les termes de M. Loi [Loi 82], « le langage est un énorme échangeur qui rassemble en profondeur les zones les plus diverses des mathématiques », autour de cette ligne de démarcation.

5.2.3. La classification des connaissances

Le savoir est d'abord classificatoire.

Cette maxime rappelle qu'il n'y a pas de savoir sans organisation des objets et concepts manipulés. Comme nous venons de le voir, construire la taxonomie des concepts d'un domaine d'expertise permet d'approcher la structure des connaissances. Ces concepts sont, en effet, au moins utilisés comme intervenants dans les feuilles terminales des autres classifications nécessaires (définition des problèmes, usage des opérations de déduction de preuve, structures langagières, assemblage, etc.). Ainsi, lorsqu'une classification est nécessaire pour introduire des abstractions finalisées à un traitement, c'est à partir des concepts mathématiques que vont être cristallisées les règles et leurs exceptions.

Les connaissances nécessaires à la pratique du mathématicien incluent ainsi le traitement du langage. Réciproquement, l'analyse du langage permet d'établir des connaissances utiles : la méthode KOD de Claude Vogel se base, par exemple, sur une analyse langagière pour l'élicitation et l'organisation des concepts et autres connaissances [Vogel 88]. Nous nous inspirons de ce livre pour dégager des points qui nous semblent importants :

- **Distinguer la classification de la terminologie.** Classifier les éléments d'un domaine consiste à les grouper sur la base de relations. La terminologie consiste à nommer les classes ainsi définies dans un langage donné. La terminologie employée traduit l'aspect naturel ou artificiel d'une classification.
- **La classification est une modélisation d'un domaine.** Elle constitue un point de vue particulier de la topologie du domaine. En particulier elle ne s'intéresse qu'à certaines relations, et introduit un biais volontaire ou non.
- **Une classification a une structure.** C'est un arbre, un arbre binaire, un treillis, etc. La terminologie associée peut être locale à chaque nœud ou relative à l'emplacement dans la structure. Comme les nœuds, les arcs peuvent être étiquetés et ordonnés, des contraintes ajoutées...
- **Une classification est associée à une sémantique.** Les arcs peuvent signifier "est une sous classe de", "est un constituant de", etc. Il est important de veiller à l'homogénéité sémantique de la classification.
- **Une classification a un rôle pratique,** comme la représentation des connaissances, l'identification, etc. Ce rôle pratique implique des critères de constitution variés : par exemple, les fonctions impaires sont définies par une propriété, tandis que les fonctions trigonométriques sont issues de relations mutuelles privilégiées.

Les classifications englobent alors tous les aspects, mathématiques et langagiers, de l'activité mathématique. Elles forment des structures hiérarchiques indispensables à l'expression de résultats mathématiques plus généraux. Savoir, en effet, qu'un résultat est vérifié pour la fonction sinus est moins intéressant que de connaître la relation plus générale vérifiée dès que la fonction f est continue.

Conformément à notre approche du langage, nous distinguons deux sortes de classification :

- classification langagière :
  la classe des symboles connus, des symboles possibles, des symboles de variable, des termes bien formés, etc ;
- classification mathématique :
  la classe des entiers, des entiers pairs, des entiers premiers, des fonctions continues, des fonctions dérivables, des fonctions réelles d'une variable réelle, etc ;

Les structures mathématiques permettent deux sortes de classification mathématique :

- une **classification intrinsèque** qui regroupe des constituants en fonction de leur structure commune :
  les entiers, les fonctions, les fonctions réelles d'une variable réelle, etc ;
- une **classification relative** qui regroupe des constituants en fonction de leurs propriétés communes :
  les entiers non nuls, les entiers pairs, les fonctions paires, etc ;

Nous cherchons à définir la deuxième classification à partir de la première qu'elle englobe. La première classification consitue alors un moyen de typage permettant de décrire des objets "nouveaux". La seconde classification permet d'utiliser un langage de description pour organiser et ne sélectionner que certains constituants d'une catégorie trop générale. Dans un contexte ensembliste la première classification définit un ensemble de référence, tandis que la seconde permet de constituer des sous-ensembles quelconques (ou de sélectionner des éléments de l'ensemble des parties).

Cette deuxième classification est appellée "relative car elle dépend de la propriété considérée et de l'usage qui en est fait : elle est alors intensionnelle. Par exemple, la propriété "être différent de l'élément absorbant 0" est utile dans un contexte multiplicatif, et souvent superflue par ailleurs. De même, la partition des entiers selon leur parité est fréquemment utilisée. La partition des nombres en "-1 et le reste" est pertinente pour l'intégration mais paraît surprenante en premier lieu. De telles classifications relatives ont alors un intérêt limité à quelques utilisations. Il est alors souvent inutile qu'elles apparaissent explicitement dans l'organisation des connaissances, même si elles doivent être rattachées à un type de problème ou déduites pour la résolution de problèmes.

5.2.4. Les structures d'organisation

L'organisation en structures et la classification sont les deux formes de base de la modélisation des connaissances mathématiques. Les mathématiques utilisent des structures de :

- **description des informations** :
  un nombre de type entier a une valeur et éventuellement des propriétés comme sa parité, sa décomposition en facteurs premiers, etc ;
- **organisation mathématique** :
  un ensemble, un couple, un groupe, etc ;
- **composition mathématique** :
  expressions formées à partir des éléments d'un ensemble et de relations mathématiques comme l'addition, la concaténation, la composition de fonction, le produit cartésien, etc ;

La composition d'objets mathématiques est le principe de formation d'expressions mathématiques. Nous distinguons deux sortes de relations en mathématiques :

- celles permettant la **génération** de nouveaux objets :
  par exemple en partant de rien, les entiers naturels puis les entiers relatifs ou les couples d'entiers ou l'ensemble des parties de N, etc.
- celles qui **réorganisent** des objets déjà connus :
  par exemple l'addition entière produit un entier naturel déjà introduit, etc.

Une relation de réorganisation consiste à effectuer une composition dont le résultat peut s'interpréter à l'aide des moyens déjà disponibles. Une relation de génération produit des objets qui ne sont en aucun cas compatibles avec l'interprétation des objets actuellement disponibles. Elle n'apparaît en mathématiques que par un choix conscient et muri. Par exemple, l'introduction de zéro, des nombres négatifs, etc, correspondent à un besoin d'extension dont la justification intuitive n'est apparue que par des interprétations de ces nombres dans des Modèles pratiques.

Les classifications tentent d'organiser des structures très variées. Les principes de généralisation qui régissent une classification sont multiples et forment l'une des préoccupations majeures des études sur l'apprentissage par un agent. Ils permettent une classification ascendante (du particulier au général) ou descendante, et sont souvent utilisés pour une classification transversale. Nous en présentons quelques uns que nous avons isolés à partir d'exemples mathématiques :

- **l'abstraction de propriétés** :
  Q* et Z présentent des similarités de comportement lorsque le mathématicien s'intéresse aux propriétés d'utilisation de leur loi de composition interne respective : x et +. Une fois ces similarités identifiées, le mathématicien peut les abstraire en créant la structure de groupe (G,⊥). Cette structure regroupe uniquement les éléments nécessaires au point de vue adopté : un ensemble, une loi, et des propriétés. Cette abstraction se traduit par une perte d'information jugée temporairement inutile. La théorie des groupes dit, en effet, bien peu de choses sur les propriétés de (Z,+)... L'abstraction est ainsi fréquemment utilisée pour déclarer deux objets égaux, à un isomorphisme près.
- **la généricité paramétrée** :
  Cette structure de groupe (G,⊥) une fois isolée et étudiée, le mathématicien cherche à exploiter le bénéfice de ses résultats. Pour cela il effectue une correspondance entre l'ensemble G, qualifié de générique, et l'ensemble à étudier tel Z/2Z, considéré comme un paramètre. Une fois la loi précisée également, il reste à vérifier que les propriétés constitutives d'un groupe sont vérifiées. Par ce paramétrage le mathématicien effectue un changement d'interprétation.
- **la généralisation de constituants** :
  Cette technique très riche est utilisable pour toute structure. La difficulté consiste à déterminer les variations de constituants (généralisation ou particularisation) intéressantes. Cette technique fait varier indépendamment ou non les paramètres intervenant dans une définition ou une classification. Par exemple pour la structure de groupe, affaiblir l'inversibilité (la supprimer

donne un monoïde, ne donner que l'inversibilité à droite est équivalent). Cette technique est couramment utilisée pour particulariser un résultat à des cas usuels ou dégénérés, en exploitant une classification ou ses instanciations. Elle peut être utilisée dans l'esprit des travaux heuristiques de Douglas Lénat, pour coupler ou découpler des paramètres (par ex. R(x,y) donne R(x,x)).

- **la mise à plat de l'information** :
  Ce principe est peu mis en avant à notre connaissance. Il permet le regroupement d'informations nécessaires et l'élimination de structures superflues (fonction d'oubli essentielle dans toute généralisation). Ainsi R. Godement [Godement 66, p 53] simplifie la définition théorique d'une fonction où le triplet (G, X, Y) est abrégé au graphe G (qui est une partie de X×Y). Nous illustrons figure 3.35 le cas de la structure d'algèbre (au sens des logiciens), triplet formé d'un ensemble univers A, d'un ensemble ordonné de relations et d'un ensemble ordonné de fonctions internes. La structure de groupe est un cas particulier d'une hiérarchie issue de l'algèbre, mais elle met à plat les informations et reste ainsi auto-suffisante (c.-à-d. indépendante de cette classification plus générale).

Exemple de mise à plat d'information

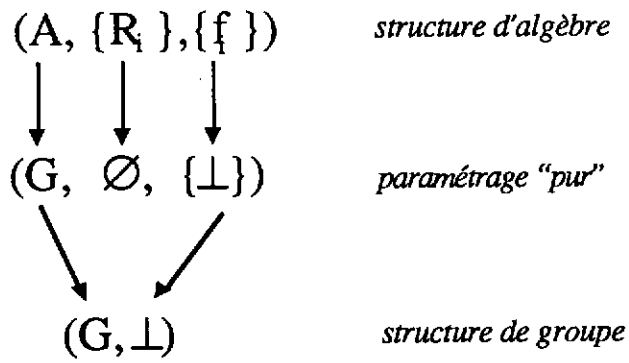


figure 3.35

- **la combinaison de structures** :
  Il s'agit d'amalgamer, de fusionner deux structures en une seule (ne pas confondre avec la composition comme dans "gof"). Il peut s'agir d'une simple juxtaposition si les deux structures sont découplées, mais cela comprend habituellement une fusion des informations communes, accompagnée de propriétés de compatibilité supplémentaires. Ainsi, deux groupes basés sur un même ensemble peuvent être combinés en une structure d'anneau, moyennant notamment une propriété de distributivité à respecter et des zéros distincts. Un groupe ordonné relève lui-aussi d'une telle combinaison entre structure de groupe et structure d'ordre...

Nous utiliserons le terme "*catégorie*" pour désigner un nœud d'une classification[4]. Pour organiser ces dépendances hiérarchiques complexes (telle une généricité associée à une mise à plat, ou une spécialisation de certains champs), il nous semble indispensable que **les conditions de passage d'une catégorie à une catégorie adjacente soient explicites**. De plus, il reste à préciser les lois mathématiques et organisationnelles qui régissent ces conditions de passage.

En général les classifications mathématiques ne sont ni disjointes ni complémentaires : il est fréquent d'avoir une catégorie C qui se décompose en C_1 et C_2 tels que $C_1 \cap C_2 \neq \emptyset$ et $C_1 \cup C_2 \neq C$. Par exemple, la catégorie des fonctions se décompose en fonctions paires, impaires, périodiques, etc.

Ces décompositions dépendent de l'intérêt de l'introduction de catégories intermédiaires. Elles peuvent d'ailleurs être insérées dynamiquement pour satisfaire des besoins nouvellement apparus. Ainsi, il est parfois intéressant de regrouper les catégories ci-dessus pour qualifier des fonctions dont le domaine d'étude peut être réduit par des transformations simples. Le plus souvent, ceci est du ressort d'une classification relative.

---

[4] Les catégories mathématiques exposées dans les traités forment une organisation complexe de type graphe acyclique. Il n'y a pas forcément d'élement minimal unique, et il y a aussi des définitions recursives croisées. De plus, les liens hiérarchiques sont issus de principes de généralisation différents.

Au terme de cette analyse, il apparaît que l'organisation des catégories mathématiques est complexe. Il faut remarquer qu'il n'existe pas, à notre connaissance, de modélisation conceptuelle notable d'un domaine mathématique. La progression récente de l'expressivité des langages de représentation des connaissances permettra probablement d'y remédier. Il serait toutefois irréaliste d'y songer sans une **étude approfondie des spécificités mathématiques.** Nous proposons notamment de :

- analyser des textes de traités mathématiques afin de :
  - déterminer les liens entre les connaissances exprimées ;
  - déterminer les stuctures des classification utilisées ;
  - proposer une modélisation qui soit validée comparativement à un passage précis ;
- recenser les difficultés et les études complémentaires à mener, notamment quant à l'utilisation dynamique de ces connaissances pour résoudre des problèmes mathématiques ;
- choisir un langage de représentation disponible afin de :
  - évaluer les biais et leurs conséquences ;
  - exploiter cette représentation dans un mécanisme de résolution ;
  - étendre le système.

C'est le sens de notre démarche, bien que nous n'ayons exploré que les deux premiers points. Nous n'avons pas réussi à obtenir une modélisation suffisamment complète pour être validée comparativement. Nous proposons donc ce que nos investigations ont pu établir, c'est-à-dire des avancées concernant certains points importants.

Sans revenir sur les acquis de la partie I, nous avons proposé dans ce chapitre une partition et des principes de classification des connaissances mathématiques. Nous poursuivons section suivante par une analyse plus détaillée de la notion de définition, afin d'illustrer notre démarche d'étude linguistique. Nous complétons ce chapitre par une exploitation possible d'une représentation à objets pour permettre une dynamique "intuitive" liée à la nature des connaissances.

5.3. UNE ANALYSE DES CONNAISSANCES : LA NOTION DE DEFINITION

5.3.1. Les dimensions d'analyse des définitions mathématiques

Définition 1.
Soient n et p deux entiers naturels, **on appelle** ppcm de n et de p le plus petit commun multiple de n et de p.

Cette définition du Langage Mathématique peut se modéliser en :

Contexte de formulation :
n et p deux entiers naturels

Construction mathématique :
le plus petit commun multiple de n et de p

Convention de présentation :
ppcm de n et de p

Cette règle des trois C : Contexte, Construction, Convention se définit comme suit :

Contexte de formulation :
il précise les objets mathématiques indispensables déjà construits au moment de l'utilisation de la définition. Pour cela, le contexte définit les catégories qui les engendrent ainsi que les propriétés supplémentaires vérifiées.

Construction mathématique :
définit la nature de la nouvelle construction en étant soumise à deux restrictions :

- les moyens de la construire doivent être disponibles au moment de la définition ;
- cette construction doit être mathématiquement validée afin que la définition le soit aussi.

Convention de présentation :
elle définit les conventions langagières (Langages de Phrases et de Formules) qui permettent d'écrire et de désigner l'objet construit en fonction de ses constructeurs, ou réciproquement d'accéder aux constructeurs et objets intermédiaires à partir de la donnée d'un tel objet.

Définition 2.
Soient x et y deux objets mathématiques, **on appelle** couple de x et de y le troisième objet mathématique ainsi défini.

Définition 3.
On dit qu'un objet z **est un** couple **s'il** existe des objets x et y tels que z = (x,y).

Définition 4.
On appelle fonction un triplet
$$f = (G,X,Y)$$
où G,X,Y sont des ensembles assujettis à vérifier les conditions suivantes :
(F 1) : on a $G \subset X \times Y$;
(F 2) : pour tout $x \in X$ il existe un et un seul $y \in Y$ tel que $(x,y) \in G$

Définition 5.
Etant donnés deux ensembles X et Y, **On appelle** application de X dans Y toute fonction ayant X pour ensemble de départ et Y pour ensemble d'arrivée.

Ces exemples[5] permettent d'introduire les dimensions d'analyse suivants :
- Le **texte de définition** (structure textuelle de la structure apparente de lettres) du document mathématique :
  la structure de surface d'un texte de définition est identifiable à l'aide de marqueurs particuliers tel "*Définition 1*" ou à l'aide de marqueurs de définitions comme "Soient " et "on appelle", "on dit que".
- **La structure profonde** de la définition textuelle (compréhension du texte) :
  cette structure correspond ici aux trois C. Elle peut subir une transformation importante due aux facilités de présentation textuelle. Souvent par exemple, des constituants du contexte de formulation se trouvent répartis au sein de la structure de surface afin d'obtenir un style de présentation agréable.
- **La nature et le type** de la définition mathématique :
  les définitions mathématiques portent parfois sur des objets purement mathématiques, parfois sur des procédés langagiers. Cette première dimension d'analyse permet de déterminer la nature de la définition. Il se pose un problème générique d'utilisation d'un langage indépendamment d'un contenu sémantique particulier. Le type d'une définition répond alors à des usages différents mais corrélés (construire, nommer, catégoriser, noter...).
- La **structure de la théorie** mathématique dans laquelle cette définition s'insère :
  cette définition n'a rien d'arbitraire car elle contribue à l'édification des mathématiques. Elle vient compléter d'autres résultats et définitions existants, par exemple elle existe grâce à un théorème d'existence et d'unicité.
- L'**utilisation active** de la définition :
  L'utilisation active permet la décomposition de la construction mathématique, par opposition à une utilisation passive qui se contente d'utiliser son nom (ou sa forme) dans un raisonnement. L'utilisation passive n'exploite alors que des propriétés issues de cette définition.

Nous distinguons donc cinq dimensions d'analyse d'une définition :
- le texte ;
- la structure profonde du texte ;
- la typologie des définitions mathématiques ;
- la validité dans la théorie ;
- les modes d'utilisation.

Il faut rajouter à cela l'organisation des ingrédients rattachés aux définitions lors de la phase de constitution, tels qu'ils apparaissent dans un traité mathématique.

[5] les définitions 3, 4 et 5 sont resp. issues des pages 51, 53 et 54 de [Godement 66].

5.3.2. Le texte d'une définition

L'analyse des exemples précédents montre qu'il y a plusieurs schémas de définition. Nous avons pour notre part effectué une analyse minutieuse de deux parties du traité d'algèbre de Roger Godement (pages 50 à 57 sur la notion de fonction, et pages 284 à 309 sur la notion d'applications bilinéraires et multilinéaires alternées).

Nous nous sommes tout d'abord intéressés aux **termes des phrases** employées lors des définitions (on appelle, on note...). Nous avons cherché les nuances susceptibles de justifier l'emploi prioritaire d'un terme, éventuellement à l'insu de son auteur. Des conjectures partielles de cette analyse ont souvent été invalidées par des passages ultérieurs. Cela rappelle très justement qu'il n'y a pas de norme en mathématique et que tout résultat est susceptible d'être étendu, précisé, remis en cause ou plus simplement qu'il est susceptible de ne pas être respecté !

Nous avons, par exemple, observé les catégories suivantes pour introduire une convention d'écriture en Langage de Formules (les nuances sont plus nombreuses en Langage de Phrases) :

- noter (on note, on désigne par une notation, etc)
  un troisième objet que l'on note (x,y), etc
  s'emploie pour définir un objet mathématique.
- écrire (on écrit, s'écrit, etc)
  on écrit x=pr1(z) ; on écrit $u \in \omega^+(A)$, etc
  s'emploie pour définir une relation mathématique.
- poser (on pose, etc)
  on pose g(x) = f(x) si $x \in X'$, c sinon ; on pose (x,y,z) = ((x,y),z) ; on déduit une troisième application h... en posant h(x) = g(f(x)) pour tout $x \in X$, etc
  s'emploie plutôt pour introduire une relation à partir de conventions de notation déjà existantes.

Notre analyse est la suivante : Lorsque l'on écrit ou que l'on note, on introduit une nouvelle forme d'écriture au sein du langage, et l'on donne son interprétation en Langage de Phrases. Lorsque l'on pose, on définit simplement l'interprétation d'un terme du langage à partir d'autres termes déjà connus : cette description est autosuffisante, si ce n'est pour la partie contextuelle, et une interprétation de la construction en Langage de Phrases est donc superflue.

Cette analyse reste incomplète et nous proposons la continuation suivante pour **bâtir une modélisation informatique incluant la gestion du langage** :

- définir les structures de données nécessaires à la définition de la règle des trois C par un langage de description spécialisé ;
- recenser, pour chaque type de définition, les schémas de présentation possibles ;
- définir les critères de choix des schémas de présentation en fonction du contenu des différents champs (par ex. présence et longueur des champs).

Nous ne développons pas plus avant cette voie de recherche de linguistique des mathématiques. La description de la définition 4 peut s'effectuer par exemple :

Contexte :

G,X,Y : ens ; (*déclaration*)

$G \subset X \times Y$;

$\forall x \in X, \exists! y \in Y$ / $(x,y) \in G$. (*suite de formules, qui sont donc des relations à verifier*)

Construction :

(G,X,Y).

Convention :

f= ; (*en LF*)

fonction. (*en LP*)

Après nommage des relations du contexte, et interprétation de cette description, il est possible d'envisager, moyennant des schémas de présentation textuels pour les constituants, la production du texte de la définition :

Définition 4.
On appelle fonction un triplet
$f = (G,X,Y)$
où G,X,Y sont des ensembles assujettis à vérifier les conditions suivantes :
(F 1) : on a $G \subset X \times Y$;
(F 2) : pour tout $x \in X$ il existe un et un seul $y \in Y$ tel que $(x,y) \in G$

La simplicité de cet exemple illustre clairement la diversité de connaissances et de traitements à mettre en œuvre dans la réalisation d'un système AMI. Nous pensons néanmoins que la difficulté principale de cet exemple réside dans l'intégration de fonctionnalités existantes pour une finalité mathématique.

La possibilité de décrire ainsi cette définition de fonction est, de plus, insuffisante. Cette définition est complétée de toute une terminologie (graphe, valeur, ensemble de départ, etc.). L'usager des mathématiques a conscience que la plupart des définitions introduites dans le langage servent à faciliter les descriptions ("application f de X dans Y"), les manipulations ("$pr_1(G)$" ou "ensemble de départ" pour X) et à capturer les "configurations" intéressantes pour l'usage mathématique (si Y=X).

Ces compléments indispensables nécessitent des études approfondies concernant l'usage du Langage Mathématique. Ces études devraient permettre de spécifier les besoins en fonction de la Construction des données d'une définition (définir les accesseurs, manipulateurs, observateurs, etc.). L'introduction d'une définition pourrait alors consister à définir tous ces ingrédients nécessaires, en interaction avec le système. Une grande partie de ce travail devant être facilité par un système AMI, la compréhension textuelle de la Définition 4. ne nous semble pas intéressante en saisie.

Il paraît donc irréaliste de disposer d'une prise en compte fine

du Langage Mathématique dans un avenir proche.

5.3.3. Introduction d'une construction mathématique

Indépendamment du traitement du langage, l'analyse des textes mathématiques permet de concevoir l'organisation et la gestion des connaissances conceptuelles en mathématique. Une définition permet de :

- caractériser une construction de façon générique ;
- désigner cette caractérisation (les définitions sans procédé de désignation spécifique ne sont qu'une affirmation ou une classification) ;
- mémoriser cette caractérisation ;
- utiliser activement cette définition dans d'autres circonstances ;

La caractérisation ainsi réifiée en définition est utilisée ultérieurement dans un raisonnement. Nous nommerons ***définition intrinsèque*** une telle définition. Nous nommerons ***définition langagière*** les termes et procédés langagiers utilisés pour désigner des composantes d'une autre définition (par exemple son nom ou ses constituants). Une définition langagière peut donc s'appliquer réflexivement à une autre définition langagière.

Une définition est alors un choix de structure-régularité-invariance intéressant, qui présente un pouvoir d'abstraction descriptif et calculatoire (opératif). Sa validité peut être langagière (notation abréviative ou terminologie abrégée) ou mathématique : on décide de considérer les objets construits de telle façon parce qu'ils vont permettre des transmissions, factorisations et récupération de propriétés intéressantes. Nous nous intéressons ici à l'introduction d'une définition intrinsèque définissant une nouvelle structure mathématique. Le cas du couple introduit dans [Godement 66, p50-51] (cf. figure 3.36) est pris comme exemple.

§ 2. La notion de fonction

1. Couples

Nous avons introduit, au § précédent, les signes $=$ et $\in$, qui servent à construire des relations. Nous allons maintenant introduire une opération qui sert à construire des objets mathématiques.

Cette opération consiste à former, à l'aide de deux objets mathématiques x et y énoncés dans cet ordre, un troisième objet, que l'on note

$$(x, y)$$

et qu'on appelle le **couple** (x, y); et l'opération consistant à former des couples est soumise à une seule règle d'emploi, que voici : *pour que l'on ait*

$$(x, y) = (u, v)$$

il faut et il suffit que l'on ait

$$x = u \quad \text{et} \quad y = v.$$

En particulier, on ne peut avoir $(x, y) = (y, x)$ que si $x = y$, ce qui montre que l'ordre dans lequel on écrit les deux objets figurant dans un couple est essentiel. On aura soin en particulier de ne pas confondre le couple (x, y) avec l'ensemble $\{x, y\}$ défini au § 1.

Remarque 1. On peut considérer la notion de couple comme un *signe fondamental* (§ 0, n° 1) qui, avec les signes

$$\text{ou}, \quad \text{non}, \quad \tau, \quad \square, \quad =, \quad \in$$

et des lettres, permettrait d'écrire les Mathématiques en langage formalisé. Mais on peut aussi exprimer la notion de couple à l'aide des autres signes fondamentaux (ou d'abréviations qui s'y ramènent) : il suffit de prendre comme définition de (x, y) l'ensemble

$$\{\{x\}, \{x, y\}\}$$

dont les *éléments* sont l'ensemble $\{x\}$ et l'ensemble $\{x, y\}$ — il est en effet visible qu'en définissant ainsi un couple on satisfait à l'axiome fondamental donnant la condition d'égalité de deux couples. Toutefois cette seconde méthode met l'accent sur un aspect de la notion de couple qui est parfaitement dénué d'intérêt. Il est donc bien préférable de s'en tenir à la méthode adoptée plus haut, la seule et unique question ayant une importance mathématique étant en effet de connaître les conditions pour que deux couples soient égaux.

On dit qu'un objet z est un couple s'il existe des objets x et y tels que $z = (x, y)$; en vertu de la règle énoncée plus haut, les objets x et y sont alors bien déterminés par la donnée de z; on dit que x est la **première projection** et y la **seconde projection** de z, et on écrit alors

$$x = pr_1(z), \qquad y = pr_2(z).$$

On dit qu'un ensemble G *est un* **graphe** *si tout élément de* G *est un couple.* Lorsqu'il en est ainsi, il existe deux ensembles X et Y caractérisés par les conditions suivantes : la relation $x \in X$ (resp. $y \in Y$) équivaut à l'existence d'un $z \in G$ tel que $x = pr_1(z)$ (resp. $y = pr_2(z)$). On écrit alors

$$X = pr_1(G), \qquad Y = pr_2(G).$$

On peut étendre comme suit la notion de couple. Étant donnés trois objets x, y, z on pose

$$(x, y, z) = ((x, y), z);$$

on dit que (x, y, z) est un **triplet**; pour que l'on ait

$$(x', y', z') = (x'', y'', z'')$$

il faut et il suffit que l'on ait

$$x' = x'', \qquad y' = y'', \qquad z' = z'';$$

en effet la relation considérée s'écrit $((x', y'), z') = ((x'', y''), z'')$, donc équivaut à $(x', y') = (x'', y'')$ et $z' = z''$, donc à $x' = x'', y' = y''$ et $z' = z''$.

De même, étant donnés quatre objets x, y, z, t on pose

$$(x, y, z, t) = ((x, y, z), t)$$

et on dit que (x, y, z, t) est un **quadruplet**, etc, etc...

2. Produit cartésien de deux ensembles

Soient X et Y deux ensembles; on peut démontrer (à l'aide des méthodes du § 0) qu'il existe un ensemble Z caractérisé par la propriété suivante : *pour que l'on ait $z \in Z$, il faut et il suffit qu'il existe $x \in X$ et $y \in Y$ tels que $z = (x, y)$*. On dit que Z est le **produit cartésien** de X et Y, et on écrit

$$Z = X \times Y.$$

L'ensemble produit $X \times Y$ est donc l'ensemble des couples (x, y) où $x \in X$ et $y \in Y$.

Le méta-procédé de création d'une structure

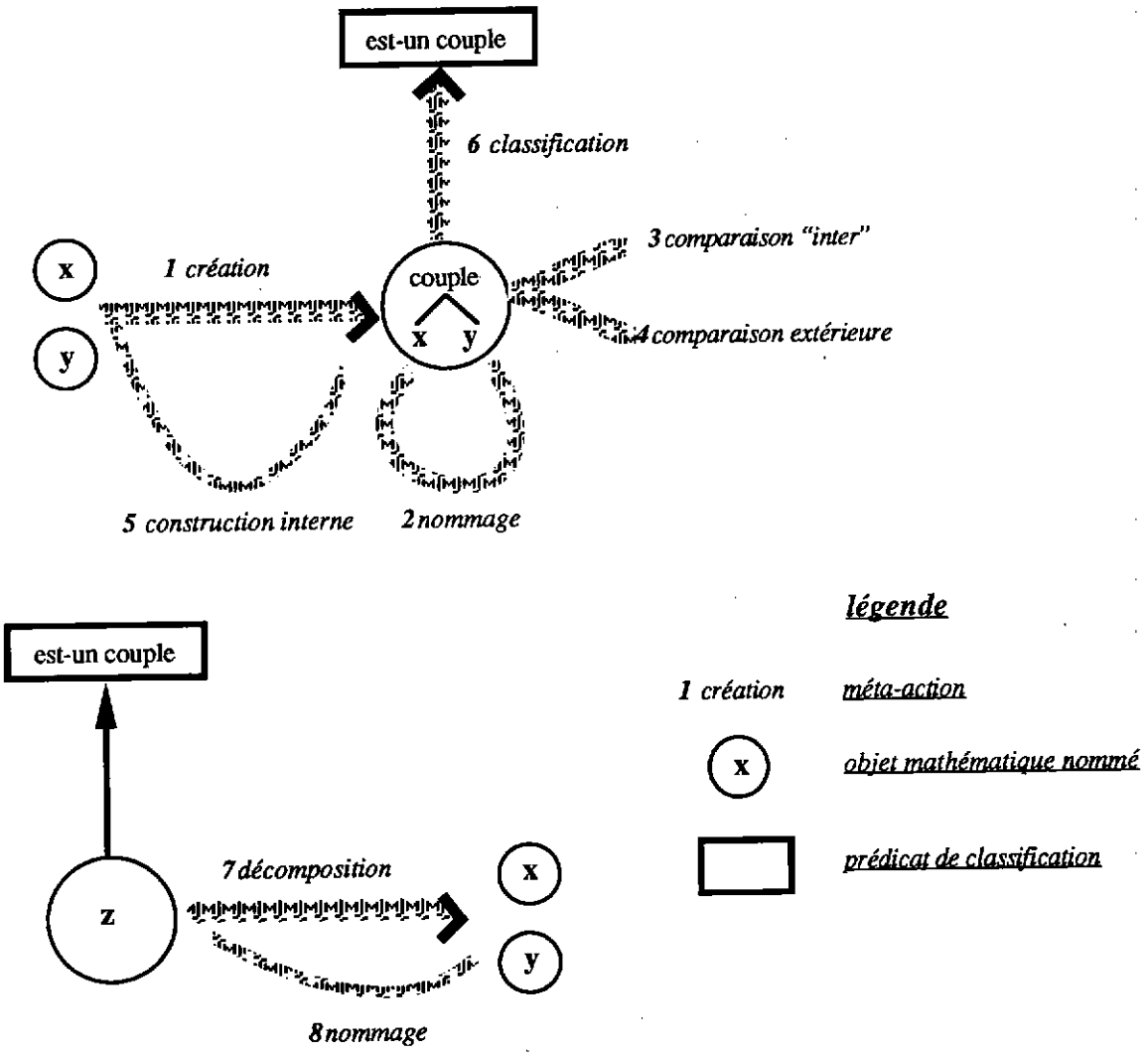


figure 3.37

Lors de la constitution de la définition, et de l'environnement mathématique nécessaire à son usage, l'activité du mathématicien requiert implicitement des manipulations de chargement (bootstrap) d'une nouvelle structure. Dans le livre de Roger Godement (p 50-51), un Modèle mathématique est supposé presque complètement défini. A ce point, la définition d'un couple (figure 3.37) :

- étend le Modèle par un nouveau procédé de construction d'objets mathématiques (au niveau métamathématique, ou plutôt, en utilisant un méta-niveau descriptif) ;

1) la capacité d'engendrer un objet qui ait une existence mathématique est présupposée. Cela permet la création d'une "structure" nouvelle à partir de deux objets quelconques déjà existants !

- étend le LF par une notation : "(x,y)" ;
- étend le LP par un nom : "couple" ;
- étend le LM par un procédé de désignation mixte "le couple (x,y)" ;

2) Cette nouvelle structure est maintenant descriptible dans le langage, au niveau mathématique.

- étend le Modèle par un procédé de comparaison entre ces nouvelles structures : "règle d'emploi que voici ..."
- effectue une comparaison pour un cas particulier, la règle plus spécifique :"...(x,y)=(y,x)..."

3) Des règles de manipulation sont définies sur cette structure, les cas particuliers intéressants sont étudiés.

- compare ce procédé de construction à un autre procédé de construction possible à partir du même contexte : "{x,y}"

4) Une nouvelle construction est comparée aux constructions environnantes pour définir sa spécificité.

- propose en remarque des digressions métamathématiques sur la construction

5) Cette construction peut être introduite à partir de constructions existantes (construction interne permettant de bâtir l'édifice) comme cela est fait traditionnellement en mathématiques.

- étend les classifications du Modèle par le prédicat de classification : "est-un-couple"

6) Après description, manipulation, évaluation, il y a reconnaissance : que peut faire le mathématicien lorsqu'il s'est donné un tel objet, peut-il accéder aux constituants ?

- étend les moyens de désignation du Modèle en réifiant les dépendances entre constituants
- caractérise les deux dépendances possibles comme étant fonctionnelles : "en vertu de..."

7) La validité de l'accès aux constituants est prouvée.

- étend le LP par deux schémas de langage voisins : "x est la première projection..."
- étend le LF par deux schémas de langage voisins (en l'occurence un symbole de fonction alors que cette notion n'existe pas encore au niveau mathématique) : "$x=pr_1(z)$,..."

8) Les accesseurs sont définis afin d'être utilisables au niveau mathématique.

La construction de couple est maintenant devenue autonome, car le mathématicien a considéré tous les usages possibles de cette structure. Elle s'apparente à la **construction d'une structure de donnée** informatique sophistiquée, comme par exemple les Types Abstraits Algébriques. Les méta-actions observées sont :

- la création d'une nouvelle "structure-objet" mathématique (méta-action principale) ;
- le prédicat de classification et les procédés de nommage qui lui sont subordonnés ;
- les moyens de comparaison qui lui permettent d'entrer en relation ;
- un procédé de construction interne, qui permet de justifier cette abstraction et sa cohérence en restant au niveau de la théorie mathématique.

Plus généralement, l'introduction d'une définition ou d'un théorème mathématique correspond à un regroupement de méta-actions constituant la structure d'organisation macroscopique de base des traités mathématiques. Des études linguistiques plus nombreuses permettraient de classifier les structures de cette organisation. L'objectif est d'aboutir à **des modélisations informatiques fournissant des schémas prédéfinis**, et des connaissances pragmatiques aidant à les instancier en fonction des données.

5.3.4. Les conséquences dans une théorie ensembliste intuitive

La suite des définitions étend cette structure en introduisant, de façon descendante, un ensemble dont les éléments sont définis par le prédicat nouvellement créé. Cette continuation est schématisée figure 3.38 :

- étend les classifications du Modèle par le prédicat de classification "est-un-graphe"

9) Cela correspond à une action de cohérence de la théorie ensembliste intuitive du Modèle.

- caractérise l'existence de deux ensembles : "X" et "Y"
- étend le LF par deux schémas de langage semblables aux précédents (la distinction se fait sur le rôle des arguments dans le contexte) : "$X=pr_1(G)$,..."

10) Les liens entre un ensemble et ses éléments sont exploités pour établir des propriétés sur les ensembles : ici la décomposition d'un objet vérifiant le prédicat est-un graphe.

- étend le Modèle en généralisant le procédé de construction, mais comme la structure de couple suffit, il la réutilise pour bâtir ses extensions : "triplet... quadruplet, etc, etc..."

11) Il y a généralisation de la structure définie afin de satisfaire à son rôle intuitif : n objets peuvent être identiques, une structure permet de les distinguer.

- utilise les procédés de déduction du Modèle pour dire que la cohérence impose l'existence d'un ensemble Z caractérisé (dépendance) par X et Y
- introduit la structure implicitement définie par cette dépendance (produit cartésien)

12) L'existence au niveau des ensemble d'un réseau de liens présentant les mêmes propriétés que celui établi au niveau des éléments, permet d'affirmer la création d'un nouvel ensemble à partir de deux autres

ensembles. Cela justifie l'introduction d'une nouvelle structure, réifiant ce processus d'exploitation des dépendances.

L'utilisation des liens ensemblistes

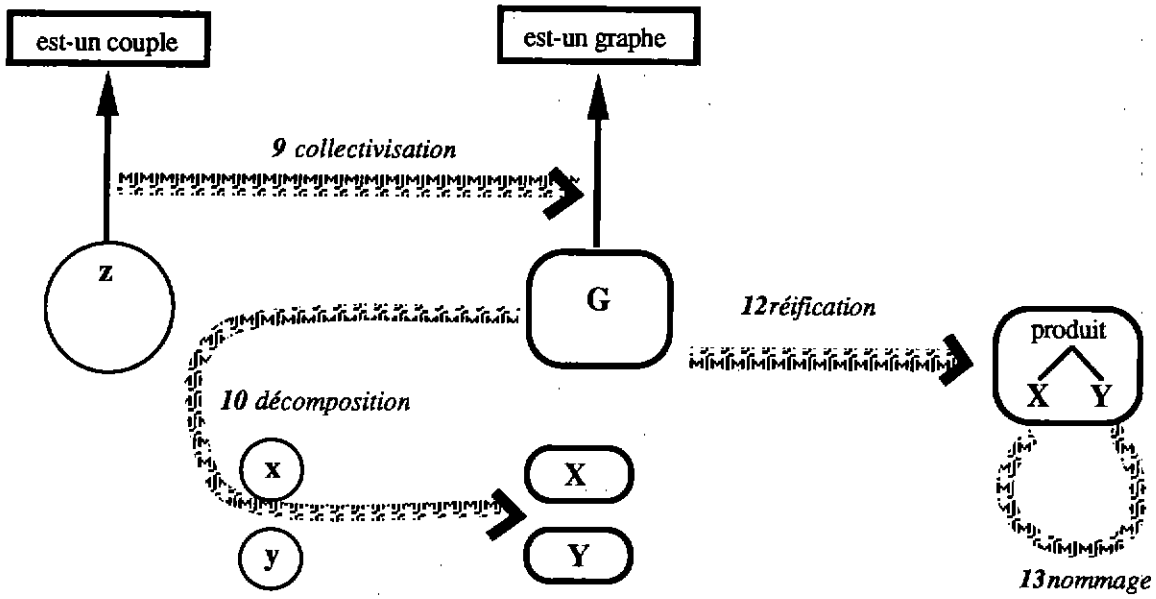


figure 3.38

Cette session de définitions, représentant deux pages d'un traité de mathématiques, est caractéristique du travail du mathématicien lors de sa **phase de constitution** de connaissances. Elle nous permet de dégager les résultats suivants :

- la création d'une structure nouvelle, ou plus généralement l'introduction d'un nouveau concept, définit les moyens nécessaires à toutes les situations d'utilisation. Cela permet de déterminer les définitions à introduire et leur signification.
- le processus d'introduction effectue un parcours minutieux permettant de justifier la validité et d'élaborer les conséquences mathématiques.

Nonobstant l'intérêt épistémologique de l'étude de ces mécanismes, la **phase d'utilisation** des connaissances peut se contenter d'une modélisation de l'état final de ce processus de chargement (bootstrap). Nous proposons d'adopter le paradigme des structure de données pour bâtir la modélisation. Cela permettrait d'organiser les informations identifiées lors de cette session, en définissant ainsi leur nature et leur rôle dans la structure. La difficulté principale reste néanmoins l'organisation à adopter à l'échelle de ces structures.

Dans l'optique d'une utilisation courante, la notion de n-uple est utilisée sans être définie par construction à partir de la notion de couple. Il y a donc modélisation directe de la notion de n-uple avec n entier identifié (comme 2 ou 101), et de celle de n-uple avec n-entier quelconque.

Par ailleurs, l'usage mathématique a entériné l'utilisation - presque abusive - de la notion de couple comme ingrédient de structuration : par exemple, "On appelle groupe un couple formé de..." (p113). Il semble plus utile, pour la modélisation des connaissances, d'introduire la notion de structure typée et étiquetée permettant de grouper les informations nécessaires à l'utilisation de ce "couple".

Nous proposons donc une **approche plus "informatique"** que celle adoptée dans l'activité du mathématicien, pour ce qui concerne la modélisation des structures de données et des connaissances manipulées. Ces restrictions ne remettent pas en cause l'importance de l'étude des traités mathématiques.

5.4. LES STRUCTURES D'ORGANISATION DE LA REPRESENTATION

5.4.1. Les langages à objets pour la représentation des connaissances

D'après notre étude des exemples partie I, le langage et l'activité mathématique dépendent de l'organisation générale des connaissances qu'ils utilisent. Nous avons choisi une approche "à objets" pour la représentation des connaissances mathématiques car elle permet une **structuration décentralisée, diversifiée et multi-niveaux** des connaissances. Il est alors possible de simuler certaines capacités d'association des mathématiciens et d'introduire des mécanismes d'exploitation élémentaires proches de la pratique des mathématiciens.

Ce modèle favorise un **traitement dirigé par les données disponibles**, et cette approche ascendante est fondamentale pour étudier et maîtriser la complexité des possibilités. Les mathématiciens ont bâti pour cela des mécanismes d'abstraction à partir des données. Notre objectif est d'élucider les concepts pertinents afin de définir un langage qui puisse décrire les activités mathématiques à un niveau d'abstraction approprié.

Une représentation permet de faciliter les raisonnements en organisant les relations de dépendances telle la spécificité d'un concept par rapport à un autre (*sorte-de*), l'appartenance d'un objet à une classe (*est-un*), ou la décomposition d'une structure en parties (*partie-de*). Ce type de raisonnement est spécifique à toute classification des connaissances.

Nous n'allons pas détailler les principes des langages à objets [Masini 89, Meyer 88, etc.]. Par contre, nous discutons pour notre usage leurs principaux atouts et leurs limites. Le paradigme objet se partitionne en fonction de son objectif :

- une approche Génie Logiciel visant à améliorer le code et sa conception ;
- une approche Intelligence Artificielle visant à la modélisation des connaissances.

Les principes restent communs, mais les choix de définition des langages sont plus sophistiqués pour la modélisation des connaissances. Nous présentons ainsi brièvement quelques thèmes importants :

- l'abstraction d'informations ;
- la modularité ;
- le partage d'informations et de comportements communs ;
- l'organisation des langages à objets ;
- la description conceptuelle et la description de la représentation.

- **l'abstraction d'informations** :
  De mêmes informations permettant des usages très variés, ce principe d'abstraction est commun aux langages à objets. Il permet de décrire des fonctionnalités en masquant les informations et les traitements qui ont été utilisés. Cela permet, par exemple, d'obtenir le contenu d'un nombre complexe sous forme usuelle, comme couple de réels, comme matrice, comme point du plan ou comme racine d'une équation réelle. Un apport intéressant de ROME [Carré 89] consiste à raffiner aussi ce procédé de sélection en fonction de la nature du client qui utilise la fonctionnalité. Cela particularise un mode d'utilisation que l'on peut alors réifier afin de l'organiser à son tour.

- **la modularité** :
  Il y a deux grandes formes de modularité dans l'organisation : celle qui consiste à bâtir une structure par composition d'autres structures (par ex. les formules), et celle qui consiste à bâtir une structure par référence à une structure commune (par ex. le paramétrage). Ces deux formes sont nécessaires en mathématiques. La première permet l'aggrégation d'informations en une structure plus complète. La dernière permet la création de classes d'objets capables d'engendrer des objets par instanciation de certains paramètres. Elle permet aussi une organisation entre ces classes par des liens d'héritage.

- **le partage d'informations et de comportements** :
  Cette modularité se heurte en mathématiques à l'hétérogénéité des traitements (rendant leur organisation difficile), à la multiplicité de situations (qui nécessite de combiner plusieurs principes et d'étudier chaque cas), et enfin à l'aspect dynamique comme des informations à compléter, à réorganiser ou dont il faut organiser l'accès. Une difficulté notable est la mise à plat d'informations (oubli de certains aspects) et l'utilisation de critères sémantiques (échappant à une structure directement observable). Il nous semble alors indispensable de décrire explicitement en quoi une sous-classe se différencie d'une classe immédiatement supérieure.

- **l'organisation des langages à objets :**
  A la suite de P. Cointe, de nombreux langages français de représentation des connaissances ont adopté une organisation réflexive complexe où tout est objet (les classes sont des objets ayant aussi un rôle pour l'organisation). Cela permet notamment de mieux particulariser le langage aux spécificités d'une application, en particularisant — même dynamiquement — l'organisation et les mécanismes de propagation. De nombreux travaux ont, par ailleurs, porté sur la nature des liens possibles (cf. [Motschnig 90]) et la sémantique de leur héritage.

- **la description conceptuelle et la description de la représentation :**
  3-KRS et [Ferber 89] introduisent une distinction particulièrement pertinente entre la description conceptuelle, chargée d'organiser le monde décrit, et la description de la représentation, chargée d'organiser les structures de données qui permettent la description. Les classes de la description conceptuelle correspondent à un ensemble d'éléments décrits en intention, tandis que les classes de la description de la représentation regroupent des structures similaires.

Cette distinction évite de biaiser inconsidérémment la description conceptuelle. La difficulté principale des langages de représentation des connaissances demeure néanmoins : leur adaptabilité rend le choix de la représentation à adopter encore plus délicat. Nous abordons, dans la suite de ce chapitre, des problèmes généraux liés à la description conceptuelle des mathématiques.

Nous postulons que les connaissances mathématiques présentent une complexité d'organisation spécifique (due à leur maturation séculaire et à leur ambition universelle). Il semble alors important de réfléchir spécifiquement sur les problèmes liés à la description conceptuelle pour les mathématiques. Néanmoins, l'étape déterminante consistera à choisir un langage à objets évolué pour affiner et mettre en œuvre une telle description, puis évaluer les limites des choix ainsi effectués.

5.4.2. Les fonctions des manipulations ensemblistes

Les catégories grammaticales mathématiques sont utilisées comme moyen de typage d'objets mathématiques. On définit ainsi R (l'ensemble des nombre réels) comme une catégorie grammaticale, elle-même élément de la catégorie grammaticale des ensembles. Ces catégories sont regroupées dans un treillis modifiable incrémentalement et dynamiquement pour introduire par exemple, la catégorie des fonctions trigonométriques entre celle des fonctions périodiques et celle des fonctions paires ou impaires.

Le propre d'une catégorie est de pouvoir regrouper les moyens de manipulation des objets de cette catégorie. On obtient ainsi le schéma présenté ci-dessous (partiellement illustré par le cas de la catégorie des ensembles) :

I CATEGORIE GRAMMATICALE :
nom : Ensemble
sous catégorie de : Structure
sur-catégorie de : Ensemble ordonné, Ensemble fini, etc.

II REPRESENTATION DES OBJETS DE LA CATEGORIE :

III OPERATIONS :
III-1 CREATION :
objets prédéfinis :
$\emptyset < N < Z < Q < R < C$; B(ooléen)
constructeurs de structures :
construction en extension : Objets→Ensemble
produit cartésien : Ensemble×Ensemble→Ensemble
produit cartésien généralisé : Ensembles→Ensemble
nième puissance de : Ensemble×Entier→Ensemble
parties de : Ensemble→Ensemble
fonctions de : Ensemble×Ensemble→Ensemble
construction par réarrangement :
construction en compréhension : Ensemble×Relation→Ensemble
union :
intersection :

complémentaire :
différence :
différence symétrique :

III-2 MANIPULATION :
test (et leur négation) :
inclus? : Objet×Ensemble→Booléen
contient? :
appartient? :
vide? :
disjoints? :
égaux? :
même-cardinalité? :
observation :
cardinalité : Ensemble→EntierInfinitaire
génère-éléments :
type :
"déconstructeurs" :

III-3 RELATIONS:
jeux de constructeurs élémentaires :
relations de définition :
relations :

Cette description d'un ensemble s'apparente à celle des Types Abstraits Algébriques. En revanche, elle n'a pas besoin d'être formellement vérifiable puisqu'il ne s'agit pas de gestion automatique mais d'une modélisation à des fins d'assistance. Les champs de description d'une catégorie sont soulignés. Nous avons indiqué certains noms de fonctions spécifiques aux ensembles, sans indiquer leur type ni leur définition. Les fonctions de création ont pour codomaine cette catégorie Ensemble, les fonctions de test ont pour codomaine les booléens B, le reste constitue les fonctions d'observation.

Les constructeurs de structures créent des ensembles inatteignables autrement. Par exemple le produit-cartésien prend deux ensembles E et F et crée un nouvel ensemble : le produit E×F. Cette construction n'est pas unique, puisque la nième puissance peut donner E×E. Elle peut être redondante, puisque l'ensemble des parties de E correspond à l'ensemble des fonctions de E dans B. Par ailleurs, la construction en extension peut procéder de plusieurs façons (crée-singleton, ajoute-élément, transfo-suite-ensemble, etc.).

Cette description de la catégorie comprend donc des équivalences et des redondances multiples. Pour la partie III-3, il n'y a pas alors une, mais des façons de choisir un jeu de constructeurs élémentaires. Les relations de définition permettent alors de définir les autres fonctions à partir de ces constructeurs. Les autres relations sont par exemple "E⊂E∪F" ou "EΔF=(E∪F)-(E∩F)".

La multiplicité des situations mathématiques impose qu'une fonction procède différemment selon la nature des informations disponibles sur ses arguments. Par exemple "A⊂B" n'est pas vérifiable sans information : si A et B sont définis en compréhension, le problème est équivalent à "$P_A(x) \Rightarrow P_B(x)$", s'ils sont définis en extension, il suffit de passer au crible les éléments de A. Mais le plus souvent, le mathématicien ne peut apporter des réponses que sous conditions : si leurs cardinalités sont disponibles, alors "A⊂B" est faux si card(A)>card(B) ; de même, si B est un doubleton contenant b, le mathématicien peut faire des hypothèses en fonction de A.

Notre description de la catégorie des ensembles n'est alors qu'un inventaire de fonctionnalités que la modélisation doit être capable de synthétiser. Chaque fonction dispose d'une ou plusieurs définitions qu'elle **gère selon la nature des informations disponibles**. La représentation des ensembles comprend alors les informations que le mathématicien est susceptible d'utiliser pour ses déductions. Un aspect important de l'activité mathématique est que le mathématicien dispose souvent d'**informations incomplètes** sur un objet. Les fonctions doivent alors aussi permettre de diagnostiquer un "échec" vu le manque d'informations disponibles, et proposer éventuellement des hypothèses pour le contourner.

Le procédé mathématique usuellement utilisé pour définir ces fonctions est de se ramener à la construction en compréhension, {x∈E/ P(x)}, pour raisonner sur les prédicats. Ce procédé est indispensable, mais l'enrichissement progressif consiste à le remplacer autant que possible par des raisonnements sur les abstractions introduites. Par exemple, utiliser directement "{x}={y}⇔x=y", "E=F⇔E⊂F∧F⊂E", les cardinalités, etc. Les fonctions qui composent cette catégorie sont alors progressivement enrichies par ces

nouvelles relations et leurs savoir-faire associés. Par ailleurs, ces fonctions sont spécialisées et augmentées pour rendre compte des sous-catégories (par ex. intervalles-d'entiers).

Il apparaît ici encore que les connaissances permettant de définir et de manipuler le concept d'ensemble "à la façon du mathématicien" soient considérables. Plus généralement, la description est soit caricaturale, avec peu d'opérateurs et des formes normales très restrictives, soit très complexe. Dans les deux cas, la description des espaces de Hilbert semble lointaine...

Cela vient renforcer l'intérêt d'une ***spécification incrémentale*** des catégories et objets. Pour arriver à une telle description, il faut, comme le mathématicien, introduire et enrichir incrémentalement les fonctions disponibles. Cela implique aussi que les fonctions de la description soient des entités autonomes et spécialisables, et qu'une catégorie soit un point de vue constituant une synthèse macroscopique.

5.4.3. Les manipulations d'éléments

Après avoir étudié comment créer un nouvel ensemble, et comment permettre leurs manipulations (telle l'égalité), nous nous intéressons maintenant à la manipulation de leurs éléments. Pour rendre compte de la diversité des objets qualifiés d'éléments d'un ensemble, nous distinguons :

- les **objets génériques**, issus d'une caractérisation descendante de l'ensemble : on a un ensemble, on manipule donc l'un de ses élements, variable ou quelconque fixé.
- les **objets identifiés**, issu d'une caractérisation ascendante de l'ensemble à partir de ses éléments, tels l'entier "0" ou la valeur de vérité "faux".

Des exemples d'objets génériques sont :
"x", "2 Arccos(x)", "(x,f(x))", "f", "E".

Des exemples d'objets identifiés sont :
"0", "0,0000003", "3,14", "π", "2 Arccos(0)", "(0,π/2)", "10^6-1", "N".

Cette distinction correspond à deux sortes de manipulations ensemblistes : l'ensemble par collectivisation et l'ensemble des éléments. L'ensemble des éléments est une définition extensionnelle. Elle correspond intuitivement à la juxtaposition des parties. Par construction, désigner une telle juxtaposition est équivalent à désigner simultanément tous les éléments. L'ensemble par collectivisation est une définition intensionnelle qui représente les façons de caractériser des éléments de l'ensemble. Intuitivement, cela correspond à des manipulations au niveau de l'ensemble vu comme objet abstrait et non comme ensemble de ses éléments. Le tout est ici réifié et la juxtaposition des parties passe en arrière plan.

Ces deux façons d'envisager un ensemble sont bien sûr fortement dépendantes. Toutefois les manipulations mathématiques autorisent à s'intéresser aux éléments d'un ensemble non spécifié, ou aux propriétés des éléments d'un ensemble alors que les dits éléments restent indéterminés. Lorsque tout est bien déterminé, cette dualité correspond à un choix d'interprétation, comme le sous-entend le mot "intensionnel". La gestion dynamique de cette dualité est toutefois l'une des caractéristiques importantes de l'activité mathématique.

Pour représenter cela, nous proposons de considérer un ensemble comme deux classes informatiques associées : la classe des objets génériques et la classe des objets identifiés, l'association s'effectuant par une fonction de concrétisation qui permet le transfert d'un élément générique en un élement identifié lorsque les informations deviennent suffisantes.

Cette concrétisation se traduit par un choix d'interprétation explicite de la part du mathématicien ou par des conséquences de la cohérence entre relations mathématiques. Pour illustrer cela, prenons l'exemple de l'ensemble E des réels dont le carré vaut 4. C'est parce que le domaine d'interprétation est suffisamment maitrisé que nous sommes à même d'établir la correspondance avec l'ensemble composé de 2 et -2. Sans cela, et nous avons déjà abordé les causes d'imperfection d'une modélisation, un objet générique de E aurait une interprétation imprécise sur les réels, voire serait identifié au réel 2 !

Nous proposons de distinguer selon la nature de l'interprétation pour appréhender la notion de variable. Nous avons repris pour cela les notions d'**informations incomplètes, imprécises et incertaines**, notions fondamentales pour la modélisation des connaissances. En utilisant le paradigme logique, les objets identifiés sont ceux dont l'interprétation est connue dans le Modèle. Les objets

génériques sont ceux dont l'information sur leur interprétation est incomplète[6] :

- Une variable est alors un objet générique imprécis : il représente tous les éléments et aucun en particulier.
- Un élément "quelconque fixé" est un objet générique incertain : il représente un seul élément de la zone d'interprétation, mais son interprétation a été explicitement "suspendue" et l'information disponible est insuffisante pour la déterminer.

Cela permet d'éliminer les quantificateurs entre relations. Il faut ajouter à cela les informations sur la dépendance fonctionnelle avec d'autres objets génériques, la portée de la liaison du symbole, la gestion des interprétations par hypothèse, le rôle dans le problème, etc. Nous proposons de regrouper les informations concernant un élément générique en créant un objet informatique comprenant :

- son procédé de désignation ("y")
- l'expression mathématique qui la définit ("f(x)")
- les informations sur son interprétation (type réel, imprécise, localité dans le problème, etc.)
- les relations dans lesquelles il intervient (il dépend de f et x, il vérifie "y>0", etc.)

La classe des objets génériques réels inclut alors la description et la gestion de ces informations, ainsi que les procédures permettant l'étude de la correspondance avec des objets identifiés. Elle comprend de plus les propriétés propres à l'ensemble des réels. De façon générale, des sous-classes sont créées lorsqu'il y a changement de nature par isomorphisme (entiers-réels) ou lorsque les propriétés sont suffisamment fortes pour faire apparaître un savoir-faire spécifique à leur traitement.

Par exemple, les entiers ont notamment pour sous-classes les entiers-premiers, les entiers-multiples-de-n, qui a elle-même pour sous-classe les entiers-pairs. Ces sous-classes comprennent notamment des procédés de reconnaissance et de génération (de quelques élément identifiés, de tous les éléments identifiés, d'un élément générique à partir d'un autre, etc.).

La classe des objets identifiés réels décrit, quant à elle, les façons possibles de définir un réel identifié comme "π" (procédé de désignation, structure de donnée contenant une valeur et sa précision, procédé de génération de la partie décimale, limite d'une suite, etc.) ainsi que les propriétés caractéristiques dans lesquels il intervient. Cette définition peut être utilisée pour déterminer des objets identifiés composés comme "$\pi^2/2$". Par ailleurs, la description des réels doit pouvoir être utilisée pour toute structure faisant intervenir des réels.

5.5. L'EXPLOITATION DES CONNAISSANCES

5.5.1. Finalités de l'organisation

L'activité mathématique est considérée ici comme un raisonnement en présence **d'informations incomplètes**. Une caractéristique des mathématiques est que le mathématicien se contente parfois d'informations très pauvres et utilise des informations "d'ordre supérieur" faisant intervenir, non plus l'objet, mais seulement des relations ou la nature de l'objet.

C'est le cas, par exemple, des raisonnements sur la cardinalité. De même, la connaissance d'une fonction ne se borne pas à connaître les couples de valeurs qui la définissent. En présence d'indétermination sur le premier élément, cette connaissance n'est pas forcément exploitable pour l'identifier à l'aide du second. En revanche, en sachant que la fonction est injective, la connaissance du deuxième élément suffit pour déterminer le couple.

Pour modéliser des aspects très élémentaires de ce raisonnement, nous proposons de chercher des mécanismes d'exploitation des liens entre objets introduits en fonction des connaissances disponibles. Il y a alors une "compilation" de relations (ou formules) en un réseau de dépendances entre objets. Ce processus cherche simplement à simuler artificiellement un comportement plausible du mathématicien[7].

---

[6] cf. [Beynon 90] pour une étude générale de la notion de variable. Pour les notions d'informations incomplètes, imprécises et incertaines, cf. par ex. [Prade 84] ; cf. aussi les raisonnements sur des informations incomplètes dans la Revue d'intelligence artificielle, vol.2 n°3,4/1988, bien qu'à notre connaissance aucune étude de l'activité du mathématicien n'ait été abordée sous cet angle.

[7] Notons que simuler le raisonnement que l'enfant développe durant la résolution de problèmes arithmétiques est déjà un problème psychologique et informatique difficile [Darcel 87].

Pour montrer l'influence des connaissances disponibles, comment le mathématicien peut-il exploiter un problème contenant "$A \cap B$" ? S'il connaît des relations entre opérateurs ensemblistes (dont l'inclusion), il peut tenter de les utiliser. Même s'il doit faire intervenir les éléments, il peut utiliser par exemple "$A \cap B \subset C$" et "si $E \subset F$ alors $x \in E \Rightarrow x \in F$" pour montrer que "$e \in A \cap B \Rightarrow e \in C$". Il n'est donc pas forcément obligé de faire appel à la définition : "$A \cap B = (x / x \in A \wedge x \in B)$". Par ailleurs, il est parfois utile de faire appel à des représentations intuitives, comme l'a fait [Merialdo 79] pour certains types de relations ensemblistes.

L'exploitation des connaissances disponibles ne produit pas seulement des déductions, elle produit aussi des conjectures et des hypothèses à vérifier. Par ailleurs, [Wolstencroft 89] relève et illustre quatre sortes de contraintes de similarité en analogie :

- structurelles (c.-à-d. syntaxiques) ;
- sémantiques (c.-à-d. sens voisins) ;
- organisationnelles (c.-à-d. proches en mémoire associative) ;
- pragmatiques (c.-à-d. rôles proches dans l'action).

Un réseau de dépendances entre objets nous semble utile pour favoriser des mécanismes basés autrement que sur la forme syntaxique des formules. Ainsi :

Etant données les relations $A \subset A'$, $B \subset B'$, $A \cap C = D$, $f(A) = D$, que peut-on dire sur $A \cap B$?

Des réponses sont par exemple :

- si $B = C$, alors $A \cap B = D$;
- si $B \subset C$, alors $A \cap B \subset D$;
- si $B \cap C = D$, alors $D \subset A \cap B$;
- $f(A \cap B) \subset f(A) \cap f(B)$;
- $A \cap B \subset A' \cap B'$.

Il serait intéressant de produire de telles réponses, car elles peuvent faciliter le raisonnement du mathématicien. Nous avons donc cherché un moyen de regrouper les relations de telle façon que tout objet puisse accéder simplement aux relations qui le concernent, que nous nommerons ***propriétés*** de l'objet. Nous cherchons de plus à abstraire autant que possible le regroupement de propriétés, de façon à ce que, par exemple, l'ensemble des parties de R tire ses propriétés de R et de l'opérateur "ensemble des parties de". Nous escomptons enfin qu'il sera possible de dégager des mécanismes d'exploitation de ces propriétés.

5.5.2. Le rattachement des propriétés

Le problème posé consiste alors à définir l'emplacement d'une propriété dans l'organisation conceptuelle, ses modes d'accès et ses mécanismes d'exploitation. Ce problème est crucial en présence d'une grande quantité de connaissances à organiser.

Les propriétés mathématiques sont non dupliquées : partager cette information revient à en définir ses diverses voies d'accès. Savoir où une propriété est accessible est l'une des caractéristiques majeure de l'organisation des connaissances mathématiques. Elle permet de disposer d'heuristiques de recherche puissantes basées sur la prise en compte des informations pertinentes à une tâche et sur leur intérêt potentiel d'utilisation. Il est, par exemple, possible de raisonner seulement sur l'existence d'un chemin reliant deux objets par le biais de propriétés.

Considérons alors les relations vraies suivantes :

$A \subset B$ (A et B étant des ensembles donnés)..... (1)

$A \subset A \cup B$ (A et B étant universellement quantifiés)..... (2)

$\emptyset \subset A$ (A étant universellement quantifié)..... (3)

La relation (1) faisant intervenir deux ensembles A et B donnés, elle permet de préciser les liens entre A et B à l'aide de l'opérateur "est inclus dans". Cette propriété peut donc être intéressante pour tenter des déductions à partir de A ou B. Elle sera qualifiée de propriété de l'objet A (resp. B). Toutefois, elle n'apporte aucune information pertinente sur l'opérateur "est inclus dans", et à ce titre ne doit pas être accessible par ce biais.

En revanche la relation (2) apporte une information sur un lien entre "est inclus dans" et "union" (les symboles A et B ne jouant qu'un rôle langagier). A ce titre un accès depuis ces deux opérateurs est utile. Plus généralement, une propriété comportant des variable liées dans laquelle un opérateur intervient, sera qualifiée de propriété de l'opérateur et accessible depuis cet opérateur.

De même la relation (3) est une propriété de l'opérateur "est inclus dans" et de l'objet "Ø". Les formules (2) et (3) présentent toutefois une dissymétrie dans le rattachement des propriétés. La formule (2) concerne plus l'opérateur "est inclus dans" puisque cette propriété ne sera utile que si l'on cherche une inclusion. Par contre, l'opérateur "union" ne servira de point d'accès principal (sans se préoccuper de l'inclusion) que lors d'un mécanisme associatif rare. De même dans la relation (3), "Ø" sera l'opérateur principal et l'accès par l'opérateur "est inclus dans" ne servira que lorsqu'on recherche (au moins) un ensemble inclus dans un autre donné.

Selon cette analyse préliminaire, une propriété ne sera rattachée (ou directement accessible) qu'à un petit nombre de ses constituants dits **constituants principaux**, qui constitueront les clés d'accès à la propriété. Le choix de ces accès sera effectué à l'aide de **règles de précédence**, traduisant l'usage empirique d'organisation des connaissances mathématiques. Ces règles définissent un ordre partiel parmi les objets mathématiques. L'ensemble des objets maximaux d'une formule forme ses constituants principaux. Ainsi la formule "$\sin^2x+\cos^2x=1$" possède deux constituants principaux "sin" et "cos" et la définition de tangente à l'aide de sinus et cosinus en comporte trois.

Nous abordons maintenant le point de vue d'un objet. Afin de les organiser et de les rattacher au niveau d'abstraction adéquat, les propriétés d'un objet mathématique sont décomposées en :

- propriétés **individuelles**, spécifiques à l'objet, qui le mettent en relation avec des objets identifiés ;
- propriétés **sociales**, mettant en relations plusieurs objets génériques (imparfaitement connus). Toute information sur l'un peut donc servir à préciser les autres. Un objet sans propriété sociale est dit *indépendant*, sinon, il appartient à un ensemble de *objets dépendants* ;
- propriétés **génériques**, qui précise le comportement individuel ou social de l'objet, non pas en tant qu'entité spécifique, mais en tant que membre d'une classe d'appartenance ;
- propriétés **collectives**, valables pour les objets qui sont aussi des classes. Elles référencent les relations définies sur cette classe.

Ainsi, les informations concernant un ensemble E quelconque fixé sont par exemple :

- valeurs : $A \cup B$ (l'égalité est donc traitée à part et non comme propriété) ;
- propriétés individuelles : $e \in E$, $E \cap C = \emptyset$ (C est donc un objet identifié) ;
- propriétés sociales : $E \neq F$ (F est donc un objet générique) ;
- propriétés génériques : E est inclus dans l'ensemble des chaînes de l'alphabet A ;
- propriétés collectives : loi de composition interne $\perp$;

L'emplacement de rattachement d'une propriété intervient lors de la consultation et l'ajout de propriétés. Les propriétés individuelles et sociales d'un objet, lui sont naturellement rattachées. Les propriétés génériques et collectives sont rattachées ou ajoutées à sa classe, plus précisément aux classes les plus générales possibles vérifiant ces propriétés. L'usage de ces propriétés fait donc appel à un mécanisme d'héritage entre les classes.

5.5.3. L'exploitation de propriétés

Prenons maintenant un exemple fictif pour illustrer une activité mathématique possible. Il s'agit d'un sous-problème ouvert, consistant à comparer deux ensembles déjà introduits E et F. **Comparer** revient à trouver des relations visant à préciser ces deux objets. Certaines relations sont des tautologies qui n'apportent aucune information (par ex. "$\emptyset \subset E$"). D'autres sont entachées de "bruit parasite" (par ex. l'introduction de G, de caractéristiques imprécisées, vérifiant "$F \cap G = E$" pour simplement exprimer que "$E \subset F$"), et peuvent être simplifiées sous une forme plus "concise". Les informations utiles sont qualifiées d'*information de comparaison.*

Une première démarche consiste à regarder si des informations de comparaison sont déjà présentes et si l'un des deux objets est parfaitement connu, afin de pouvoir disposer d'informations précises pour positionner le second. Faute de telles informations, une recherche d'information peut être décidée. L'une des sous-tâches consiste à comparer l'appartenance respective d'éléments caractéristiques. Une autre sous-

tâche consiste à comparer les cardinalités de E et F. Ces sous-tâches proviennent d'heuristiques permettant une première évaluation, et utiles pour d'autres tentatives comparatives.

Les seules informations disponibles sur F concernent ses éléments : f, g et h sont des éléments de F, construits à partir de relations (valides) du problème. Un savoir-faire de gestion de la cardinalité peut alors déduire que "card(F)≥1". En effet seul l'examen plus détaillé des constructions de ces éléments peut éventuellement préciser s'ils sont distincts ou non. De même, la connaissance que F ne comporte pas d'autre élément nous permet de déduire que "1≤card(F)≤3". La justification de cette information est mémorisée, ainsi que les conditions à vérifier pour la raffiner : "distincts?(f, g, h)"

L'examen de E nous apporte plusieurs informations, la sous-tâche de cardinalité étant menée conjointement à deux niveaux : celui des ensembles et celui de leurs éléments. Comme e et e' appartiennent à E, "card(E)≥1" ; de plus E est égal à "A∪B" et possède d'autres propriétés. La recherche d'information au niveau des ensembles consiste alors à utiliser une propriété de l'opérateur de cardinalité relativement à l'union. On obtient, par exemple, la relation (4) et sa forme simplifiée (5) :

$$\forall E,F \; ; \; card(E \cup F) = card(E) + card(F) - card(E \cap F) \quad \ldots\ldots (4)$$

$$\forall E,F \; ; \; card(E) \leq card(E \cup F) \quad \ldots\ldots (5)$$

Dans le cas de propriétés conditionnelles comme (6) :

$$\forall E,F \text{ disjoints}; \; card(E \cup F) = card(E) + card(F) \quad \ldots\ldots (6)$$

une sous-tâche peut être lancée pour décomposer "A∪B" en ensembles disjoints "A-B", "B-A" et "A∩B", et par ce biais trouver les relations (4) et (5). Si aucune propriété de cardinalité concernant l'union n'est disponible, une propriété peut être cherchée par combinaison de relations. La connaissance de relations comme (7) ouvre une possibilité de relier cardinalité et union. La recherche d'information sur l'opérateur de cardinalité relativement à l'intersection peut alors aboutir sur (8) :

$$\forall E,F \; ; \; E \subset E \cup F \quad \ldots\ldots (7)$$

$$\forall E,F \text{ tels que } E \subset F \; ; \; card(E) \leq card(F) \quad \ldots\ldots (8)$$

Une synthèse de (7) et (8) peut alors être effectuée pour aboutir à la relation (5). Dans tous ces cas, comme "E=A∪B" dans le problème, cette recherche permet de relier la cardinalité de E à celle de A ou B. Une recherche d'information sur la cardinalité de A et B peut être lancée. Si A est infini, le système peut tenter d'exploiter cela pour E, et aboutir à la conclusion que E est infini.

Ce procédé permet de considérer l'activité mathématique comme un processus de recherche, de propagation et de déduction d'information. Il permet d'exploiter des informations partielles mais suffisantes pour la tâche cherchée. Plusieurs voies sont souvent possibles et dépendent, entre autres, des connaissances générales disponibles. Quelques conclusions possibles sont alors (classement selon un critère d'intérêt croissant, bien que la plupart des conclusions ne soient pas comparables par l'implication) :

- E et F n'ont pu être comparés ;
- $F \cap G \subset E$;
- $E \neq F$;
- e' = f est un élément commun à E et F;
- E est infini ;
- card(F) = 1, soit f = g = h;
- $F \subset E$;
- $F \subset A$ d'où $F \subset E$;
- $F \cup \{j, k\} \subset E$;
- $E = \{e, e'\}$ et $E \neq F$;
- E = F ;
- informations complètes sur les éléments de E et F, d'où toutes les comparaisons potentielles où interviennent E et F sont possibles ;

La notion d'information est donc relative, une connaissance, c'est-à-dire une information accessible par le système, peut être :

- plus ou moins **informée**, relativement à une théorie. Ainsi (4) permet de déduire (5) moyennant des relations plus fondamentales, alors que (5) ne permet pas de déduire (4). La relation la moins informée est alors redondante. La plupart des informations ne sont pas comparables.

- plus ou moins **directe**, relativement à une théorie et ses mécanismes de déduction. Les notions de coût et de complétude peuvent alors être définies. Par exemple (5) est plus difficile à déduire à partir de (4) qu'à partir de (7) et (8). Il est alors judicieux de disposer de (5) si sa déduction s'avère problématique.
- plus ou moins **adaptée**, relativement à une tâche, théorie et mécanismes de déduction. Ainsi (5) suffit pour montrer que si A est infini, E l'est aussi. De même, la décomposition des entiers en facteurs premiers est plus adaptée que leur valeur pour le calcul du pgcd.

Lorsque les connaissances s'appliquent à un objet identifié (et donc "connu" a priori), elles sont aussi exhaustives que la modélisation de la base de connaissances le permet. Par exemple, certains grands nombres premiers ou certaines décompositions en facteurs premiers peuvent être mémorisées, les autres devant être calculées par une méthode rattachée à la classe de tous les nombres entiers.

Pour mener à bien une telle exploitation des connaissances, nous escomptons la compilation des relations en un réseau de dépendances entre objets. Cette modélisation et son mécanisme d'exploitation restent à préciser, en vue d'obtenir un processus réflexe et de mettre en œuvre des raisonnements intuitifs spécialisés. L'activité élémentaire d'exploitation des connaissances et de manipulation d'objets mathématiques peut notamment être modélisée par quatre processus microscopiques :

- *la recherche* de relations pertinentes ;
- *l'introduction* de nouveaux objets ;
- *l'accumulation* des propriétés d'un objet ;
- *le compactage* de propriétés, qui vise à améliorer l'information disponible ;

La recherche de relations vise à explorer une partie de la modélisation. L'accumulation organise les propriétés d'un objet et permet ainsi une vérification partielle de leur cohérence. Le compactage cherche principalement à préciser les objets génériques en procédant selon deux modes. Le plus direct exploite la classe d'origine de l'objet dans la modélisation, en cherchant à identifier l'objet générique comme objet générique d'une sous-classe, ou mieux comme objet identifié. Sinon le compactage examine les propriétés de l'objet et leurs conséquences possibles pour chercher à le préciser via ses dépendances.

Comme nous l'avons noté pour la cardinalité, la modélisation joue un rôle clef dans le guidage de l'activité. Soit "n" l'objet générique entier premier, à préciser, que faire lorsque le raisonnement conclut par ailleurs que "n est pair" ? L'accumulation introduit cette propriété, et le compactage va chercher à déterminer si ces informations sont corrélables. Comme aucun lien direct n'existe entre "premier" et "pair", elle lance une recherche de relations qui développe l'une, puis l'autre, des définitions présentes dans la modélisation.

$$n \text{ est-pair ssi } \exists k,\ n=2*k \quad \ldots\ldots (9)$$

$$n \text{ est-premier ssi } \forall k,\ k \mid n \Rightarrow (k=1 \vee k=n) \quad \ldots\ldots (10)$$

La définition (9) est directement exploitable par introduction d'un nouvel objet générique m. L'accumulation se charge alors d'introduire que n=2*m. Pour tenter d'utiliser (10), le compactage cherche à relier la divisibilité avec la multiplication. La recherche produit la définition (11).

$$k \text{ divise } n \text{ ssi } \exists l,\ n=k*l \quad \ldots\ldots (11)$$

Le compactage disposant d'une propriété basée sur la même structure multiplicative, l'applique pour 2 et m. Elle accumule dans m la propriété (m=1 ∨ m=n). 2 étant un objet identifié, l'accumulation se traduit par une vérification de la propriété (2=1 ∨ 2=n), ce qui provoque l'identification de n avec 2. Les objets intervenant dans les propriétés de n sont alors compactés en utilisant n=2. L'objet m est alors identifié à 2, et oublié puisqu'il n'est plus référencé par le problème.

Un tel processus de raisonnement pourrait être effectué sur des objets mathématiques considérés comme agents autonomes lors d'une tentative de créativité contrôlée. Son intérêt principal est d'explorer la collaboration entre raisonnements de natures différentes (symbolique dirigé, réflexe, résolution d'équations, intuitif, etc.).

Pour conclure ce chapitre, l'organisation et la modélisation de connaissances mathématiques est un problème essentiellement prospectif. Nous avons dégagé des besoins allant d'une modélisation cognitive jusqu'à une approche exclusivement informatique. Nous avons illustré la nature, la diversité et la quantité de connaissances, en nous focalisant sur la description "très littérale" du domaine, indépendamment des connaissances de résolution. Nous avons ainsi constaté que la **modélisation du domaine** était l'une des causes majeures de limitation des systèmes informatiques.

6. OPERATIONNALISATION DES FORMULES

6.1. L'USAGE DES FORMULES ET DES IDENTITES

6.1.1. Un exemple d'utilisation typique en mathématiques

La constitution de Mathématiques Assistées par Ordinateur offre, à disposition immédiate du mathématicien, un éventail de connaissances du domaine beaucoup plus complet que ce qui est humainement mémorisable. La **sélection et l'utilisation des informations** pertinentes à l'état courant du problème représente alors un enjeu essentiel.

Il y a notamment une différence importante entre la donnée d'une formule et son utilisation comme règle pour une opération de déduction. Nous étudions cette différence dans le cas d'identités remarquables sur la sommation, en postulant que les principes que nous développons s'appliquent aussi aux théorèmes plus complexes à décrire.

Nous développons les besoins à partir de l'étude d'un exemple concret de résolution de problème, issu de [Graham 89, p180] (figure 3.39). Il s'agit d'une autre preuve du problème déjà présenté partie II (cf. la figure 2.8). La stratégie couramment adoptée par le mathématicien consiste en une **analyse comparative entre les données du problème et les formules connues**, entraînant un raffinement progressif des tâches à effectuer.

L'analyse des données du problème indique que l'expression à simplifier a pour caractéristique principale d'être une sommation utilisant des coefficients binomiaux. Les connaissances relatives aux sommations, aux coefficients binomiaux, et à leur usage simultané sont alors activées. Pour ce qui est des coefficients binomiaux, elles comprennent notamment les identités de la table 174 (figure 2.8) et celles de la table 169 (figure 3.39). Une fois cette caractérisation générale effectuée, la qualification du contenu de ces tables est comparée avec la caractérisation des données du problème.

Les identités de la table 169 sont caractérisées comme une sommation incluant un produit de deux coefficients binomiaux. Les lois élémentaires de manipulation de Σ indiquent que ces coefficients sont liées à la variable d'index (connaissance pragmatique concernant la manipulation des expressions non liées). Ces identités seraient rejetées pour notre problème sans **une heuristique ou un oracle** indiquant que "k" est une expression mathématique équivalente à un coefficient binomial (propriété d'un cas spécifique des coefficient binomiaux : $\binom{n}{1} = n$). Ici, la structure prime sur la valeur mathématique !

Cette idée trouvée, l'utilisation d'une formule de la table devient possible et nécessite une analyse comparative plus détaillée. Le **modèle d'analyse des formules** présenté en partie I s'applique alors, en commençant par la sommation à simplifier dans le problème :

- *zone d'intérêt* le terme principal du Σ ;
- *structure prégnante* le produit des coefficients binomiaux jusqu'à leurs arguments ;
- *thème d'intérêt* la présence ou l'absence de la variable d'index.

Le *contenu caractéristique* obtenu par cette analyse donne :

$$\sum_{xxxx} \boxed{\binom{\text{OUI}}{\text{NON}}\binom{\text{OUI}}{\text{NON}}}$$

Parmi les trois identités de la table 169 ayant cette structure prégnante, seule l'identité (5.26) a un contenu caractéristique identique. Les autres identités sont écartées car déclarées incompatibles pour la tâche en cours. Le contenu caractéristique est alors raffiné par une analyse détaillée ayant pour thème d'intérêt le signe de la variable d'index dans les deux zones la contenant. Le contenu caractéristique résultant est alors :

pour la sommation à simplifier $\binom{\text{NEG}}{}\binom{\text{POS}}{}$

pour l'identité (5.26) $\binom{\text{POS}}{}\binom{\text{NEG}}{}$

322

Problem 4: A sum involving two binomial coefficients.

Our next task is to find a closed form for

$$\sum_{k=0}^{n} k\binom{m-k-1}{m-n-1}, \qquad \text{integers } m > n \geq 0.$$

Wait a minute. Where's the second binomial coefficient promised in the title of this problem? And why should we try to simplify a sum we've already simplified? (This is the sum S from Problem 2.)

Well, this is a sum that's easier to simplify if we view the summand as a product of two binomial coefficients, and then use one of the general identities found in Table 169. The second binomial coefficient materializes when we rewrite k as $\binom{k}{1}$:

$$\sum_{k=0}^{n} k\binom{m-k-1}{m-n-1} = \sum_{0\leq k\leq n} \binom{k}{1}\binom{m-k-1}{m-n-1}.$$

And identity (5.26) is the one to apply, since its index of summation appears in both upper indices and with opposite signs.

But our sum isn't quite in the correct form yet. The upper limit of summation should be $m-1$, if we're to have a perfect match with (5.26). No problem; the terms for $n < k \leq m-1$ are zero. So we can plug in, with $(l, m, n, q) \leftarrow (m-1, m-n-1, 1, 0)$; the answer is

$$S = \binom{m}{m-n+1}.$$

This is cleaner than the formula we got before. We can convert it to the previous formula by using (5.7):

$$\binom{m}{m-n+1} = \frac{n}{m-n+1}\binom{m}{m-n}.$$

Similarly, we can get interesting results by plugging special values into the other general identities we've seen. Suppose, for example, that we set $m = n = 1$ and $l = q$ in (5.26). Then the identity reads

$$\sum_{0\leq k\leq l} (l^2 - k^2) = \binom{2l+1}{3}.$$

The left side is $(l+1)l^2 - (1^2 + 2^2 + \cdots + l^2)$, so this gives us a brand new way to solve the sum-of-squares problem that we beat to death in Chapter 2.

The moral of this story is: Special cases of very general sums are sometimes best handled in the general form. When learning general forms, it's wise to learn their simple specializations.

Table 169 Sums of products of binomial coefficients.

$$\sum_k \binom{r}{m+k}\binom{s}{n-k} = \binom{r+s}{m+n}, \qquad \text{integers } m, n. \quad (5.22)$$

$$\sum_k \binom{l}{m+k}\binom{s}{n+k} = \binom{l+s}{l-m+n}, \qquad \text{integer } l \geq 0, \text{ integers } m, n. \quad (5.23)$$

$$\sum_k \binom{l}{m+k}\binom{s+k}{n}(-1)^k = (-1)^{l+m}\binom{s-m}{n-l}, \qquad \text{integer } l \geq 0, \text{ integers } m, n. \quad (5.24)$$

$$\sum_{k\leq l} \binom{l-k}{m}\binom{s}{k-n}(-1)^k = (-1)^{l+m}\binom{s-m-1}{l-m-n}, \qquad \text{integers } l, m, n \geq 0. \quad (5.25)$$

$$\sum_{0\leq k\leq l} \binom{l-k}{m}\binom{q+k}{n} = \binom{l+q+1}{m+n+1}, \qquad \text{integers } l, m \geq 0, \text{ integers } n \geq q \geq 0. \quad (5.26)$$

figure 3.39

Une première *micro-tâche* consiste à réduire cette différence en permutant les deux coefficients binomiaux (l'utilisation de la commutativité et de l'associativité est un automatisme associé à des représentations intuitives, même si la validité d'une telle opération peut être vérifiée en faisant appel explicitement à ces propriétés). L'appariement entre les arguments des coefficients binomiaux de la sommation à simplifier et ceux de l'identité est alors effectué. Il y a donc retour à la zone d'intérêt précédente en tenant compte de la permutation des coefficients. Le thème d'intérêt est alors l'appariement entre les variables rencontrées, en parcourant successivement les arguments.

Le contenu caractéristique obtenu est ainsi (l, m, n, q) ← (m-1, m-n-1, 1, 0). Sous ces conditions, et modulo la permutation des coefficients, les deux termes principaux sont rendus identiques. Il reste à comparer les bornes du Σ et vérifier les ***conditions de validité*** concernant les variables. Pour vérifier que l>m≥0 dans l'identité, le mathématicien cherche si m-1>m-n-1≥0 dans le problème, sachant que par hypothèse m>n≥0 dans le problème.

Stupeur ! Parallèlement à ces conditions de validité, survient un **problème** plus sérieux : l'identité est applicable pour 0≤k≤m-1, tandis que la sommation du problème porte sur 0≤k≤n. Le mathématicien a donc recours à une identité importante de la sommation :

$$\sum_{j=p}^{q} a_j + \sum_{j=q+1}^{r} a_j = \sum_{j=p}^{r} a_j \quad \text{.....(1)}$$

Cette identité est utilisée ici avec (p, q, r) ← (0, n, m-1) afin d'isoler la première sommation. Le mathématicien doit alors étudier (mentalement ou pas) le terme :

$$\sum_{k=n+1}^{m-1} \binom{k}{1} \binom{m-k-1}{m-n-1}$$

Il constate que m-k-1<m-n-1, c'est-à-dire que le deuxième coefficient est nul, par convention des coefficients binomiaux utilisés dans ce contexte de définition généralisé. Le mathématicien devait donc penser à comparer les deux arguments du coefficient lors de son analyse mathématique de la formule, pour parvenir à conclure que cette sommation était nulle. Il y a donc une **expertise de vérification de "normalité"**, incluant la vérification des conditions de validité, l'absence de cas "dégénéré" et la prise en compte des contraintes que cela engendre. Cela peut être modélisé par une tâche de fond systématique.

Après synthèse de ces tâches, le résultat cherché est alors le membre droit substitué de l'identité (5.26). Cette substitution s'accompagne, au niveau général de la preuve choisie, par une simplification mathématique. Le suivi général de cette preuve par un mathématicien confirmé, ou sa construction par un mathématicien "expert", se contente de la **visualisation simultanée des résultats suivants** :

- la sommation du problème ;
- l'identité (5.26) ;
- le changement de variable à effectuer (à partir du moment où il est construit) ;
- la solution finale S=..., construite éventuellement argument par argument.

6.1.2. Une démarche de raisonnement générale

Cette façon mathématique de procéder est fréquente. Elle est tellement utile et caractéristique, qu'il est indispensable de l'automatiser "presque entièrement" pour envisager des Mathématiques effectivement Assistées par Ordinateur. La session précédente peut alors être analysée pour dégager les besoins informatiques qu'elle nécessite.

La modélisation de la **démarche de raisonnement** utilisée est alors essentielle. Ce type de raisonnement se décrit par la tentative de mise en œuvre du plan suivant :

1) trouver une identité applicable ;
2) effectuer son application ;
3) aviser selon la tâche englobante du problème.

Toute l'expertise relative à ce plan très succint provient du savoir-faire déployé pour sa gestion et sa réalisation. Ce plan correspond à une démarche issue des souhaits de la partie II :

1) trouver <relation à instancier> pour réaliser <effet> ;
2) utiliser <relation à instancier> pour transformer <formule> ;
3) gérer les tâches de résolution de problème.

Nous proposons une démarche d'assistance à l'application d'un plan, incluant une **analyse parallèle** des données du problème et des connaissances du système.

La démarche d'application de l'étape 1) consiste à décrire les données à transformer afin de définir ce qui est recherché, puis à le comparer avec la description des identités disponibles. Ces descriptions respectives sont progressivement affinées jusqu'à la **sélection d'une identité applicable**. Ce raffinement s'accompagne d'une focalisation des centres d'intérêt du mathématicien. Notre exemple peut alors se résumer ainsi :

description sur l'exemple	**caractérisation des données**	**caractérisation associée**
$\sum$ et $( )$	description du contexte de définition du problème	contexte mathématique
$\sum_{xxxx}$ $\boxed{a_k ( )}$	description qualitative du terme principal du Σ	groupement de connaissances choisi selon cette description (table 169)
$\binom{OUI}{NON}\binom{OUI}{NON}$	description qualitative selon le thème d'intérêt choisi	identités correspondantes (seule l'identité 5.26 convient)

Nous avons ainsi ici trois niveaux de description permettant des identifications de plus en plus fines, jusqu'à la sélection d'une identité applicable permettant de tenter son application dans l'étape 2) :

- une identification contextuelle générale ;
- une identification relative aux tables utilisables ;
- une identification relative aux identités.

Ce **raffinement successif** se fait en "Z". Par exemple, la description qualitative du terme principal du Σ est choisie car le contexte mathématique commun aux sommations et aux coefficients binomiaux comprend des tables d'identités triées selon la nature du terme principal. Comme aucune identité ne correspond directement à cette description, le mathématicien a un sous-problème à résoudre : tenter d'adapter une des identités.

Il y a plusieurs solutions en général (nous en avons étudié deux). La solution retenue ici consiste à considérer "k" comme un coefficient binomial, et sélectionner ainsi la description de la table 169. Le raffinement vers les identités se fait alors en choisissant la variable d'index comme thème d'intérêt. Ici encore, la correspondance souhaitée engendrera un sous-problème menant à la permutation des coefficients binomiaux.

L'étape 2) d'application de l'identité sélectionnée, débute lorsqu'une seule identité reste envisagée. Elle se recouvre avec la fin de l'étape 1) : elle utilise en général la zone d'intérêt courante pour commencer l'appariement entre variables, selon un parcours des arguments basé sur la structure prégnante. L'instanciation d'une identité fait appel à la résolution de nombreuses équations et inéquations mathématiques simples. Elle débouche ici sur un nouveau problème à la charge de l'étape 3).

6.1.3. Besoins élémentaires des Mathématiques Assistées par Ordinateur

Comme cela a été annoncé partie II, et revu pour cet exemple, la sélection et l'utilisation d'informations mathématiques par un système informatique requiert une **expertise d'analyse et d'utilisation de formules.**

Nous avons ainsi un **mécanisme de sélection général** relativement proche des représentations intuitives du mathématicien. L'informatisation de ce mécanisme requiert :

- les **critères pertinents pour la description** de la formule à différents niveaux et pour son raffinement successif (connaissances liées au domaine de travail) ;
- des **procédés de comparaison** aptes à utiliser cette description, à engendrer des sous-problèmes et à guider les raffinements.
- la capacité de **résoudre des sous-problèmes** dirigés par la différence entre description actuelle et description souhaitée ;
- l'**intervention du mathématicien** pour les choix, les points d'embarras ou la récupération d'échecs.

L'utilisation d'une identité par instanciation sur une formule connue est, pour sa part, à automatiser quasi-totalement. Cela requiert :

- un mécanisme d'**appariement** entre expressions mathématiques ;
- un mécanisme de **résolution** d'équations et inéquations mathématiques simples ;
- la **récupération d'échecs** : explicitation du problème pour une résolution difficile, et surtout, détection de la cause d'échec et génération de conditions qui rendraient l'instanciation applicable.

Cette robustesse de récupération d'échecs est primordiale en mathématiques. Dans notre exemple, l'instanciation "boite noire" de l'identité (5.26) amènerait à un échec cuisant en ne s'appliquant que dans le cas singulier du problème où n=m-1. La structuration de notre analyse des formules permet de diagnostiquer que l'instanciation se déroule parfaitement, jusqu'à la prise en compte de la borne supérieure du Σ.

Il y a alors soit échec, soit cas particulier, soit sous-problème engendré pour tenter d'utiliser l'instanciation avec une borne différente de celle des données. Une continuation automatique, ou un recours au mathématicien, sont alors nécessaires. Cela engendre une démarche de raisonnement similaire...

Les besoins élémentaires sont aussi exprimables en fonction des connaissances spécifiques au domaine, nécessaires pour cette mise en œuvre. Hormis la description des concepts, nous distinguons ici :

- pour ce qui concerne les **relations mathématiques** :
  - des identités "courantes" concernant l'addition, la soustraction, la multiplication, la division, l'élévation à la puissance, la factorielle, etc.
  - des identités singulières courantes, comme $1 = 0! = x^0 = (-1)^{2n}$, etc.
  - des identités importantes, comme (5.26) ou l'identité (1) concernant les bornes du Σ.
  - des identités singulières importantes, comme $\binom{n}{1} = n$, $\sum_{j=n}^{n} a_j = a_n$, etc.
- pour ce qui concerne l'**exploitation de ces relations** :
  - l'exploitation heuristique d'identités singulières importantes, comme introduire un Σ ou un coefficient binomial à partir de a_n, etc.
  - la détection de cas singuliers et de conditions de validité, comme la détection de la nullité d'un coefficient binomial dont le premier argument est inférieur au second ;
  - la qualification heuristique des possibilités et des difficultés de résolution en fonction de la description ;
- pour ce qui concerne le **savoir-faire général** concernant l'analyse des formules :
  - critères pertinents de description des formules ;
  - détermination des zones d'intérêt prioritaires ;
  - thèmes d'intérêt de comparaison de zones sous influence d'un Σ (les termes liés).

6.2. L'ANALYSE D'UNE IDENTITE

6.2.1. La nature d'une identité

Nous voulons assister les tâches élémentaires d'utilisation d'une identité, à partir de son expression sous une forme externe par le mathématicien. Comme les systèmes de raisonnement interviennent après l'analyse d'une formule, nous considérons que la structure mathématique de l'identité est déjà disponible. Mais alors, que faire ? Considérer qu'il s'agit d'une **règle de réécriture** à utiliser en remplaçant éventuellement les variables par des sous-arbres ? Avant d'étudier cette possibilité "de base", voyons comment procèdent les systèmes de raisonnement.

Les identités remarquables sont aussi des règles expertes.

Une identité remarquable telle que l'écrit le mathématicien n'existe pas ! Rien d'étonnant d'après R.J. Fateman, qui décrivait onze façons de démontrer que "$\cos^2 x + \sin^2 x = 1$" avec MACSYMA, alors que le système n'avait aucune connaissance de cette identité [Fateman 85]. Il semble que cela reste valable pour les systèmes de raisonnement où certaines identités ne sont connues que sous forme de règle d'expertise.

Par exemple, si l'identité "$\int \cos x = \sin x$" est connue de CAMELIA [Vivet 84], "$\int f + \int g = \int f+g$" n'est connue que par la règle procédurale d'un plan d'action :

> Pour calculer la primitive de f+g :
> calculer la primitive de f, calculer la primitive de g, puis les ajouter.

Par cette distinction, Martial Vivet distingue entre un savoir sûr ou non dans l'aboutissement du calcul de la primitive. Cette nuance, à visée "automatique", nous concerne peu pour l'assistance. L'assistance s'intéresse par contre à la **production de telles règles à partir des identités mathématiques**, en utilisant des connaissances destinées à les rendre opérationnelles.

Par ailleurs (p 55), M. Vivet note la distance entre la formulation de la distributivité du produit sur la somme : "(a+b)c = ac+bc", et le diagnostic conduisant à son utilisation :

> Il faut être pleinement conscient que cette propriété est utile dans un sens, toutes les fois où l'on a besoin de ramener une somme à un produit (factorisation ...) et dans l'autre sens, toutes les fois où l'on a besoin d'éclater un produit pour faire apparaître une somme.

Dans ces deux cas, les règles produites peuvent être définies à partir de la structure prégnante des identités. L'usage opératif des identités remarquables provient alors de deux types d'expertise distincts :

- Une **expertise d'analyse** qui est capable, connaissant les concepts du domaine, de chercher à utiliser une identité (même nouvelle, même fausse), et de fournir les règles précédentes.
- une **expertise d'exploitation**, issue de l'expérience acquise lors de la résolution de certains types de problèmes du domaine. Par exemple, l'importance de l'intégration par parties n'est pas directement déductible de l'énoncé de l'identité, mais provient des méthodes développées autour de son expérience d'utilisation.

Le rôle de l'expertise d'exploitation peut être schématiquement considéré comme utiliser, pondérer, compiler et gérer des résultats produits par l'expertise d'analyse, à des fins de résolution de problème. L'expertise d'exploitation est alors plus concernée par le guidage subtil de la résolution de problème, que par les ressources élémentaires utilisées pour sa mise en œuvre.

Nous pensons alors qu'il est possible - et nécessaire - d'isoler une expertise d'analyse suffisamment générale pour permettre l'assistance au mathématicien. Nous considérons que l'expertise d'exploitation éventuellement disponible proviendra de notre approche expérimentale (par ex., le mathématicien introduit son expérience d'utilisation pour telle situation). Nous nous intéressons dorénavant à l'expertise d'analyse.

Les fonctions de base à assurer, une fois une identité connue, sont de :

- savoir détecter que son application est possible ;
- décrire l'effet de son application, de façon plus ou moins intuitive et détaillée.

Il est alors possible de proposer des solutions efficaces pour :
- savoir quand l'application de cette identité est souhaitable ;
- exprimer comment tenter de se ramener à ses conditions d'application.

Il est en effet possible de dégager une expertise d'utilisation basée sur l'analyse des identités. M. Vivet cite par exemple le théorème : "lim f+g = lim f + lim g (lorsque x tend vers a)" et décrit son savoir-faire d'utilisation (p 56) :

> SI j'ai à calculer la limite : lim F(x)
> SI F(x) apparaît comme une somme f(x)+g(x), ALORS calculer lim f, calculer lim g puis faire la somme des résultats obtenus.
> SI F(x) n'est pas une somme (et que je ne sais pas quoi faire d'autre) ALORS tenter de faire apparaître F(x) comme une somme (cf. ce qui a été dit sur la distributivité ci-dessus) puis appliquer la démarche précédente.

Ce type d'expertise apparaît comme relativement général, et pourrait par exemple s'appliquer tout aussi bien pour le calcul de primitives. La seule différence notable consiste en la pondération du dernier cas (SI F(x) n'est pas une somme), qui semble plus intéressant pour les limites que pour le calcul intégral (où l'intégration par parties et les autres méthodes développées sont prépondérantes).

Ce dernier cas montre enfin que cette expertise doit utiliser une stratégie fins-moyens afin de générer des sous-problèmes tels "tenter de faire apparaître F(x) comme une somme". Cette stratégie fins-moyens est de portée générale. Elle s'applique par exemple pour faire apparaître un produit dans l'intégration par parties de :

$$\int \text{Log } x \, dx = \int 1 \cdot \text{Log } x \, dx = x \text{ Log } x - \int x \, \frac{1}{x} dx = x \text{ Log } x - x.$$

Cette expertise d'analyse est donc **couplée à des capacités générales de résolution de problème**. L'intérêt de son explicitation est de mettre en évidence des motivations "profondes" du mathématicien, afin de savoir les remettre en cause en cas d'insuccès. Par exemple, l'expertise précédente suppose que décomposer une structure facilite la résolution. Il est parfois possible d'utiliser l'égalité dans l'autre sens, si la somme amène une simplification. La présence de l'identité est donc primordiale, afin de pouvoir envisager toutes les possibilités.

6.2.2. L'état des lieux

Force est de constater que cette expertise d'analyse et d'utilisation n'est pas directement étudiée dans les systèmes de démonstration existants. Les règles expertes proposées sont déjà un résultat compilé pour l'utilisation, au détriment de la flexibilité d'usage des identités et d'une facilité d'interaction étendue (guidage, apprentissage, explication, etc.). Ici encore, cela se traduit par un langage d'expression trop spécialisé. Tous les systèmes de raisonnement ont néanmoins développé des langages sophistiqués pour exprimer leurs règles expertes.

De nombreuses règles expertes sont décrites dans [Nicaud 87] en introduisant trois niveaux de langage, illustrés ici pour la règle "transfo/a2-b2" (p 81-82) :
- une description globale
  factoriser une différence de deux carrés
- un pseudo-langage
  Si l'expression du problème contient une sous-expression f qui est une différence de deux carrés dont les racines sont r et s, avec le coefficient c
  Alors créer un problème équivalent en réécrivant f sous la forme c(r-s)(r+s)
- le langage SIM
  si (pb **exp** ee)(ee **s-terme** *chemin* ch *est* f) (f **a2-b2** *rac1* r *rac2* s *coef* c)
  alors (**pbsuiv** *exp* "&c(r-s)(r+s)" *chemin* ch)
  propriete *effet* facto *concept* a2-b2

Cette règle est complétée d'une autre règle, correspondant aux cas où la sous-expression f est une somme contenant plus de deux termes. Le langage SIM est le langage effectivement utilisé par APLUSIX. La qualité exemplaire d'APLUSIX nous permet d'apprécier les abstractions qu'il reste à introduire pour exploiter directement l'identité mathématique selon la description globale :

utiliser "$a^2 - b^2 = (a+b)\ (a-b)$" pour factoriser <formule>.

Il faut pour cela :
- **séparer** l'identité des règles annexes concernant son savoir-faire d'utilisation (prise en compte d'un coefficient c, nombre et disposition des termes, sens d'utilisation, effet recherché, etc.) ;
- **rendre explicite** la fonction de reconnaissance (et d'accès aux termes) afin de la séparer d'une utilisation particulière ;
- **caractériser** automatiquement l'effet d'une identité en fonction de son utilisation ;
- **interpréter** dynamiquement le tout, de façon à ce que l'explication de l'opération effectuée soit la description en Langage Mathématique du raisonnement suivi.

Ces propositions préconisent une meilleure séparation des traitements allant vers une déclarativité et une généralité accrue dans la prise en compte des connaissances mathématiques. Cela est indispensable dès que la quantité de connaissances (et donc d'identités) est importante.

Les systèmes actuels qui nous semblent les plus avancés dans cette voie sont MUSCADET et PRESS. Ils disposent tous deux d'un **méta-niveau** leur permettant de raisonner directement sur les identités mathématiques.

MUSCADET résoud des problèmes difficiles dans des domaines conceptuels complexes. Il dispose d'une méta-action qui génère dynamiquement des règles en fonction de l'identité introduite [Pastre 84, p 144]. A partir de l'identité "$A \cap B = \{x / x \in A \wedge x \in B\}$", il génère toutes les façons élémentaires de l'utiliser, soit ici quatre règles :
- **si** $x \in A \cap B$ **alors** $x \in A$
- **si** $x \in A \cap B$ **alors** $x \in B$
- **si** $x \in A$ **et** $x \in B$ **alors** $x \in A \cap B$
- **si** il faut montrer $x \in A \cap B$ **alors** montrer $(x \in A \wedge x \in B)$

Cette méta-action se décompose en produire de nouveaux faits, et propager des buts. Il est aussi possible de généraliser ce traitement, en générant la façon d'utiliser l'identité paresseusement en fonction des besoins, et en propageant les buts après étude des relations pouvant être utilisées.

PRESS [Bundy 83] est un module algébrique à visée expérimentale. Il résoud notamment des équations grâce à une stratégie qui exploite une analyse des formules et des identités. Par exemple, il trouve la solution d'une équation du second degré à partir de :
- une analyse ayant pour thème d'intérêt le degré de la variable ;
- l'identité "$(a+b)^2 = a^2 + 2ab + b^2$" ;
- un appariement puissant associé à un résolveur d'équations simples ;
- une stratégie de résolution élaborée.

PRESS ne s'applique que pour les équations et inéquations algébriques, et se concentre sur la stratégie de résolution permettant une automatisation totale. Vu notre recherche d'assistance, nous nous intéressons à des identités plus diverses et à une stratégie locale centrée sur l'utilisation des identités.

6.2.3. Les possibilités d'utilisation

Pour répondre à ces enjeux, nous proposons de repenser le raisonnement du mathématicien à partir d'une **expertise d'analyse et d'utilisation de formules**. Nous avons pris comme exemple l'identité :

$$\sum_{j=p}^{q} a_j + \sum_{j=q+1}^{r} a_j = \sum_{j=p}^{r} a_j \quad \text{.....(1)}$$

Dans une identité, toutes les variables libres sont supposées universellement quantifiées, et leur type doit être précisé ou déduit du contexte. Le mécanisme d'analyse explicite notamment les **données mathématiques** relatives à l'identité :
- les objets variables "p", "q", et "$(a_j)_{j \in I}$" sont reconnus, typés et universellement quantifiés ;
- la définition de la sommation introduit une variable muette "j" pour chaque Σ ;
- un réseau de contraintes de définition de la sommation, qui synthétise les contraintes "$p \leq q < r$".

Nous disposons alors pour le membre gauche de la Représentation Interne suivante (figure 3.40) :

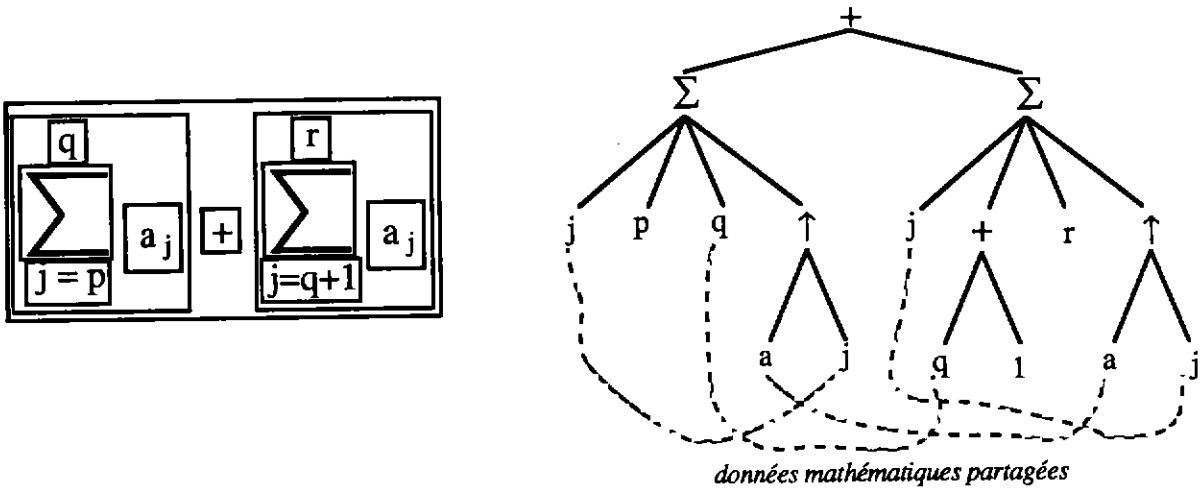


figure 3.40

Un système de réécriture utilisant cette identité dans le sens gauche-droite peut proposer plusieurs solutions :

- un **appariement structurel**, cherchant impérativement l'arbre de la structure mathématique, et valable par exemple pour :

$$\sum_{k=0}^{n} b_k + \sum_{l=n+1}^{p} c_l \quad \text{.....(a)}$$

- un **appariement entre constituants mathématiques**, valable par exemple aussi pour :

$$\sum_{k=0}^{n-1} k + \sum_{l=n}^{p} l \quad \text{.....(b)}$$

- une **unification de contraintes mathématiques**, valable par exemple même pour :

$$\sum_{k=0}^{n} b_k + \sum_{l=m}^{p} l \quad \text{.....(c)}$$

L'appariement structurel gère les équivalences de nom et les substitutions d'objets variables. Cela soulève néanmoins le problème pragmatique du nommage de la partie droite produite. Disposer uniquement de l'appariement structurel apparaît clairement comme trop rigide. Cela nécessiterait un système complexe dont l'expertise serait dans la gestion des règles aboutissant à la préparation astucieuse de la structure recherchée.

L'appariement entre constituants mathématiques gère de plus des équations et inéquations mathématiques simples (symboliques et numériques). Cette capacité tient compte du fait que "k" ou que "$(c_k+1)!$" peuvent être considérés comme des termes indicés de forme "b_k". Elle lui permet aussi de remplacer des égalités comme "q=q", par des relations mathématiques entre sous-arbres :

La borne inférieure du second Σ est égale à la borne supérieure du premier Σ plus un.

Ces possibilités permettent alors de résoudre le cas (b), voire le cas $\sum_{k=0}^{n-1} k + \sum_{l=p-1}^{p} l$, sachant que p est le successeur de n. Certaines relations mathématiques entre variables sont alors prises en compte lors de cet appariement.

L'unification de contraintes mathématiques permet de mieux **intégrer l'utilisation de l'identité dans le raisonnement mathématique**. Cette ouverture se traduit par deux possibilités supplémentaires notables :

- des relations mathématiques non gérées par le résolveur de contraintes sont susceptibles d'être prises en compte explicitement, par raisonnement ou par recours au mathématicien ;
- des hypothèses supplémentaires peuvent être générées sur les objets génériques.

Alors que l'appariement entre constituants n'aboutissait que si le système savait démontrer que "m=n+1", l'unification de contraintes aboutit s'il est possible que "m=n+1" et que cette hypothèse vaille la peine d'être envisagée. L'introduction de cette nouvelle condition est gérée dans le raisonnement mathématique, de façon à revenir dessus, par exemple, en cas de contradiction ultérieure.

Ce contrôle explicite du mathématicien permet de limiter les possibilités d'unification inutile. Par exemple, l'usage veut qu'une hypothèse comme "m=n+1", reliant deux variables indépendantes du problème, soit refusée pour permettre l'application d'une identité. Par contre, l'usage autorise une hypothèse comme "m≠n+1" afin de poursuivre le raisonnement dans presque tous les cas. L'hypothèse "m=n+1" est alors nécessaire pour traiter le cas résiduel ; elle le serait aussi, en désespoir de cause, pour trouver un cas particulier où le problème est soluble.

De même, il est toujours possible de créer artificiellement la suite "a_j" qui correspondrait exactement à l'identité. Mais l'usage refuse d'introduire de nouveaux objets sans lien pertinent avec les précédents. Par contre le cas (c) peut être résolu si les entiers de m à p sont des éléments constitutifs de la suite "(b_k)", ou s'ils constituent une extension intéressante pour le traitement du problème.

Il est donc souhaitable de disposer de **plusieurs modes d'interaction** avec le mécanisme d'appariement, afin d'autoriser des possibilités d'unification différentes selon la situation. Il est notamment utile que l'unification puisse générer des hypothèses supplémentaires et charger éventuellement le raisonnement de les vérifier. Cela requiert, par exemple, des moyens approchés comme le test d'une conjecture sur des cas particuliers ou des tentatives heuristiques pour vérifier l'inégalité de deux ensembles via leur cardinalité ou leurs éléments connus.

6.2.4. Notre proposition d'analyse

Nous cherchons à associer les identités mathématiques à leur savoir-faire d'utilisation pour les manipulations de formules. Aussi, l'expertise d'analyse qui nous intéresse ici ne concerne plus la structure mathématique et ses données associées. Elle concerne une **description de la formule permettant une caractérisation pertinente à son utilisation.**

Nous avons cherché à décomposer notre identité de façon à permettre une utilisation ultérieure aussi souple que possible. Nous parlerons d'*applicabilité* d'une identité lorsque le mathématicien cherche à savoir si elle est applicable. C'est pourquoi la **reconnaissance des formes mathématiques** de l'identité permet de produire trois éléments :

- la ***structure prégnante*** (cf. partie I) ;
- les ***contraintes d'applicabilité*** ;
- la ***description conceptuelle*** de l'identité.

La structure prégnante de l'identité (1) est : "A + B = C", telle que "A", "B" et "C" soient des "sigma" quelconques mais a priori distincts.

Les contraintes d'applicabilité sont des relations mathématiques à vérifier pour qu'une instanciation soit possible. Elles incluent notamment les conditions de validité de l'identité. Pour l'identité (1), cela donne le *réseau de contraintes* mathématiques suivant :

$$\sum_{j=p}^{q} a_j + \sum_{j=q+1}^{r} a_j = \sum_{j=p}^{r} a_j \qquad \inf(A)\leq\sup(A),\ \inf(B)\leq\sup(B),\ \inf(C)\leq\sup(C),$$

$$\inf(A)=\inf(C),\ \sup(A)+1=\inf(B),\ \sup(B)=\sup(C),$$

$$A + B = C \qquad \text{expr}(A)=\text{expr}(B),\ \text{expr}(A)=\text{expr}(C),\ \text{expr}(B)=\text{expr}(C).$$

Le fait d'introduire des contraintes mathématiques permet, par exemple, d'obtenir un même réseau avec "q-1" et "q" au lieu de "q" et "q+1".

La description conceptuelle regroupe les descripteurs utiles pour le raisonnement, comme la partie droite est factorisée, la fraction en partie gauche n'est pas sous forme réduite, la variable x est isolée en partie

gauche, etc. Une description possible de (1) est qu'elle contient des sommations, que le contexte local est additif, et que les expressions principales des sommations sont identiques.

L'application de l'identité peut alors être étendue par une analyse selon deux directions complémentaires :
- étude des zones d'intérêt A, B et C ;
- étude des variations de la structure prégnante.

L'étude des zones d'intérêt est utile car le mathématicien ne dispose que d'une ou plusieurs des zones possibles dans sa recherche d'utilisation d'identités remarquables. Il est donc utile de savoir quelles sont les contraintes entre B et les autres zones d'intérêt, ou de déterminer des propriétés de A connaissant deux zones que l'on a pu apparier avec B et C. Cette étude permet enfin de trouver les façons minimales suffisantes à une instanciation totale.

Cette étude se base sur l'analyse du réseau de contraintes. Cette analyse détermine la complexité des traitements de résolution du résolveur (ici équations, inéquations simples), et surtout elle diagnostique les données nécessaires et suffisantes à la résolution de toutes les contraintes. Cette analyse est compilée ici en trois ***schémas d'applicabilité*** pour les cas où deux des trois zones sont disponibles. Il faut recourir au réseau de contraintes pour les autres situations.

Ainsi, lorsque B et C sont disponibles et leurs relations vérifiées, le schéma d'applicabilité correspondant indique que toutes les caractéristiques de A peuvent être obtenues par les relations suivantes : inf(A)=inf(C), sup(A)=inf(B)-1, expr(A)=expr(B).

Par contre, l'analyse du réseau de contraintes montre qu'ici, une seule zone d'intérêt n'est jamais suffisante à la détermination totale des autres. Cela entraîne que l'utilisation de l'identité lorsqu'une seule zone est disponible nécessite un oracle pour déterminer le reste (sauf conditions particulières au problème).

L'étude des variations qui modifient la structure prégnante permet l'adaptation de l'identité indépendamment d'un ***mode d'application***. Ainsi, l'identité peut classiquement être utilisée en réécriture selon un mode d'application "gauche → droite" (la partie gauche, instanciée dans la formule à transformer, est remplacée par la partie droite pareillement instanciée), voire "droite → gauche". Mais cela est trop restrictif et demande une préparation presque nuisible sur la formule à transformer.

Le mathématicien sait adapter l'identité (1) à l'aide d'*identités courantes* (sur l'addition, la multiplication et l'égalité). C'est pourquoi, nous proposons de généraliser le mode d'application pour couvrir des situations telles que "- A → B - C" ou "(D - A) + E + C = F → D + E + B = F".

A partir de l'identité telle qu'elle est introduite par le mathématicien, nous proposons alors une extension du mode d'application selon des modules d'identités courantes (+, -, =, etc.), et selon des ajouts de décoration permettant par exemple d'introduire des termes inutiles. Nous qualifions le mode d'application avant ces décorations de *mode fondamental*.

Les ***conditions d'applicabilité*** de l'identité forment la synthèse de ces deux directions d'analyse. Elles constituent le test final résultant de cette reconnaissance des formes progressive. Elles regroupent ce qui est indépendant d'une formule à transformer, c'est-à-dire notamment les schémas d'applicabilité et les modes fondamentaux possibles. Ces conditions permettent de vérifier rapidement qu'une identité est applicable après l'avoir sélectionnée et appariée selon des critères de description approchés.

L'intérêt de cette approche d'analyse est qu'elle **s'applique indifféremment aux identités et aux formules du problème** que le mathématicien doit transformer. Les conditions d'applicabilité permettent d'anticiper sur la reconnaissance des formes des formules à transformer, afin d'utiliser un module comparateur exploitant une expertise d'utilisation.

Nous disposons ainsi d'un mécanisme général d'analyse d'une identité en vue de son application. Il permet, par exemple, l'application de l'identité dans le cas de notre exemple initial :

$$\sum_{k=0}^{n} \binom{k}{1}\binom{m-k-1}{m-n-1} = \sum_{k=0}^{m-1} \binom{k}{1}\binom{m-k-1}{m-n-1} - \sum_{k=n+1}^{m-1} \binom{k}{1}\binom{m-k-1}{m-n-1}$$

Le mode fondamental est ici "A → C - B". A gauche figure la formule à transformer, et à droite ce que le mathématicien cherche à produire. Les expressions déjà disponibles pour le mathématicien sont indiquées en gras : ainsi, "C" résulte de son raisonnement actuel visant à utiliser une autre identité. **Ce dont le**

mathématicien dispose se situe donc de part et d'autre de la règle de réécriture à employer. Cela constitue une autre originalité prise en compte dans notre mécanisme d'analyse.

Notre proposition permet de ne pas figer un mode d'application pour une identité ; elle cherche à mieux séparer les types de traitement (modularité) et leur contrôle (déclarativité) pour obtenir une organisation modulaire et déclarative des identités mathématiques. Elle vise à expliciter une expertise "profonde" d'analyse d'une identité à partir de sa saisie, analyse qui est actuellement à la charge de l'expert informaticien.

6.3. L'EXPLOITATION D'IDENTITES

6.3.1. Acquisition d'une nouvelle identité

Les identités mathématiques sont nombreuses et redondantes, bien que nous ayons déjà fait abstraction de leur mode d'application. Une **organisation des identités** disponibles est alors nécessaire et l'**acquisition d'une nouvelle identité** fait évoluer cette organisation.

Nous étudions ici un **dialogue d'acquisition possible pour l'identité (1)**. Ce dialogue amorce une session spéciale où l'usager saisit l'identité en marge de son problème courant, c.-à-d. il saisit la structure apparente :

$$\sum_{j=p}^{q} a_j + \sum_{j=q+1}^{r} a_j = \sum_{j=p}^{r} a_j \quad \text{.....(1)}$$

Le module d'analyse est chargé de rendre cette structure apparente utilisable dans des raisonnements mathématiques. Il indique le rattachement de (1) aux identités sur la sommation. Il considère par défaut qu'il n'y a pas de condition de validité particulière, et produit les conditions de validité provenant des conditions de définition de la sommation :

p, q, r entiers naturels

a_j expression indicée

$p \leq q < r$

L'usager introduit aussi des informations générales concernant l'identité, comme son nom et son origine (justification mathématique par une preuve, cas particulier d'une autre identité, conjecture, etc.).

Le **module d'analyse** affiche la structure prégnante produite : "A + B = C", et demande si les modes d'application possibles prennent en compte toutes les combinaisons. L'usager a ainsi la possibilité de revenir sur la structure prégnante, de limiter l'étendue des modes d'application, et de préciser la nature des simplifications à essayer après l'application. Par défaut, si l'identité utilise les modules "=" et "+ et -", et si elle s'applique quels que soient les positions des termes, elle peut transformer "(D - A) + E + C = F" en "D + E + B = F".

Les modes d'application retenus peuvent recevoir un nom spécifique et sont typés en fonction de leur déclenchement. Ainsi, le mode "droite → gauche" de (1) nécessite un oracle sur le choix de la variable "q", oracle qui peut à son tour être rattaché aux techniques de décomposition d'un intervalle entier. Ce mode se nomme : "décomposition des bornes", le mode "gauche → droite" : "regroupement des bornes", et l'identité (1) : "manipulation des bornes". L'usage de ces noms n'étant pas normalisé, il faut prévoir "d'assouplir" leur utilisation par le mathématicien.

Le module d'analyse cherche aussi les dépendances avec d'autres identités, présentes ou générées par particularisation et généralisation. Par exemple, l'identité (1) peut déjà être présente dans le cas où "p=0". Cette identité résultant d'une instanciation directe de variable, elle peut être supprimée et remplacée par une préférence par défaut.

L'une des techniques de découverte utilisée par Douglas Lenat concerne l'étude des cas singuliers. Par exemple, décréter de rendre égales deux variables x et y indépendantes ou telles que $x \leq y$. Ces techniques peuvent être utilisées en général, ou bénéficier, comme ici, de la connaissance de cas singuliers

intéressants. Les informations rattachées à la sommation donnent le cas singulier : inf(S)=sup(S), avec l'expression simplifiée équivalente faisant disparaître l'opérateur de sommation.

Le module d'analyse choisit donc d'étendre le réseau de contraintes d'applicabilité en rajoutant cette contrainte selon toutes les combinaisons possibles de Σ. Le même mécanisme d'analyse est alors utilisé dans ces nouveaux cas, et la plupart des informations sont déjà présentes. Le cas "p=r" étant incompatible, le **système propose les identités** correspondant à "p=q", "q+1=r" et à leur conjonction :

En utilisant la propriété : $\sum_{j=n}^{n} a_j = a_n$, nous obtenons les identités particulières :

$$a_p + \sum_{j=p+1}^{r} a_j = \sum_{j=p}^{r} a_j \quad \text{(cas p=q)} \qquad \text{(2)}^1$$

$$\sum_{j=p}^{q} a_j + a_{q+1} = \sum_{j=p}^{q+1} a_j \quad \text{(cas q+1=r)} \qquad \text{(3)}$$

$$a_p + a_{p+1} = \sum_{j=p}^{p+1} a_j \quad \text{(cas p=q et q+1=r)} \qquad \text{(4)}$$

Les identités (2) et (3) sont intéressantes car elles correspondent à un changement de forme important. L'usager les accepte comme identités particulières issues de (1). Elles héritent des modes de (1) et de leurs caractéristiques, tandis que le mathématicien indique les caractéristiques qui leur sont propres. Elles mémorisent de plus la nature de leur particularisation (cas "p=q" pour (2)).

L'identité (4) était déjà présente dans le système comme cas particulier de la définition récursive du Σ. Elle est conservée en ajoutant un lien de particularisation issu de (2) et (3) et donc indirectement de (1). Le fait de retrouver des identités existantes, ou d'aboutir à des identités trop simples pour être intéressantes, rend parfois inutile l'héritage des caractéristiques (c'est le cas ici).

Ce dialogue fictif prépare donc les modes d'application d'une identité, et étudie ses particularisations et généralisations possibles. Il permet donc **la structuration, l'ajout et la suppression d'identités, ainsi que leur conservation avec ajout de liens.** La plupart des actions possibles sont automatiques, mais le déroulement s'effectue par un guidage interactif du mathématicien.

Pour ce qui concerne **l'organisation inter-identités**, nous sommes concernés ici par deux types de liens :

- particularisation et généralisation ;
- précédence dans la preuve.

Une identité dépend explicitement des identités qui interviennent dans sa preuve. Pour prendre le cas simple de l'identité :

$$\frac{p}{q} + \frac{p'}{q'} = \frac{pq'+qp'}{qq'} \qquad \text{(5)}$$

Elle est justifiée à partir des identités :

$$\frac{p}{q} + \frac{p'}{q} = \frac{p+p'}{q} \qquad \text{(6)} \qquad\qquad \frac{np}{nq} = \frac{p}{q} \qquad \text{(7)}$$

Nous nous sommes particulièrement intéressés ici aux dépendances par particularisation et généralisation. Elles ne font intervenir que la formulation "résultat" de l'identité et ne présentent pas de lien direct avec la précédence dans la preuve. Par exemple (6) est une instance de (5) (après simplification...) mais (7) n'en est pas une. De plus certaines preuves produisent d'abord une identité plus générale.

---

[1] La numérotation du système est ici adaptée à celle de notre manuscrit.

Bien que plusieurs preuves soient possibles, la précédence dans la preuve permet la production d'ilots de dépendances entre identités, ainsi que des raisonnements sur la suppression d'une identité. Il est même envisageable d'exploiter un ensemble d'identités incohérentes à des fins pédagogiques, dans l'esprit de BUGGY [Brown 78], jusqu'à mise en évidence d'une contradiction.

Le raisonnement à partir des identités disponibles permet, par exemple, de conjecturer la forme de l'addition et de la multiplication de deux nombres complexes, sachant les propriétés qu'elles doivent respecter. Il est possible de développer ainsi le sens des conjectures et de leur réfutation, selon des techniques de raisonnement approché décrites notamment dans [Cipra 83]. Par exemple :

$$\frac{p}{q} + \frac{p'}{q'} = \frac{p+p'}{q+q'} \quad \ldots\ldots(8)$$

généralise l'identité sans dénominateurs et reste cohérente avec la conservation des symboles et de leur symétrie. Par contre, elle contredit (6). Reste donc à proposer d'autres conjectures, ou à tenter d'esquisser une preuve utilisant les identités disponibles.

L'essentiel de l'organisation inter-identités reste néanmoins à la discrétion du mathématicien. C'est lui qui choisit les identités dont il va disposer, et établit un compromis entre des identités très fines et des "*méga-identités*".

L'usage informatique favorise l'exploitation de méga-identités, essayant de résoudre d'un trait des situations aussi vastes que possible. La liste des identités nécessaires étant longue, cette option nécessite néanmoins la composition d'identités pour résoudre les problèmes. Le savoir-faire nécessaire à cette composition s'accommode mieux d'identités plus maniables, dont le rôle et l'effet soient caractéristiques. De plus, la composition d'identités vient spécialiser une abstraction et multiplie les cas particuliers.

Par exemple, une autre version de (5) pourrait être produite afin de prendre en compte la commutativité de l'addition, mais la description serait trop détaillée. De même, (5) pourrait aussi être étendue en :

$$\left(\frac{p}{q}\right)^{\alpha} + \left(\frac{p'}{q'}\right)^{\beta} = \frac{p^{\alpha}q'^{\beta} + q^{\beta}p'^{\alpha}}{q^{\alpha}q'^{\beta}} \quad \ldots\ldots(9)$$

mais cette multiplicité de paramètres limite l'intérêt général, et nécessite des simplifications parasites pour retrouver (5). Notons aussi que (5) et (9) "bouclent" en ce sens que chacune est déductible de l'autre par instanciation et renommage : la structure la plus simple est alors préférée.

L'existence d'un grand nombre de méga-identités ne semble pas devoir être un apport déterminant de l'informatisation des mathématiques. Il nous semble prioritaire de **nous concentrer sur le savoir-faire d'utilisation d'identités.**

Toutefois, il peut être utile de se servir ponctuellement d'une méga-identité comme solution générale d'un problème-type fréquemment utilisé. Cela permet de déterminer les cas où le problème courant est résolu, quitte à remplacer automatiquement l'usage de cette méga-identité par l'adaptation de sa preuve aux données du problème courant. Cela permettrait de bénéficier de leur usage, en préservant la compréhensibilité, grâce à l'utilisation d'identités usuelles plus fines.

Un autre intérêt d'une méga-identité consiste à capturer un savoir-faire simplificateur dans des cas particuliers. Par exemple, il est intéressant d'envisager le cas où les dénominateurs de (5) sont "q^{α}" et "q^{β}", en introduisant "ppcm(α,β)" (le plus petit commun multiple de α et β).

6.3.2. L'utilisation d'une identité pour les transformations

En raison de la diversité des situations en mathématiques, de nombreuses identités sont applicables simultanément, et à plusieurs positions, sans qu'un critère de décision simple puisse lever l'embarras.

Nous avons choisi d'analyser la formule à transformer comparativement aux identités connues, afin de déterminer les identités intéressantes, leur position dans la formule et leur mode d'application. Nous avons proposé une analyse informatique permettant de déterminer les conditions d'applicabilité d'une identité à

partir de la saisie de sa structure apparente. Nous avons, de plus, dégagé des principes généraux pour l'utilisation des identités mathématiques.

Nous présentons ainsi un mécanisme de sélection et d'utilisation d'identités susceptibles de faire progresser le problème. La figure 3.41 visualise ainsi la démarche de raisonnement générale introduite section 6.1, et détaillée pour l'utilisation de l'identité (5.26).

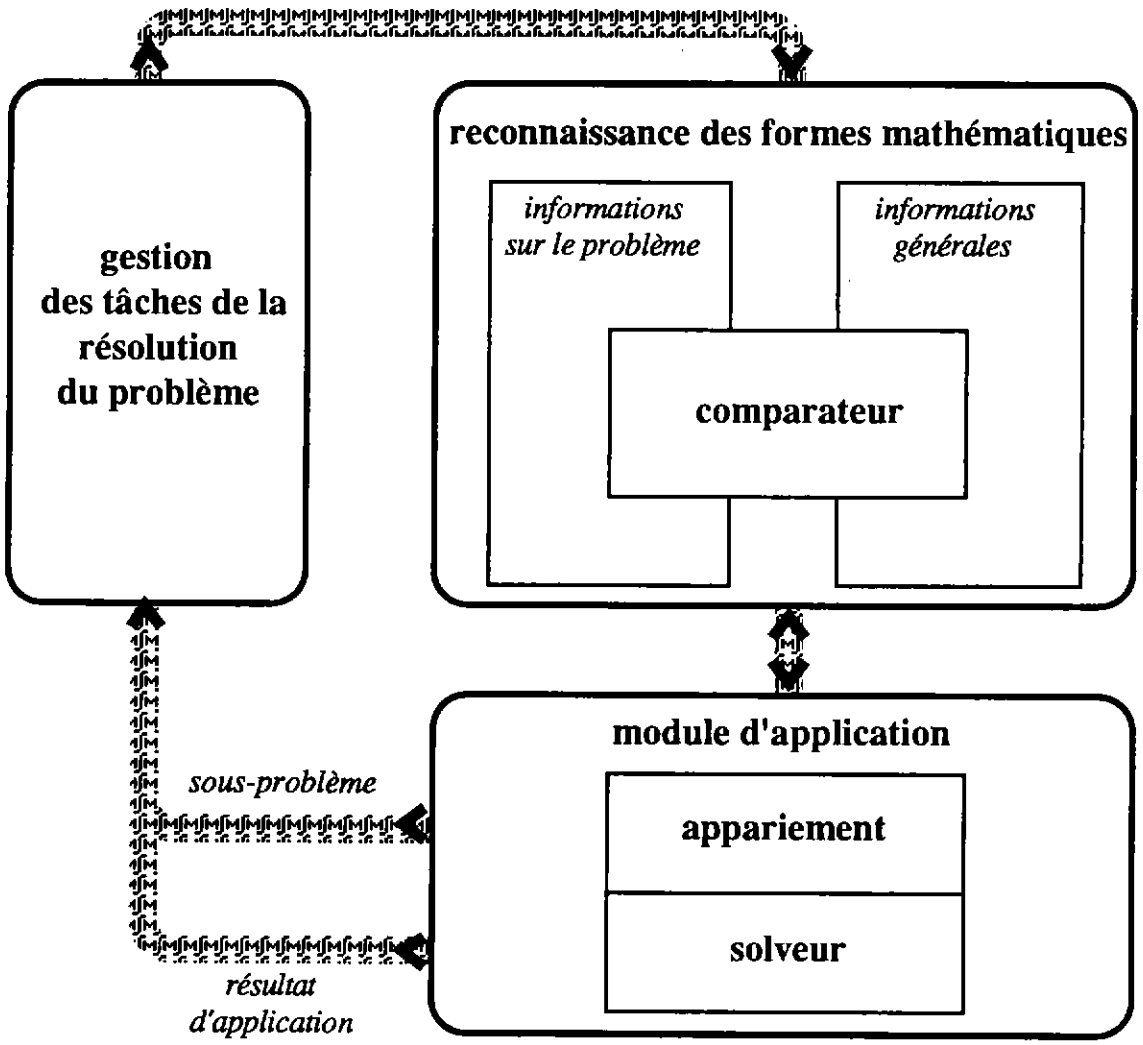


figure 3.41

La **reconnaissance des formes mathématiques** repose sur le modèle d'analyse des formules présenté précédemment. Elle compare les informations sur le problème (structure prégnante, zones d'intérêt, description conceptuelle, effet à obtenir, etc.) avec les informations générales concernant notamment les identités.

Le **module d'application** est utilisé dès qu'un appariement entre expressions mathématiques est nécessaire, ou lorsqu'un prédicat est à évaluer (vrai, faux, possible, absurde, ne sais pas). Il utilise un solveur spécialisé dans des contraintes mathématiques simples (issu par ex. de travaux comme CLP($\Re$) ou PROLOG III). Par ailleurs, il exploite les propriétés connues sur les objets et concepts mathématiques.

Le module d'application permet la récupération d'échecs de deux façons : par compte rendu au comparateur, et par appariement partiel, avec génération d'un sous-problème à traiter dans le raisonnement global. La

coopération entre ces trois modules peut être illustrée avec la démonstration par récurrence de la formule de Gauss (la somme des n premiers entiers), qui contient :

$$F(n+1) = \sum_{k=0}^{n+1} k$$

La planification (ou la suggestion de l'usager) demande le calcul de F(n+1) en transitant par F(n). Le module de reconnaissance de formes mathématiques analyse leur contenu mathématique, leurs similarités et cherche une identité qui les relie. Il a notamment le choix entre remplacer les sigma par leur définition, et utiliser une identité de décomposition des termes ou des bornes afin de relier deux sigma.

Les quatre phases d'utilisation d'une identité lors d'une manipulation de formule dirigée par l'effet à obtenir (effet issu de la planification ou d'une commande de l'usager) sont alors :

1. analyser la formule présente afin d'obtenir des objectifs de transformation ;
2. analyser et classer les schémas d'applicabilité des identités disponibles en fonction de ces objectifs (stratégie dite fins-moyens) ;
3. tenter l'application de la meilleure identité si son schéma d'applicabilité est complet ou sinon tenter de le compléter (raisonnement ou question à l'usager) ;
4. appliquer l'identité choisie comme indiqué précédemment.

Les deux premières phases sont à la charge de la reconnaissance des formes, les deux suivantes au module d'application. Ces quatre phases peuvent bénéficier du raisonnement global, ou lui rendre compte. Si une tentative d'application est décidée et réussit sur la formule, elle précise l'emplacement d'application, calcule les substitutions, génère en cas de besoin de nouvelles variables, construit la nouvelle formule et tente les simplifications prévues avec le mode d'application.

Parfois aussi, l'identité est suggérée, et il faut déterminer son lieu et mode d'application parmi certaines identités. Dans ce cas, la comparaison est facilitée mais la description fournie pour le choix de l'identité peut être conceptuelle. La technique des schémas d'applicabilité couvre ces possibilités.

Dans notre exemple, l'identité (3) est choisie pour obtenir la transformation (10), ce qui permettra de faire progresser la preuve et de rendre compte de cette progression au raisonnement global :

$$\sum_{k=0}^{n+1} k = \left(\sum_{k=0}^{n} k \right) + n+1 \quad \text{.....(10)}$$

$$\sum_{k=p}^{n+1} a_k = \left(\sum_{k=p}^{n} a_k \right) + a_{n+1} \quad \text{.....(3bis)}$$

Ici, l'identité (3bis) est présentée par le système comme justification de la transformation (10). Cela permet de détailler maintenant un dernier aspect de l'utilisation d'identités : leur utilisation pour l'**explication de la preuve**, ou plus simplement pour le suivi de cette preuve.

Hormis la possibilité de développer ou de réutiliser une preuve annexe, les capacités d'explication de la preuve dépendent grandement de la présentation des identités utilisées. Nous abordons deux facteurs :

- l'habillage mathématique de l'identité ;
- le choix d'une identité parmi des variantes plus ou moins générales.

L'**habillage mathématique** de l'identité explique les différences entre (3bis) et (3) : il y a eu permutation des membres, et choix de "k" et "n" au lieu de "j" et "q". Cet habillage utilise des noms de symboles visant à faciliter l'appariement par le mathématicien. Il choisit des symboles identiques dès que cela est possible "sans perte de généralité" pour l'identité. Il présente de plus l'identité selon le mode d'utilisation adopté.

Le **choix d'une identité** parmi des variantes plus ou moins générales est un problème plus délicat. Pourquoi, par exemple, ne pas choisir l'identité (1) ou choisir (3bis) avec 0 comme borne inférieure ? Nous avons tranché en décidant que l'identité appliquée correspond à l'identité la plus spécifique disponible. Par exemple l'identité :

$$\frac{p}{q}\,r = \frac{pr}{q} \quad \text{.....(11)}$$

fait partie des identités courantes. Elle est utilisée plusieurs fois dans la démonstration de la formule de Gauss. Il serait impensable de la décrire à partir de l'identité suivante (même si elle est elle-même démontrée comme cela) :

$$\frac{p}{q}\frac{r}{s} = \frac{pr}{qs} \text{ (avec s=1 et simplification des produits) .....(12)}$$

6.3.3. Conclusion pour l'exploitation des formules

Pour fournir un environnement de manipulation naturel, il est indispensable de favoriser une gestion des connaissances par l'usager et le système indépendamment de l'intervention d'un expert informaticien. Le problème qui se pose alors est de fournir au mathématicien - ou à l'apprenant - le moyen d'introduire des connaissances, de les utiliser, et d'en superviser l'organisation.

Ce travail s'insère dans une approche d'analyse linguistique de textes et de productions mathématiques visant à déterminer des connaissances pratiques du mathématicien, afin de les utiliser pour une interaction ergonomique. La gestion des identités proposée fournit un **micromonde évolutif de formules**, indépendant d'un domaine d'expertise restreint. Elle contribue à une utilisation modulaire et déclarative des connaissances.

Nous nous sommes particulièrement concentrés sur l'exploitation des identités mathématiques car cela nécessite une expertise d'intérêt général, peu mise en évidence jusqu'alors. De plus, ce type de raisonnement, visant à l'exploitation d'identités connues, est crucial pour notre approche d'assistance. Il constitue un raisonnement sur des tâches élémentaires, susceptible de faciliter grandement le rôle d'un Assistant mathématicien intelligent.

Nous avons notamment proposé un modèle général d'utilisation d'identités, susceptible d'être équivalent aux connaissances expertes utilisées dans les systèmes de raisonnement. Par exemple, SEME utilise une heuristique pour tenter l'application de la formule du binôme (décrite en pseudo-langage dans [Baron 85]) :

> Pour évaluer une expression-sigma, si son terme général contient un seul symbole de combinaison C, si le premier argument de C ne contient pas la variable liée alors que le deuxième argument la contient, on peut essayer d'appliquer la formule du binôme ou de s'y ramener.

Cette heuristique nous apparaît comme une compilation, sous forme de règle autonome, du mécanisme d'analyse d'identité que nous avons proposé. Il est alors raisonnable d'envisager des capacités de preuve automatique comparables à celles des systèmes de raisonnements, décuplant ainsi l'activité du mathématicien et la vérification de ses productions.

Nos propositions appellent toutefois à être développées. L'expertise d'analyse des formules et la prise en compte des effets produits dans le raisonnement n'en sont qu'à leur balbutiements. De plus le guidage de la stratégie fins-moyens d'utilisation des identités peut être **enrichi par une expertise d'exploitation**. Nous proposons de rattacher, par dialogue, cette expertise aux identités concernées. Il est par exemple possible d'identifier des situations où telle identité sera (ou ne sera pas) intéressante. C'est ce que propose, en substance, Monique Baron lors de ses réflexions [Baron 85].

Par ailleurs, l'expertise d'analyse de formules est d'autant meilleure que le savoir-faire d'utilisation est important. Cela est clair avec la reconnaissance de formes mathématiques, lorsque la structure devient cachée. L'identité "$a^2 - b^2 = (a+b)\,(a-b)$" doit être recherchée dès la description d'un carré, lorsque l'effet factorisation est cherché. Cette expertise d'exploitation doit être rattachée à l'identité, afin de déclancher la recherche d'un autre carré, éventuellement masqué, comme dans "$8-2x^{2}$". De plus, les limites doivent être précisées, afin d'éviter "3-12=3(1-4)=3(1-2)(1+2)=-9" ou "$2x-8=2(\sqrt{x}-2)\,(\sqrt{x}+2)$" (avec $x\geq 0$).

Enfin, l'expertise d'analyse des formules n'est pas totalement indépendante des méthodes générales de raisonnement dans le domaine. L'intérêt de la relation :

$$\binom{n}{r} = \binom{n-1}{r} + \binom{n-1}{r-1}.$$

ne s'affirme que comme outil privilégié d'une méthode dite "de perturbation" dans [Graham 89], méthode très efficace pour résoudre les problèmes difficiles de sommation en produisant une relation récurrente.

En développant ce type de raisonnement d'exploitation des identités mathématiques, notre ambition n'est pas forcément une automatisation totale. Elle vise notamment la **mise en œuvre d'un dialogue avec le système**, afin de planifier la résolution, sachant que l'instanciation et l'exécution de ce plan pourront être essentiellement à la charge du système.

Notre problème principal, présenté section 6.1 et visible figure 2.8, peut alors faire l'objet d'un dialogue entre mathématicien et Ami. Ce dialogue permet d'édifier la planification de la résolution à partir des principes d'analyse précédents :

▶ calculer $S = \sum_{k=0}^{n} k \binom{m-k-1}{m-n-1}$

✍ Je ne trouve aucune identité applicable, ni aucune piste d'application. Dois-je considérer des formules proches ?

▶ si le terme k n'était pas là

✍ L'indice de la sommation apparaît dans l'argument supérieur du coefficient binomial mais pas dans l'argument inférieur (expertise d'analyse). Nous pouvons tenter d'appliquer l'identité (5.10) de sommation sur l'argument supérieur. Vous pouvez accepter ce plan, le développer, en choisir un autre ou changer d'hypothèse !

▶ accepté

✍ Pour l'appliquer, il faut absorber k dans le coefficient binomial. Je tente d'appliquer une identité pour cela...

✍ Je ne trouve aucune identité applicable, la seule identité possible est l'absorbtion (5.6), mais elle ne s'applique que pour (m-k) $\binom{m-k-1}{m-n-1}$. Voulez-vous accepter ce plan, en choisir un autre ou revenir en arrière ?

▶ utiliser k = m-(m-k)

✍ Je développe ce plan en écrivant k = m - (m-k). Pour appliquer l'identité (5.6), je sépare la somme produite en deux expressions, nous obtenons alors :

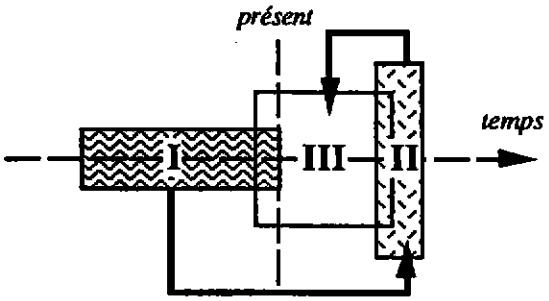
Ce dialogue, reproduisant fidèlement celui de la figure 2.8, pourrait avoir lieu dans la mesure où un raisonnement global intégrerait les stratégies d'utilisation d'identités. L'Ami serait alors en mesure de disposer du plan de résolution et de continuer à l'appliquer en écrivant les formules S=...

L'explication de la démarche de raisonnement permet au mathématicien de suivre son évolution et de guider les choix importants. Pour retomber sur l'identité (5.26) que nous avons aussi étudiée, le mathématicien aurait pu répondre, au lieu de l'hypothèse "si le terme k n'était pas là", :

▶ utiliser $k = \binom{k}{1}$

Cette voie de recherche sur l'activité du mathématicien est particulièrement prometteuse, car elle laisse imaginer l'**usage du Langage Mathématique comme moyen de "programmation"**. Bien entendu, il ne s'agit pas d'une simple utilisation de mots, mais d'un mécanisme plus profond permettant de réaliser les tâches élémentaires de l'activité "à la façon du mathématicien". Les langages de Phrases sont alors complémentaires des langages de Formules et des langages Graphiques représentant les tâches et les structures de données.

CONCLUSION



1. UNE ETUDE DE L'ACTIVITE MATHEMATIQUE

Il serait présomptueux de prétendre que les mathématiques sont peu explorées ! Par contre il est réaliste de soutenir que l'**activité mathématique est mal connue**. Nous en avons exploré certains aspects dans un but précis : l'assistance informatique au mathématicien. Notre contribution a consisté à poser explicitement cette étude en ces termes.

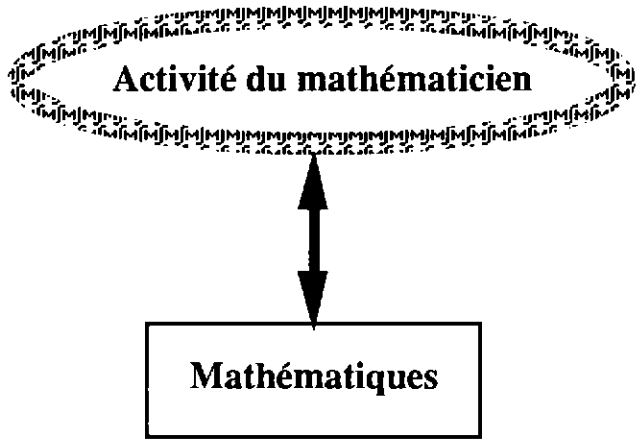


figure Ω

Nous avons entrepris cette étude à partir de la conviction intuitive que les mathématiques constituent un **domaine d'application excessivement prometteur** pour l'informatique, bien qu'encore peu exploitable. Notons que c'est principalement l'inverse qui se produit actuellement, puisque l'informatique est le vecteur de l'exploitation "appliquée" des théories mathématiques.

D'ailleurs, l'utilisation de l'informatique pour faire des mathématiques apparaît souvent **paradoxale** : Nous sommes partagés entre l'impression que l'informatique peut résoudre la majorité des problèmes, et le sentiment qu'elle n'améliore en rien l'activité mathématique. Ce paradoxe n'est qu'apparent. Les outils disponibles, quoique performants, sont conçus pour produire un résultat mathématique, et non reproduire et assister l'activité du mathématicien. La distinction entre mathématiques et activité mathématique lève ce paradoxe.

Par ailleurs, les systèmes experts en raisonnement abordent l'activité du mathématicien avec une volonté d'autonomie sur un domaine restreint. Les recherches développées pour cela ces trente dernières années ont montré leur efficacité mais aussi leurs limites. Une démarche d'assistance à l'activité du mathématicien peut les compléter afin d'aboutir à des systèmes plus étendus, susceptibles d'accueillir les mathématiciens au travail. Le problème principal est alors l'**utilisation du Langage Mathématique**, et l'élaboration des moyens de communication avec le système.

Nous prônons une prise de conscience des enjeux scientifiques d'une **linguistique des mathématiques**. Nous avons présenté la diversité des thèmes d'analyse et leur difficulté. Nous avons illustré notre thèse par des propositions originales visant à fournir des pistes interdiscipinaires nouvelles. Nos propositions se répartissent selon deux thèmes :

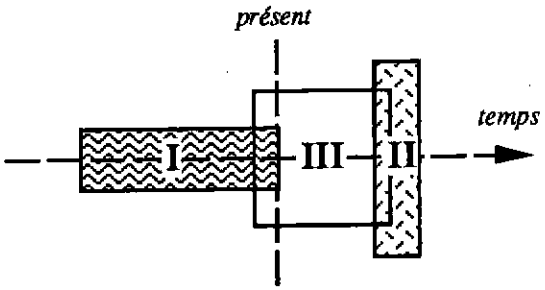
- l'**analyse pratique du langage**, dont la spécificité suscite le développement d'une linguistique des mathématiques ;
- un AMI, utilisant un **Assistant mathématicien intelligent**, pour une pratique différente des MAO (Mathématiques Assistées par Ordinateur).

Les difficultés principales de cette étude ont été de **définir les problèmes pertinents**, afin de bien les traiter. Elles ont débuté avec la notion même de MAO, qu'il est difficile de définir vu la pluralité des utilisations possibles, et que nous avons distinguées du traitement informatique de problèmes mathématiques. Aborder les MAO c'est déjà faire un choix :

- Evaluer des outils et techniques disponibles pour en proposer des améliorations ;
- Traiter des thèmes communs aux outils et applications des MAO.

Ce choix est à rapprocher de la distinction entre "amont" et "aval". En aval, il s'agit plutôt d'améliorer ou de regrouper des fonctionnalités existantes : ce sont principalement les techniques disponibles qui contraignent les recherches. En amont, il s'agit principalement de définir des sujets communs et de fournir une vue d'ensemble des problèmes à résoudre et de leur maturation actuelle. Ce sont alors les tâches du mathématicien et les potentialités du domaine qui sont prépondérantes[1].

Le choix "amont" de **fournir des outils d'assistance au cycle de vie des démonstrations** nous a amené à découper notre étude selon les trois parties suivantes :



I) Quelle est la nature de l'activité mathématique ?

II) Que peut-on souhaiter ?

III) Que peut-on proposer ?

figure 0.1

Notre choix d'une approche amont est autant conjoncturelle que structurelle. De nombreux sujets "aval" ont été en effet abordés par des équipes dont les outils, les compétences ou les moyens nous dépassaient. Vu l'ampleur du champ disponible en amont, nous avons été incités à innover plutôt qu'à améliorer l'existant. En contrepartie, notre apport est qualitatif. Nous avons néanmoins réalisé deux implémentations sur machine lisp explorer, qui nous ont servi de repère pour la modélisation ergonomique, afin de conciler innovation et réalisme.

2. PRESENTATION CHRONOLOGIQUE

Nous avons proposé dans cette étude un **module ergonomique de manipulation de formules** baptisé SOFTMATH, ainsi qu'un processus de développement ascendant d'un AMI (Atelier Mathématique Intégré). Nous présentons ces outils dans cet ordre chronologique car c'est une démarche ascendante (ici encore) qui est à leur origine. Cette perspective diachronique montre comment nous avons su éviter des voies qui allaient s'avérer non prometteuses, et dévoile la reconstruction que nous avons effectuée ultérieurement.

Les manipulations de formules ne sont effectuées sur ordinateur que si le mathématicien y trouve un intérêt par rapport aux facilités procurées par un support papier. Or, malgré les imperfections d'EDIMATH, il semblait évident que les saisies graphiques de formules allaient atteindre leur rendement optimal. D'un autre coté, l'autre voie ouverte pour susciter l'intérêt du mathématicien consistait à exploiter les systèmes de calcul formel. Les solutions existantes nécessitaient des équipes de recherche spécialisées car les solutions étaient déjà au stade pré-industriel.

Pour intéresser le mathématicien, nous avons donc exploré une troisième voie qui s'est traduite en deux étapes : SOFTMATH et un AMI. SOFTMATH part de la constatation évidente que, pour **outrepasser les limitations graphiques** des manipulations de formules, il faut en représenter le sens et bénéficier ainsi de capacités de calcul. Ce problème est similaire à celui des éditeurs de programmes "intelligents". Nous nous sommes ainsi inspirés des environnements de programmation, qui introduisaient des éditeurs structurés afin que l'écriture se fasse en interaction avec l'évaluation du sens. Nous avons donc cherché à décrire la syntaxe abstraite des formules à l'aide d'une représentation modulaire "à objets" des constituants des formules [Moulis 87].

Conjointement à cette prise en compte de la structure, l'assistance du mathématicien requiert de préserver les qualités ergonomiques des formules et de leurs manipulations. L'inconvénient des éditeurs structurés, inconvénient particulièrement sensible dans les versions existantes à cette époque, est que les structures sont presque explicitement construites par l'usager et que les manipulations sont contraignantes. Pour

[1] Cette approche amont est préconisée par exemple par Jacques Calmet pour la création de systèmes de calcul algébrique formel "intelligents" [Calmet 87].

autoriser les manipulations sur la représentation la plus adéquate, nous avons introduit les **trois structures couplées** (apparente, textuelle et mathématique). La structure apparente, souvent préférée comme support visuel, permet implicitement d'accéder aux structures textuelles ou mathématiques manipulées.

Ces deux objectifs d'**ergonomie de l'interaction** et de manipulation structurée permettent de caractériser SOFTMATH. L'essentiel de cette partie III a alors consisté à bâtir une organisation des connaissances, ainsi que des traitements nécessaires à ces manipulations de formules. Cela nous a conduit à étudier et étendre les fonctionnalités des environnements existants en fonction des besoins et de la modélisation cognitive propre aux mathématiques.

Mais il semble peu réaliste de réaliser un environnement de manipulation de formules sophistiqué qui serve uniquement de "super-interface" aux systèmes de calcul formel et de traitement de texte. L'évolution des systèmes de calcul allaient les amener à développer des interfaces plus limitées, dont le but principal serait d'exploiter les algorithmes de calcul formel disponibles. Pour exploiter les connaissances mathématiques et l'expressivité des manipulations de formules, nous avions besoin d'un AMI, qui soit un atelier de développement d'applications mathématiques pour le concepteur, et un atelier de **construction de preuves et de démonstrations** pour le mathématicien.

Nous avons ainsi étudié minutieusement les aspects cognitifs de l'activité mathématique. Nous avons étudié l'intérêt de la logique comme outil d'analyse - du langage et des mathématiques - puis exploré la pratique de l'activité mathématique et les moyens de respecter la créativité du mathématicien. Nous avons illustré à partir d'exemples qu'il restait à développer une **linguistique spécialisée en mathématiques**. Un AMI s'avère prometteur pour cela, car il offrirait un modèle informatique et permettrait une étude expérimentale de l'activité mathématique.

Cette étude introductive nous a aussi permis de délimiter notre approche des mathématiques aux aspects langagiers et aux tâches élémentaires. De nombreux travaux proposent en effet des modèles heuristiques ou des modèles de raisonnement permettant de se substituer aux facultés créatives du mathématicien. La meilleure limite de leur automatisation est qu'en trente ans, depuis les résultats spectaculaires de la DAT vers 1960 (notamment dans les théories logiques formelles), aucun domaine mathématique conséquent n'a été balisé en utilisant les déductions dites naturelles. Nous avons donc adopté une approche complémentaire qui est celle de l'**assistance au mathématicien**.

Ces deux points - l'étude de l'activité et du Langage Mathématique d'une part, et l'assistance à une activité de construction de preuves et démonstrations d'autre part - nous ont conduit à étudier les mécanismes de résolution de problème et de constitution d'une preuve à granularité variable. Avec le langage et les connaissances nous disposions alors des ingrédients permettant l'interaction nécessaire à l'assistance au mathématicien, à condition de couvrir le cycle de construction. Nous avons de plus présenté partie II des **modèles d'interaction** permettant de construire une preuve et sa démonstration associée, en réutilisant au mieux l'information déjà disponible dans un AMI.

Cette approche d'assistance apporte ainsi un regard différent sur la conception de systèmes destinés aux mathématiques. Le **raisonnement** reste toujours la composante principale des mathématiques, mais son utilisation requiert prioritairement la maîtrise du support d'expression : le langage. Même s'il prend en compte certaines situations typiques, le raisonnement à la charge du système est beaucoup plus laborieux que celui du mathématicien. Par ailleurs, les **connaissances** disponibles et leur mise en œuvre sont primordiales pour la réalisation des tâches d'assistance.

3. NOTRE CONTRIBUTION VERS UN AMI

Cette étude peut être envisagée comme un **premier pas vers l'édification de MAO**, au sens de l'assistance à l'activité mathématique. Les systèmes de manipulation de formules classiques n'ont pas les moyens de reproduire l'activité mathématique naturelle. Ils peuvent juste en capturer les aspects essentiels relativement à la tâche à laquelle ils sont destinés. Notamment, ils n'ont pas :

- les connaissances pour le faire ;
- une architecture informatique intégrée, apte à utiliser des connaissances ;
- les modèles nécessaires à une exploitation mathématique diversifiée.

Nous avons en effet apporté plusieurs ingrédients conceptuels indispensables comme la prise en compte du langage, les preuves à granularité variable et les trois structures couplées des formules : structures apparente, textuelle et mathématique. De plus, même si ces aspects étaient disponibles, les systèmes

classiques n'auraient pas les moyens informatiques permettant de les exploiter. En effet, les systèmes à venir nécessiteront un grand nombre de connaissances, incluant des connaissances pragmatiques. De plus, l'assistance se traduit par une réorganisation importante des ressources au profit de l'interaction, incluant notamment des capacités de déduction, de présentation, et une plus grande liberté d'expression.

Une telle étude effectue une avancée **pour les Mathématiques Assistées par Ordinateur**. Avant, nous ne savions pas précisément ce qu'il était intéressant d'approfondir. Maintenant, nous avons à disposition de nombreuses études et propositions "à la carte". De plus, notre travail permet d'évaluer l'intérêt, les difficultés ou les limites des projets concernant l'activité mathématique. C'était le prérequis indispensable à l'émergence de véritables Mathématiques Assistées par Ordinateur.

Plus généralement, cette étude peut s'envisager sous divers points de vues, qui considèrent à la fois l'étude cognitive de l'activité mathématique et l'utilisation d'un AMI :

- **pour le linguiste** : l'étude cognitive se révèle porteuse d'une linguistique des mathématiques. Un AMI est surtout un modèle informatique d'élicitation, d'organisation et d'acquisition des connaissances de traitement du Langage Mathématique.
- **pour le mathématicien** : l'étude cognitive l'intéresse plus particulièrement s'il est épistémologue ou enseignant. Par ailleurs, un AMI constitue la seule proposition existante de production de preuves avec leurs démonstrations associées. Les lois et les conventions qui régissent la production de telles preuves et de démonstrations ont été très peu étudiées. Nous l'avons fait en partie pour savoir quelles informations fournir au système.
- **pour l'informaticien** : l'étude cognitive et un AMI enrichit l'informatique, de la même façon que l'apport le plus important pour le mathématicien n'est pas de construire une preuve mais d'enrichir les mathématiques. Cette proposition de conception d'un assistant informatique intelligent est d'un intérêt direct ; il est réhaussé par l'apport technique dû à la constitution d'un AMI et à son exploitation.
- **pour la didactique des mathématiques** : une description plus fine de l'activité mathématique, des fonctions du langage et de ses manipulations permet d'expliciter le comportement des mathématiciens afin de mieux les comprendre, mieux leur enseigner, et corriger leurs travers. Il n'est d'ailleurs pas étonnant que les rares travaux existants sur l'activité mathématique soient issus de besoins didactiques.
- **pour les recherches en MAO** : cette étude présente des sujets de recherche bien définis et recadre leur intérêt dans une perspective globale. Par exemple, les mécanismes de généralisation permettant l'introduction de notations graphiques à l'aide de points de suspensions sont d'une utilité immédiate pour l'interface de tout système de calcul formel. Ces mécanismes de généralisation sont aussi indispensables à tout apprentissage en vue de qualifier abstraitement une formule (ce qui est d'un intérêt vital pour l'exploitation d'une base de théorèmes).
- **pour les connaissances mathématiques et paramathématiques** : un AMI constitue la seule proposition actuelle pour représenter des démonstrations et leur preuve associée. Cette étude contribue à une réflexion encore balbutiante sur l'acquisition et la représentation de connaissances mathématiques, et commence à organiser l'ensemble des connaissances à éliciter.
- **pour les systèmes à vocation mathématique**, la mise à disposition d'un environnement générique est un défi important. Il évite de refaire systématiquement un travail commun pour se concentrer directement sur les fonctionnalités spécifiques. Il permet de plus une synergie entre ces fonctionnalités par leur intégration dans un AMI.
- **pour l'ergonomie de l'interaction** :
  - le souci constant d'ergonomie : dans la formulation, à l'aide de concepts et d'expressions "naturels" ; dans la recherche, en utilisant des procédés humainement traduisibles en explication, et dans la présentation, en respectant les processus perceptifs du mathématicien.
  - la recherche d'un processus de simulation-communication à l'aide d'une manipulation directe des objets de la simulation (formules, preuves, concepts) et un assistant qui gère la communication. Cet assistant observe les manipulations, prend des initiatives lors d'une formulation ou une présentation difficile et est l'interlocuteur de l'usager quand survient une difficulté dans la formulation, la recherche ou la présentation du problème.
  - le paradigme de l'approche Intercom qui cherche à respecter la créativité de l'usager dans une activité "naturelle". Il consiste schématiquement en un développement expérimental ascendant à partir du langage et des tâches élémentaires.
  - la notion d'assistance à la conception, ainsi que la nécessité d'intégrer le cycle de conception de l'usager pour tenter de circonscrire des tâches contrôlables en limitant les paramètres demandés à l'usager.

Pour cela, ces recherches utilisent d'autres techniques informatiques et interagissent avec elles :

- **traitement du Langage Naturel** : les mathématiques constituent un domaine opératif non restreint où le langage a un rôle de tout premier plan grâce à sa flexibilité et à la précision des concepts mathématiques. Nous avons proposé une représentation des connaissances originale associant langage et domaine conceptuel, ainsi que des idées intéressantes pour l'analyse et la génération.
- **systèmes faisant des mathématiques** : souvent, cette étude recadre leurs perspective d'utilisation à long terme et peut orienter certains efforts de développement. L'organisation informatique d'un AMI et le traitement du dialogue et des connaissances, a pour ambition de servir de base à l'écriture d'applications spécifiques.
- **domaines de conception** : les travaux que nous avons effectués concernent directement l'Assistance au Travail Scientifique. L'approche Intercom et les principes d'interaction sont applicables dans des domaines comme la CAO-BTP ou la conception de scènes à partir de représentations conceptuelles.
- **Sciences Cognitives** : le principal intérêt est la mise à disposition d'un outil d'expérimentation pour l'analyse de tâches, mais surtout pour étudier et tester des modèles cognitifs. Un thème parmi d'autres serait la "visualétique", qui consiste à décider et comprendre comment présenter une information visuelle à partir d'une information abstraite (par exemple visualiser un ensemble de données ou produire un graphe interactif à partir d'une preuve).
- **sciences de l'ingénieur** : il serait trompeur de considérer l'assistance comme un sous-produit de l'automatisation. Il y a par exemple automatisation de certaines tâches élémentaires dans un mécanisme d'assistance. La distinction entre assistance et automatisation concerne donc principalement l'approche du domaine. L'automatisation requiert un domaine souvent très limité pour lequel il existe une procédure de décision. L'assistance se subdivise selon ses objectifs :
  - transfert sur ordinateur des **fonctionnalités minimales** : par exemple, traitement de texte ou construction interactive de preuve en logique;
  - transfert sur ordinateur d'une **activité "naturelle"** aussi complète que possible, nécessitant un atelier et des moyens sophistiqués, comme cela émerge par exemple pour la génération de documents techniques.

4. DEFIS ET PERSPECTIVES DES MAO

Les défis informatiques des mathématiques à venir sont :

① l'usage des mathématiques pour constituer des modèles informatiques ;

② la création de bases de connaissances mathématiques ;

③ la compréhension de documents mathématiques ;

④ la production de raisonnements et de démonstrations.

① : Les mathématiques sont utilisées de deux façons pour contribuer à **l'élaboration de modèles informatiques** :

- mathématiser certains aspects informatiques et résoudre le problème mathématique ainsi produit. C'est le cas par exemple pour la preuve de propriétés d'un programme. En présence d'outils appropriés, cela débouche sur le point suivant.
- mécaniser certains aspects mathématiques par l'élaboration d'un modèle informatique. C'est ce point que nous détaillons maintenant.

Un résultat mathématique peut être utilisé à des fins internes ou externes aux mathématiques :

- Un résultat qui sert à des *fins internes* est utilisé comme une donnée servant à produire d'autres résultats. L'essentiel du travail du mathématicien consiste ainsi à exploiter de tels résultats.
- Un résultat est exploité à des *fins externes* lorsqu'il sert dans un modèle mathématique conçu pour être utilisé par une autre science. Il doit alors être dépouillé de ses liens avec d'autres résultats afin que le dit modèle soit indépendant.

Il y a ainsi des *résultats primaires*, tels l'inverse d'une matrice donnée utilisée pour modéliser un phénomène physique, et des *résultats secondaires*, car indirectement exprimables par le biais d'une méthode (ici de diagonalisation de matrice). De tels résultats secondaires sont par essence internes, car leur expression utilise une bonne part des possibilités du Langage Mathématique.

L'informatique joue ici un rôle essentiel puisqu'elle offre des langages génériques permettant d'exprimer des méthodes scientifiques suffisamment précisées. Des résultats mathématiques peuvent alors être transcrits en résultats informatiques, c'est-à-dire en procédures génériques de production de résultats primaires. Un aspect très intéressant de ce procédé est que les résultats primaires obtenus peuvent aussi être utilisés de façon "interne", au service des mathématiques.

Ce défi ① nécessite donc des modèles mathématiques (conceptuels) et informatiques (calculables), avec un domaine d'application (par ex. physique, mathématique ou informatique). Les améliorations consistent à faciliter le travail sur les modèles en liaison directe avec le domaine. L'intégration que nous proposons entre mathématiques et informatique contribue à cette amélioration.

② : Le projet de **base de connaissances mathématiques** est le plus urgent au vu de la production annuelle de publications et de théorèmes[2]. Il semble que dans un premier temps, il faille informatiser le texte des publications et en produire manuellement le résultat principal pour sa réutilisation. Un système d'archivage, de diffusion et de gestion de ces résultats s'impose. Cela pourrait aboutir, dans un premier temps, à des traités de mathématiques informatisés disposant d'outils de recherche et de traitement des informations.

Plutôt qu'une simple indexation de théorèmes, il est souhaitable de chercher aussi à organiser les concepts, leurs propriétés et leurs relations. Leur structure servirait de support naturel aux informations à rattacher. La réutilisation de théorèmes se ferait alors facilement par instanciation de leur énoncé et garantie de leur bonne application. Cette connaissance sémantique permettrait enfin une analyse plus fine et une utilisation plus riche des informations.

Notre travail est utile dans cette optique. Une base de connaissances mathématiques constitue la condition sine qua non à la réalisation des défis ③ et ④. Elle s'apparente à l'existence d'une encyclopédie ou d'un catalogue conceptuel (au sens de [Sowa 84]).

③ : La **compréhension de documents mathématiques** est un enjeu complémentaire de même envergure. La production d'un graphe de preuve annoté par du texte peut être envisagée. Ce texte doit néanmoins être structuré selon les schémas mathématiques usuels. Une condition minimale d'intérêt semble être la création d'un langage international pour les mathématiques, tel qu'il est proposé dans [Strichartz 89]. Un tel langage permet l'indépendance vis à vis des mots des langages naturels respectifs en respectant les schémas mathématiques, dans l'esprit de la traduction automatique.

Dans son propos, Robert S. Strichartz note déjà l'importance de l'étude de l'usage du langage. Il relève aussi l'obstacle rédhibitoire d'une structure de phrase trop rigide ou peu coopérative pour lever les ambiguités. Il écarte de ce fait une sorte de "programmation" dans un langage de preuve ou de démonstration prédéfini.

La compréhension requiert une structure fine du contenu mathématique des documents. Quand cette structure n'est pas disponible lors du processus de production, il faut disposer d'un moyen de la reconstituer par analyse. Les travaux en Langage Naturel montrent que les connaissances pragmatiques jouent un rôle majeur dans cette compréhension. Dans tous les cas, l'étude de l'usage du Langage Mathématique et de la génération de démonstrations à partir de preuves est indispensable.

④ : Le défi de **production de raisonnements et de démonstrations mathématiques** nécessite obligatoirement l'existence d'un réseau de concepts. Il bénéficie des succès de la Démonstration Automatique de Théorèmes dans les domaines où des procédés de calcul mécanisés ont été découverts. Mais dans la majorité des cas, une approche de type Système-Expert à base d'heuristiques reste seule envisageable. La structure de la base de connaissances correspond à une analyse très fine du domaine, et particulièrement de l'usage pratique du mathématicien.

L'étude des représentations et du langage utilisés par le mathématicien apporte ici une contribution essentielle pour la production de raisonnements et de démonstrations. Contrairement à un langage instauré par décret, le Langage Mathématique apparait indétronable. Il constitue une clé permettant d'identifier et

[2] L'enjeu scientifique de diffusion d'information et d'utilisation des moyens informatiques est, par exemple, affiché dans la présentation du projet EUROMATH.

d'analyser les facteurs de l'activité mathématique en vue d'informatiser sa production. Les connaissances mathématiques peuvent alors être progressivement explicitées par la pratique des mathématiques et son étude expérimentale.

Ceci sous-tend la création d'un milieu d'accueil, un Atelier Mathématique Intégré AMI. Les balbutiements actuels sur ces défis discréditent les tenants des mathématiques "faciles" car formelles et bien maitrisées. Une analogie avec la programmation montre l'intérêt de ces défis, ainsi que les solutions possibles, leurs difficultés et leurs limites.

MAO : Le principal **obstacle au développement des MAO** est paradoxalement le caractère général et international des mathématiques ! Les mathématiques sont conceptuelles et utiles à toute activité scientifique ; par ailleurs, elles ne sont pas constituées en industrie. Il n'y a pas actuellement de volonté de concertation, et des moyens suffisants, pour envisager des projets à caractère prospectif en mathématiques. Il semble donc qu'il faille relever graduellement ces défis, en profitant des besoins technologiques, tout en focalisant l'attention de la communauté scientifique sur ces défis.

Un épistémologue relèverait un autre obstacle plus insidieux : sans remonter jusqu'à Pascal ou Leibnitz, les solutions pour mécaniser en partie l'activité mathématique ont suivi l'évolution scientifique et technique. Pour un usage informatique, nous notons par exemple :

- la Démonstration Automatique de Théorèmes des années 60 ;
- les propositions de formalisation de démonstrations "formelles" [De Bruijn 68] ou "naturelles" [Arnold 68] ;
- les environnements de résolution de problèmes mathématiques tels [Vance 73], basés sur un langage de programmation algébrique dans le style d'APL ;
- les systèmes de calcul algébrique formel et leurs développements récents ;
- les Systèmes-Experts en raisonnement ;
- les approches cognitives de l'activité du mathématicien.

Ces solutions avancées n'ont pas directement abouti jusqu'à présent, sans qu'il soit évident de déterminer l'influence respective de la méthode proposée et des moyens mis en œuvre. Aussi, nous nous bornerons à esquisser quelques retombées possibles de l'évolution technologique actuelle. Nous pensons qu'un domaine sujet à un développement informatique prometteur est la **théorie des nombres** :

- ce domaine fait partie de la couche mathématique générale souvent indispensable à un usage mathématique non restreint ;
- les besoins issus de l'informatique sont nombreux : ce n'est pas par hasard que les outils de preuve actuels sont développés pour les besoins informatiques. De plus l'intérêt des informaticiens est indispensable pour le développement de systèmes informatiques complexes ;
- les problèmes se prêtent bien à l'utilisation d'algorithmes de calcul symbolique, ce qui rend l'ordinateur indispensable comme outil de calcul ;
- les preuves de ces problèmes sont minutieuses et indispensables, ce qui les rend longues et fastidieuses pour le mathématicien. Ce type d'argumentation se prête particulièrement bien à une assistance.

Pour ce qui concerne l'activité mathématique en général, une retombée pratique à court terme de l'évolution technologique est une **amélioration du traitement des formules, preuves et démonstrations**. La saisie de formules ou de textes mathématiques devrait être facilitée par la reconnaissance vocale dès que les technologies seront opérationnelles. La mise en page de démonstrations devrait pouvoir être grandement assistée et bénéficier des facilités de calcul pour les aspects mathématiques. La présentation graphique sous forme de preuve, des arguments de cette démonstration devrait être quasi-automatique, tandis qu'un éventail de définitions et théorèmes devrait être disponible. Enfin des utilitaires graphiques dédiés, par exemple pour la géométrie ou l'Automatique, devraient permettre la correspondance de données entre représentations, avec saisies incrémentales sur la représentation la plus appropriée au développement actuel du problème.

Une retombée scientifique probable à moyen terme est l'**organisation informatique de données mathématiques**. Elle regrouperait les connaissances de domaines mathématiques (définitions et propriétés), ainsi que les principaux théorèmes et leurs démonstrations. Quoique les démonstrations seraient essentiellement textuelles, afin de décrire le texte de la publication, les principaux arguments utilisés devraient être représentés mathématiquement afin de permettre des vérifications de dépendance et de cohérence. Une banque mondiale à l'usage des mathématiciens devrait à plus long terme regrouper les publications mathématiques et faciliter la découverte et le travail de synthèse. Cette banque remplacerait notamment les anciennes tables de fonctions et fournirait les développements en série appropriés aux approximations cherchées (des programmes génériques spécialisés devraient permettre de les utiliser directement).

Une retombée industrielle envisageable à moyen terme, est l'utilisation des techniques dans le cadre d'un **atelier de Génie Logiciel spécialisé** dans la création de programmes pour les applications scientifiques et techniques. Il combinerait édition de documents, manipulations de formules et calcul symbolique, bibliothèque de programmes, générateur de code à partir de formules et Système-Expert dans l'utilisation de cette bibliothèque. Il créerait donc des programmes scientifiques spécialisés à la carte à partir de spécifications mathématiques, grâce à une panoplie de méthodes de résolution génériques.

Plus généralement, l'assistance au mathématicien suggère un **programme interdisciplinaire** impliquant la compréhension d'une activité humaine et sa modélisation par la machine. Nous proposons une approche Intercom et une collaboration des investigations des diverses disciplines autour d'un projet d'AMI. Ce programme est susceptible de retombées scientifiques et techniques dès son lancement : nous en avons d'ailleurs présenté précédemment quelques apports interdisciplinaires.

Une retombée scientifique à long terme de ce programme serait une phase d'**exploitation expérimentale des données de l'activité mathématique** recueillies jusqu'alors avec un AMI. Nous postulons que plus les systèmes informatiques deviendront "intelligents", et plus les outils d'analyse seront sophistiqués et capables de performances conceptuelles. Mais pour cela, il faut d'abord des données, et c'est la qualité des données qui sera le premier facteur limitant. Nous avons proposé pour cela de transférer l'activité mathématique sur ordinateur en modélisant même ses composantes les plus fines. L'analyse des données ainsi recueillies serait d'un intérêt capital non seulement pour les mathématiques, mais aussi plus généralement pour l'étude des activités de conception.

5. BILAN

Si la diversité des MAO traduit leur richesse, leur disparité résulte d'une absence d'**approche globale** unifiante. A l'heure où tout est informatisé, il n'existe aucun recueil informatique de connaissances mathématiques. De plus, si les traités de mathématiques exposent leur aspect structurel, l'aspect opératoire est méconnu : il reste, en grande partie, l'apanage de l'expérience des mathématiciens.

L'informatique s'intéresse néanmoins à cet aspect opératoire, pour appliquer ses capacités de déduction aux Mathématiques. Le livre [Bundy 83], présente un panorama des travaux, et le domaine formel dans lequel ils raisonnent. Ce domaine formel est souvent bien éloigné de celui du mathématicien, et de ce fait, hermétique aux techniques mathématiques.

Notre approche dite "naturelle" cherche a priori à fournir au mathématicien la forme usuelle du langage, des concepts et des formules. Elle assisterait ainsi la production textuelle de démonstrations et la construction de preuves mathématiques, y compris dans des domaines fortement conceptuels.

Nos propositions de **Mathématiques Assistées par Ordinateur** restent très modestes quant à la nature des tâches élémentaires à assister. De plus, les techniques à utiliser sont suffisamment bien maitrisées, malgré un besoin de perfectionnement et d'intégration.

Un article récent signé d'un groupe de huit personnes [Abelson 89] permet de comparer ces besoins avec ceux exprimés dans les activités scientifiques de calcul. Leur analyse, le dialogue imaginé et leurs propositions d'un assistant à l'ingénieur sont notablement plus ambitieuses. Leurs ambitions vont en effet largement au delà de tâches élémentaires, et elles visent une entière automatisation. Ils argumentent pourtant leur proposition à partir de techniques intelligentes existantes quoique éparses.

De plus nous avons exposé des problèmes généraux aux systèmes à vocation mathématique. L'approche que nous avons présentée est la seule qui permette d'envisager une **assistance cognitive au travail sur un domaine mathématique réaliste**. Les projets tels CYC [Lenat 90] montrent la différence qualitative et quantitative entre un système utilisable dans un micro-domaine et un système plus général. Il faut notamment repenser les connaissances nécessaires et leur acquisition. C'est ce que nous avons commencé en mathématiques.

Les défis informatiques que nous avons soulevés représentent des impératifs à long terme favorisant l'expansion des mathématiques. L'étude et la compréhension du Langage Mathématique usuel y jouent un rôle déterminant. Pour des raisons technologiques, psychologiques ou sociales, peu de travaux ont toutefois été réalisés dans ce sens. Trois erreurs ont favorisé cette lacune :

1. le raisonnement du mathématicien est considéré comme de la "logique à l'état pur", d'où l'équivalence : raisonnement mathématique = logique ;
2. la nature, la diversité et la quantité des connaissances du mathématicien ont été largement sous-estimées ;
3. les mécanismes d'utilisation des connaissances et le rôle des activités paramathématiques restent incompris.

En retour, les autres sciences et techniques sont susceptibles de bénéficier de l'étude du langage et de l'activité mathématique. Un atelier mathématique se trouve en effet au centre de *problématiques pluridisciplinaires* : Sciences Cognitives pour celles liées à l'acquisition, la représentation et l'utilisation des connaissances, Langage Naturel, environnements de manipulation, Génie Logiciel, systèmes de calcul ou de démonstration, etc. Enfin, les mathématiques bénéficient d'un travail d'organisation conceptuelle séculaire mais empirique. Une **étude expérimentale des connaissances** implicites d'un domaine aussi achevé, ne peut qu'enrichir les Sciences Cognitives.

Plus particulièrement, de nombreuses études restent à faire dans le domaine de la linguistique des mathématiques. Le meilleur moyen de soulever et résoudre ces problèmes est la modélisation et la réalisation informatique de travaux utilisant ***le Langage Mathématique comme support d'une activité mathématique naturelle***. Cet abord encore inexploité présente des caractéristiques uniques :

- parce que la nature formelle des mathématiques garantit que la modélisation des concepts de base est adéquate ;
- parce que le travail de modélisation déployé pour rendre compte de l'activité mathématique est transposable à d'autres domaines conceptuels, notamment les domaines scientifiques ;
- par l'universalité du champ d'application des mathématiques ;
- par l'intérêt de l'étude de l'activité mathématique pour les Sciences Cognitives ;
- par l'essor des techniques "de pointe" utilisées pour cette réalisation.

De nombreux aspects des mathématiques ou de l'activité mathématique ne peuvent être étudiés actuellement faute de moyens de modélisation, de données observables, de traitement de ces données, et surtout de chercheurs pour faire avancer cela. Après analyse du langage et de l'activité, nous proposons une approche Intercom de conception d'un système d'assistance, avec un développement expérimental tant dans la phase initiale, que pour l'exploitation d'un AMI.

*Il apparait ainsi qu'une **communication "naturelle" avec un système** est un objectif très difficile. Elle fait appel aux techniques informatiques les plus sophistiquées pour simplement parvenir à des résultats honorables. Cet objectif constitue dès lors un axe de développement prometteur dont nous avons illustré l'interdisciplinarité. Cette problématique est celle de l'Assistance au Travail Scientifique.*

Il semble que les nouvelles questions suscitées par notre étude ne puissent qu'encourager les recherches dans ce domaine très ouvert des MAO. Avec les moyens et le développement informatique actuels, l'enjeu technologique englobe désormais les Sciences Cognitives. Le manque d'études cognitives limite considérablement la faisabilité des systèmes informatiques - en mathématiques comme ailleurs. Nous avons illustré comment de telles études sont appellées à jouer un rôle de plus en plus déterminant dans la conception des systèmes informatiques.

RÉFÉRENCES BIBLIOGRAPHIQUES

OUVRAGES MATHEMATIQUES CITES

[Bossut M. DCC. LXXXII.] BOSSUT : *Cours de mathématiques* ; chez Jaubert, librairie du roi pour le génie et l'artillerie, M. DCC. LXXXII.

[Duris 84] P. DURIS, Z. GALIL : *A Time-Space Tradeoff for Language Recognition* ; Math. Systems Theory 17, pp. 3-12, 1984.

[Godement 66] R. GODEMENT : *Cours d'algèbre* ; deuxième ed., Hermann, Paris 1966.

[Graham 89] R.L. GRAHAM, D.E. KNUTH, O. PATASHNIK : *Concrete mathematics* ; Addison-Wesley, 1989 (avec une attention particulière au ch.2 : SUMS, pp. 21-41).

[Pac 86] J.L. PAC, M. SAMUELIDES : *Probabilités II* ; Cours ENSAE, 1986.

[Quinet 62] J. QUINET : *Cours élémentaire de mathématiques supérieures : tome 3 Calcul intégral et premières applications* ; troisième édition, Dunod, Paris, 1962.

[Slotine 86] J.-J. E. SLOTINE, J.A. COETSEE : *Adaptative sliding controller synthesis for non-linear systems* ; Int. J. Control, vol. 42, n°6, pp. 1631-1651, 1986.

[Whitworth 1875] W.A. WHITWORTH, C. TAYLOR, R. PENDLEBURY, J.W.L. GLAISHER : *The MESSENGER OF MATHEMATICS* ; MacMillan and Co., London end Cambridge, 1875.

LOGIQUE ET OUTILS

[Barwise 78] J. BARWISE ed. : *Handbook of Mathematical Logic*, North-Holland, 1978.

[Carnap 58] R. CARNAP : *Introduction to Symbolic Logic and its Applications*, Dover, New-York, 1958.

[Chou 88] S.C. CHOU : *Mechanical geometry theorem proving* ; Reidel Publ. Company, Dordrecht, 1988.

[Constable 86] R.L. CONSTABLE & al. : *Implementing Mathematics with the Nuprl Proof Development System* ; Prentice-Hall, 1986.

[De Bruijn 68] N.G. DE BRUIJN : *The mathematical language AUTOMATH, its usage and some of its extensions* ; Symp. on Automatic Demonstration, Versailles, Lecture Notes in Mathematics, vol. 125, Springer-Verlag, 1968, pp. 29-61.

[Fallot 89] L. FALLOT : *Une aide interactive à la construction de preuves en logique du premier ordre* ; Thèse Université de Bordeaux I, février 1989.

[Formel 89] projet Formel : *The Calculus of Constructions : documentation and user's guide* ; Rapport technique INRIA N°100, Août 89.

[Kleene 71] S.C. KLEENE : *Introduction to metamathematics* ; North-Holland, 1971.

[Pabion 76] J.F. PABION : *"Logique Mathématique"* ; Hermann, Paris, 1976.

[Parigot 88] M. PARIGOT : *Mathématiques et preuves de programmes* ; Le courrier du CNRS, suppl. au n°69 : images des mathématiques, 1988, pp. 40-47.

[Shapiro 85] S. SHAPIRO : *Second-order languages and mathematical practice* ; J. of Symb. Logic, vol 50, N°3, sept 85.

[Shoenfield 67] J.R. SCHOENFIELD : *Mathematical Logic*, Addison-Wesley, 1967.

[Thayse 89] A. THAYSE & co-auteurs : *Approche logique de l'intelligence artificielle* ; tome 2 : De la logique modale à la logique des bases de données, Dunod, Bordas, Paris, 1989.

[Veronis 89] J. VERONIS : *Un modèle logique de l'erreur dans le dialogue homme-machine en langage naturel* ; RIA, vol 3, n° 1, 1989, pp. 31-78.

[Wang 70] H. WANG : *Logic, computers, and sets* ; Chelsea publishing company, New York, 1970.

OUVRAGES DE REFLEXION ET ACTIVITES CREATIVES

[Cipra 83] B. CIPRA : *ERREURES ...et comment les trouver avant le prof...* ; Interédition, Paris, 1983.

[Cuppens 87] R. CUPPENS : *Comparaison de plusieurs démonstrations d'un résultat simple d'arithmétique* ; Bull. IREM-APMEP, IREM de Toulouse, juin 1987, pp. 18-23.

[Davis 85] P.J. DAVIS, R. HERSH. : *L'Univers Mathématique (The mathematical experience)* ; Gauthier-Villars, Paris, 1985. Des mêmes auteurs : *L'Empire Mathématique (Descartes' Dream. The World according to Mathematics)* ; 1988.

[Djedi 89] N. DJEDI, R. CAUBET, N. MATALLAH, M.Z. SANDOUK : *Un système à base de connaissances pour la modélisation géométrique en synthèse d'images* ; VISUDA, Juin 1989.

[Gilhooly 88] K.J. GILHOOLY : *Thinking : Directed, undirected and creative* ; Academic Press, 1988.

[Glaeser 83] G. GLAESER : *Genèse et maturation précoce d'une découverte mathémathique* ; Gazette des Mathématiciens, SMF, n°21, Mars 1983.

[Grabiner 86] J. V. GRABINER : *Is Mathematical Truth Time-Dependent* ; dans [Tymoczko 86], pp. 201-213.

[Hadamard 75] J. HADAMARD : *Essai sur la psychologie de l'invention dans le domaine mathématique* ; Coll. «discours de la méthode», Gaulthier-Villars, 1975 (original 1945).

[Hanrot 88] S. HANROT, P. QUINTRAND, E. CHOURAQUI, P. DUGERGIL, P. FRANCOIS : *Un système de CAO intelligent en architecture* ; EuropIA 88, Hermes, Paris, 1988.

[Kline 89] M. KLINE : *Mathématiques: la fin de la certitude* ; Christian Bourgois ed., 1989.

[Lakatos 84] I. LAKATOS : *Preuves et Réfutations* ; Hermann, Paris, 1984. (Titre original : *Proofs and Refutations* ; Cambridge University Press, 1976.)

[La Recherche 90] publi-information de la DRET, La Recherche 227, vol. 21, p. 1535, déc. 90.

[Ledizés 88] J.M. LEDIZES : *L'apport des systèmes experts pour l'aide à la conception* ; dans : *Intelligence artificielle et CAO en BTP* (journées CIMA), Hermes, Paris, 1988.

[Loi 82] M. LOI : *rigueur et ambiguité* ; Dans : *Penser les mathématiques*, Seuil, 1982, pp. 108-132.

[Pleiade 67] : *Logique et connaissance scientifique* ; Encyclopédie de la Pléiade, sous la direction de Jean Piaget, Ed. Gallimard, 1967.

[Pólya 86] G. POLYA : *Generalization, Specialization, Analogy* ; dans [Tymoczko 86], pp. 95-124.

[Quintrand 85] P. QUINTRAND & al. : *La Conception Assistée par Ordinateur en Architecture* ; Hermes, 1985.

[Tymoczko 86] T. TYMOCZKO : *New directions of the philosophy of mathematics* ; Birkaüser, Boston, 1986.

[Wallas 26] G. WALLAS : *The art of thought ;* London, Jonathan Cape, 1926 (ref. tirée de [Gilhooly 88]).

LANGAGES ET MATHEMATIQUES

[Anis 90] J. ANIS : *Une approche linguistique de la programmation ?* ; Bull. Ens. Public & Info, n°57, mars 90.

[Beynon 90] M. BEYNON, S. RUSS : *Variables in mathematics and computer science* ; Univ. of Warwick, Research report RR141, 1990.

[Cartier 89] P. CARTIER : *La vérification des démonstrations mathématiques;* Pour la Science n°146, p. 5, déc. 1989.

[Dhombres 81] J. DHOMBRES : *L'écriture mathématique;* Séminaire de philosophie et mathématiques ENS Ulm, IREM Paris Nord, coll. Philosophie-mathématiques n°29, juin 1981.

[Gerbier 87] Y. GERBIER, H. ICART SEGUY : *Les marqueurs logico-discursifs car, comme, parce que, puisque* ; Thèse U. Toulouse Mirail, juin 1987.

[Grice 75] H.P. GRICE : *Logic and conversation* ; dans P. Cole et J. Morgan : *Syntax & Semantics*, Academic Press, pp 41-58, 1975.

[Iverson 79] K. IVERSON : *Notation as a tool of thought* ; ACM Turing Award Lectures : The first Twenty Years (Programming Languages and systems : 1979), Addison-Wesley, 1987, pp. 339-389.

[Kayser 88] D. KAYSER : *Le raisonnement à profondeur variable* ; Actes des journées du PRC-IA, Toulouse, 1988.

[Laborde 82] C. LABORDE : *langue naturelle et écriture symbolique - deux codes en interaction dans l'enseignement mathématique* ; Thèse d'état INPG, didactique des mathématiques, 10 déc. 1982.

[Lacombe 84] D. LACOMBE.: *Les composantes du "raisonnement" mathématique;* Actes du Colloque de l'ARC, Orsay, 1984. (cf. également : Bulletin de liaison n° 2, IREM, Toulouse, pp. 87-98, 1987.)

[Lacombe 88] D. LACOMBE.: *La démonstration formelle : qu'est-ce que c'est ? A quoi çà sert ?* ; Actes de la 2° univ. d'été IA et enseignement des mathématiques, IREM, Toulouse, pp. 5-26, 1988.

[Larkin 87] J.H. LARKIN et H.A. SIMON: *Why a Diagram is (Sometimes) Worth Ten Thousand Words* ; Cognitive Science, 11, pp. 65-99, 1987.

[Lo Jacomo 75] F. LO JACOMO : *Le Langage Mathématique* ; mémoire de sémiologie Paris V, 1975.

[Martinet 85] A. MARTINET : *Syntaxe générale* ; Armand Colin, Collection U, 1985.

[Moulis 87] G. MOULIS : *Vers une representation du langage mathematique* ; Université d'été : Intelligence artificielle et enseignement des mathématiques, IREM, Toulouse, Juillet 1987, pp. 139-154.

[Solow 82] D. SOLOW : *How to read and do proofs : an introduction to mathematical thought process* ; Wiley & sons, New York, 1982.

[Strichartz 89] R.S. STRICHARTZ : *An International Language for Mathematics* ; The mathematical intelligencer (Opinion column) , vol 11, N° 1, 1989.

MODELISATION COGNITIVE ET DIDACTIQUE

[Balacheff 87] N. BALACHEFF : *Processus de Preuve et Situations de Validation* ; Educational Studies in Mathematics, vol. 18, 1987, pp. 147-176.

[Brown 78] J.S. BROWN, R.R. BURTON : *Diagnostic models for procedural bugs in basic mathematical skills* ; Cognitive Science, vol 2, pp. 155-191.

[Bruillard 91] E. BRUILLARD : *Enseignement intelligemment assisté par ordinateur et mathématiques : Une vision hypertexte des environnements d'apprentissage* ; Thèse Univ. du Mans, 14 fév. 1991.

[Caverni 88] J.P. CAVERNI, C. BASTIEN, P. MENDELSOHN, G. TIBERGHIEN : *Psychologie cognitive : modèles et méthodes* ; Presses Univ. de Grenoble, 1988.

[Chaudron 89] L. CHAUDRON : *Analyse de l'activité de démonstration mathématique : contribution à l'étude du raisonnement approché* ; Thèse ENSAE, Décembre 1989.

[Cuppens 88a] R. CUPPENS : *Bugs dans les compétances procédurales: l'histoire des "étapes de réparation des bugs"* ; Bull. de liaison Inter-IREM n° 2, Toulouse, juin 1988, pp 33-66.

[Cuppens 88b] R. CUPPENS : *La résolution de problèmes mathématiques par un homme ou une machine* ; Deuxième université d'été : Intelligence artificielle et enseignement des mathématiques, IREM, Toulouse, juillet 1988, pp 85-110.

[Darcel 87] N. DARCEL, M.C. ESCARABAJAL : *Résolution de problèmes et formalisme objet* ; MARI-COGNITIVA 87, mai 1987, pp 85-92.

[Denis 89] M. DENIS: *Approches cognitives de l'image mentale* ; Intellectica, vol 8, n°2, pp 85-107, 1989.

[Dumont 89] B. DUMONT : *Questionnements et interprétations des erreurs en mathématiques* ; Thèse d'état, Université Paris 7, janvier 1989.

[Dybjer 82] P. DYBJER : *Mathematical Proofs in Natural Language* ; First Meeting of the Swedish Artificial Intelligence Society, Uppsala, Report 5, Programming Methodology Group, april 1982.

[IREM 90] *Actes de l'université d'été informatique et enseignement de la géométrie* ; IREM de Toulouse, 1990.

[Janvier 87] C. JANVIER (Ed.) : *Problems of representation in the teaching and learning of mathematics* ; Lawrence Erlbaum Assoc., Hillsdale, 1987.

[Lindsay 80] P. LINDSAY, D. NORMAN : *Traitement de l'information et comportement humain : une introduction à la psychologie;* Éditions Études Vivantes, Montréal, 1980.

[Newel 72] A. NEWELL : *Human Problem Solving* ; Prentice-Hall, 1972.

[Nicaud 88] J.F. NICAUD, M. VIVET : *Les tuteurs intelligents : réalisations et tendances de recherches* ; TSI vol 7, n° 1, 1988, pp. 21-47.

[Norman 87] D.A. NORMAN : *Cognitive Engineering-Cognitive Science*, Interfacing Thought, MIT Press, 1987, pp. 325-336.

[Pastre 78] D. PASTRE : *Observation du mathématicien : Aide à l'enseignement et à la démonstration automatique de théorèmes* ; Educational Studies in Mathematics, vol. 9, 1978, pp. 461-502.

[Pólya 57] G. POLYA : *How to solve it* ; (2° ed.), Doubleday, New York, 1957.

[Rissland 78] E.L. RISSLAND : *Understanding Understanding Mathematics;* Cognitive Science, Vol 2, 1978, pp 361-383.

[Schoenfeld 85] A. SCHOENFELD : *Mathematical Problem Solving* ; Academic Press, New York, 1985.

[Silver 85] E. A. SILVER (Ed.) : *Teaching and Learning Mathematical Problem Solving: Multiple Research Perspectives* ; Lawrence Erlbaum Assoc., Hillsdale, 1985.

[Sleeman 82] D.H. SLEEMAN, J.S. BROWN (eds) : *Intelligent Tutoring Systems* ; Academic Press, London, 1982.

OUTILS DE CALCUL ET RAISONNEMENT

[Abelson 89] H. ABELSON, M. EISENBERG, M. HALFANT, J. KATZENELSON, E. SACKS, G.J. SUSSMAN, J. WISDOM, K. YIP : *Intelligence in scientific computing* ; Communications of the ACM, Vol 32, n° 5, May 1989, pp. 546-562.

[Baron 85] M. BARON : *Quelques réflexions après la réalisation de SEME* ; Cognitiva 85, Paris, 4-7 Juin 1985, pp. 257-262.

[Bledsoe 86] W.W. BLEDSOE : *Some thoughts on proof discovery ;* Proc. of the IEEE Symposium on Logic Programming, Salt Lake City, 22-24 septembre 1986, pp. 2-10.

[Bundy 75] A. BUNDY : *Analysing Mathematical Proofs (or Reading Between the Lines)* ; IJCAI 1975, pp. 22-28.

[Bundy 83] A. BUNDY : *The Computer Modelling of Mathematical Reasoning ;* Academic Press, London, 1983.

[Bundy 85] A. BUNDY : *Discovery and Reasoning in Mathematics* ; IJCAI 85, Vol. 2, 1985, pp. 1221-1230.

[Calmet 87] J. CALMET : *Intelligent computer algebra system : myth, fancy or reality ?* ; LNCS n°296 (Trends in Computer Algebra), pp. 2-11.

[Davenport 86] J. DAVENPORT, Y.SIRET, E. TOURNIER : *Calcul formel - Systèmes et algorithmes de manipulations algébriques* ; Masson, 1986.

[Davis&Lenat 82] R. DAVIS, D. B. LENAT : *Knowledge-based systems in artificial intelligence* ; McGraw-Hill New-York, 1982.

[Fateman 85] R.J. FATEMAN : *Eleven Proofs of* $sin^2x+cos^2x = 1$, ACM SIGSAM Bulletin, Vol. 19, n° 2, May 1985, pp. 22-28.

[Greiner 88] R. GREINER, B. SILVER, S. BECKER, M. GRUNINGER : *A Review of Machine Learning at AAAI-87* ; Machine Learning, vol 8, n°1, aout 1988, pp. 79-92.

[Jenks 84] R.D. JENKS : *A Primer: 11 keys to New SCRATCHPAD* ; Proc. EUROSAM 84, LNCS N°174, Springer Verlag, 1984, pp.123-147.

[Jurkovic 86] N. JURKOVIC : *Educational symbolic manipulation on a microcomputer* ; ACM Symsac 86, Waterloo, Ontario, july 21-23, 1986, pp. 154-156.

[Laublet 88] P. LAUBLET : *Apprentissage automatique en mathématique* ; Deuxième université d'été : Intelligence artificielle et enseignement des mathématiques, IREM Toulouse, juilllet 1988, pp. 111-133.

[Laublet 91] P. LAUBLET : *Hybrid Knowledge Representation and Theorem Proving in Mathematics* ; Rapport LAFORIA n°04/91, février 91.

[Merialdo 79] B. MERIALDO : *Représentation des ensembles en démonstration automatique* ; Thèse 3eme cycle Paris VI, mars 1979.

[Moses 71] J. MOSES : *Symbolic Integration: The Stormy Decade* ; CACM, vol. 14, n° 8, 1971, pp. 548-560.

[Nicaud 87] J.F. NICAUD : *APLUSIX : un système expert de résolution pédagogique d'exercices d'algèbre* ; Thèse Université de Paris-Sud, Centre d'Orsay, Décembre 1987.

[Pastre 84] D. PASTRE : *MUSCADET : Un système de démonstration automatique de théorèmes utilisant connaissances et métaconnaissances en mathématiques* ; Thèse d'état Paris VI, 22 nov. 1984.

[Pastre 85] D. PASTRE : *Maths et Méta* ; Colloque Intelligence Artificielle, Toulouse, Publication du Lab. C.F.Picard, 16-20 septembre 1985.

[Pastre 89] D. PASTRE : *MUSCADET: An Automatic Theorem Proving System Using Knowledge and Metaknowledge in Mathematics* ; Artificial Intelligence, n° 38, 1989, pp. 257-318.

[Pastre 90a] D. PASTRE : *Check or discover proofs* ; Rapport LAFORIA n°21/90, juillet 90.

[Pastre 90b] D. PASTRE : *Difficulty and quality of proofs* ; Cognitiva 90, Madrid, novembre 90.

[Pitrat 70] J. PITRAT : *Un Programme de Démonstration de Théorèmes*, Dunod, 1970.

[Purtilo 89] J.M. PURTILO : *Minion: An environment to organize mathematical problem solving* ; ACM-SIGSAM ISSAC 89, July 17-19, 1989, pp. 147-154.

[Risch 69] R. H. RISCH : *The problem of integration in finite terms* ; Transactions of the AMS, vol. 139, may 1969, pp. 167-189.

[Slagle 63] J.R.SLAGLE : *A heuristic program that solve symbolic integration problems in freshman calculus* ; in Computer and Thought, McGraw-Hill, 1963, pp. 191-203.

[Steels 85] L. STEELS : *Second generation Expert Systems* ; Future Generation Computer Systems, vol.1, n°4, juin 85, pp. 213-221.

[Sterling 89] L. STERLING, A. BUNDY, L. BYRD, R. O'KEEFE, B. SILVER : *Solving Symbolic Equations with PRESS* ; J. Symbolic Computation, n°7, 1989, pp. 71-84.

[Sutor 87] R.S. SUTOR, R. D. JENKS : *The Type Inference and Coercion Facilities in the Scratchpad II Interpreter* ; Proc. SIGPLAN'87 Symposium on Interpreters and Interpretive Techniques, St. Paul Minnesota, June 24-26, 1987, pp. 56-63.

[Vance 73] : R.R. VANCE : *The structure of an experimental, interactive, mathematical problem-solving system* ; Ph. D. of Computer Science, Purdue University (diffusion UMI), 1973.

[Vivet 84] M. VIVET : *Expertise mathématique et informatique : CAMELIA, un logiciel pour raisonner et calculer ;* Thèse d'état, Paris VI, juin 1984.

[Wolfram 88] S. WOLFRAM : *Mathematica : A system for doing Mathematics by Computer* ; Addison-Wesley, 1988.

OUTILS DE FORMULATION ET DE PRESENTATION

[Alem 88] L. ALEM : *Edora - An expert system for assistance in biological modeling* ; Rapport INRIA n° 99, oct. 1988.

[Ambler 89] A.L. AMBLER, M.M. BURNETT : *Influence of Visual Technology on the Evolution of Language Environments* ; IEEE Computer, Oct 1989, pp. 9-22.

[Arnold 68] A. ARNOLD. : *Présentation d'un langage de formalisation des démonstrations mathématiques naturelles* ; Symposium on Automatic Demonstration, Versailles, Lecture Notes in Mathematics, vol. 125, 1968, pp. 6-28.

[Arnon 88] D. ARNON, R. BEACH, K. McISAAC, C. WALDSPURGER : *Caminoreal : an interactive mathematical notebook*, Proceedings of the International Conference on Electronic Publishing, Document manipulation and typography, Nice, April 20-22, 1988, pp. 1-18.

[Bahlke 86] R. BAHLKE, G. SNELTING : *The PSG system : from formal language definitions to interactve programming environments*, ACM Transactions on Programming Languages and Systems, Vol 8, n° 4, October 1986, pp. 547-576.

[Baulac 89] Y. BAULAC, J.M. LABORDE : *Sur l'interface d'un Cahier de brouillon informatique pour la géométrie* ; Rapport IMAG (lsd), RR 787-I-, octobre 1989.

[Beaudoin-Lafon 85] M. BEAUDOIN-LAFON : *Vers des interfaces graphiques évolués : UFO, un méta-modèle d'interaction* ; Thèse Univ. Paris Sud, 31 Oct. 1985.

[Brown 88] M.H. BROWN : *Exploring Algorithms Using Balsa-II* ; IEEE Computer, Mai 1988, pp 14-36.

[Conklin 87] J. CONKLIN : *Hypertext: An Introduction and Survey* ; IEEE Computer, sept 1987, pp 17-41.

[Cuppens F. 88] F. CUPPENS : *Comment fournir des réponses coopératives aux requêtes à une base de données* ; Thèse ENSAE, Novembre 1988.

[Halasz 88] F.G. HALASZ : *Relections on NoteCards: seven issues for the next generation of hypermedia systems* ; CACM vol 31, n°7, july 1988, pp. 836-852.

[Kaltenbach 87] M. KALTENBACH, C. FRASSON : *DYNABOARD: User animated display of deductive proofs in mathematics* ; Int. J. Man-Machine Studies (1989) 30, pp. 149-170. (cf. aussi une version préliminaire de M. KALTENBACH : rapport INRIA N°700, juillet 1987).

[Knuth 79] D.E. KNUTH : *TEX and Metafont, New Directions in Typesetting* ; Digital Press, 1979.

[Liem 89] I. LIEM : *Visualisation de l'exécution des programmes pour l'enseignement de la programmation* ; Thèse Univ. Joseph Fourier, Grenoble I, 22 sept. 1989.

[McArthur 88] D. McARTHUR, C. BURDOF, T. PRIMSETH, A. ROBIN, C. STASZ : *Multiple representations of mathematical reasoning* ; ITS-88, Montréal, juin 1988, pp. 485-490.

[Quadrat 89] J.P. QUADRAT, A SULEM, J.P. CHANCELIER, C. GOMEZ : *PANDORE: Un système expert pour l'optimisation des systèmes dynamiques* ; INRIA, 1989.

[Quint 83] V. QUINT : *Un système interactif pour la production de documents mathématiques* ; TSI, vol. 2, n° 3, mai-juin 1983, pp. 179-190.

[RIA 90] Revue d'intelligence artificielle, Numéro spécial Explication, vol 4, n°2, 1990.

[Zukerman 86] I. ZUKERMAN, J. PEARL : *Comprehension-driven generation of meta-technical utterances in math tutoring* ; AAAI 86 t.1, pp. 606-611. Aussi : I. ZUKERMAN : *Computer generation of meta-technical utterances in tutoring mathematics* ; Ph.D. Univ. of California, UMI, 1986.

INTERFACES ET ERGONOMIE

[Barthet 88] M.F. BARTHET : *Logiciels interactifs et ergonomie* ; Dunod, 1988.

[Boy 88] : G.A. BOY : *Assistance à l'opérateur : une approche de l'intelligence artificielle* ; Ed. Teknea, 1988.

[Canals 88] G. CANALS, D. COLNET, S. CRUZLARA SILVA, J.C. DERNIAME : *GEPI : un générateur d'environnements de programmation intégrés,* Journées Internationales : Le génie logiciel et ses applications, Toulouse, 5-9 décembre 1988, Vol 1.

[Coutaz 84] J. COUTAZ-RAYMOND, M. HERRMANN : *Adèle et le médiateur compositeur*, 2ème Colloque de Génie Logiciel, Nice, 4-6 juin 1984, pp. 195-212.

[Coutaz 86] J. COUTAZ : *Abstractions pour Interfaces Interactives* ; TSI, vol 5, n° 4, 1986, pp. 239-250.

[Coutaz 90] J. COUTAZ : *Interfaces homme-ordinateur : conception et réalisation* ; Dunod, 1990.

[Dang 88] W. DANG : *Intelligence in a User Interface Management System,* ERGO-IA Biarritz, 4-6 octobre 1988, pp.125-137.

[Falzon 89] P. FALZON : *Ergonomie cognitive du dialogue* ; Presses Universitaires de Grenoble, 1989.

[Franchi-Zannettacci 88] P. FRANCHI-ZANNETTACCI : *Attribute specifications for graphical interface generation* ; Rapport Inria n° 937, Déc 88.

[Gatine 87] O. GATINE : *Calcul symbolique : un programme de dérivation* ; Rapport de T.E.P. ENSAE, 1987.

[INRIA 87] A. BISSERET, P. FALZON, B. SENACH, resp. trois articles sur le thème : *L'ergonomie cognitive de l'interaction homme-calculateur* ; bull. liaison INRIA n°115, 1987.

[Kajler 90] N. KAJLER : *Eléments de spécification pour l'interface utilisateur d'un système de calcul algébrique formel* ; Rapport INRIA, n°1220, mai 1990. (cf aussi LNCS vol 429, pp. 235-244)

[Mathé 90] N. MATHE : *Assistance intelligente au contrôle de processus : application à la télémanipulation spatiale* ; Thèse ENSAE n°57, 22 nov. 1990.

[Moulis 89] G. MOULIS : *Manipulation de formules mathématiques* ; ISMM conf. mini and micro. and their applications, Zurich, juin 1989, pp. 54-58.

[Rissland 84] E.L. RISSLAND : *Ingredients of intelligent user interfaces,* Int. J. Man-Machine Studies, vol 21, 1984, pp. 377-388.

[Scapin 88] D.L. SCAPIN : *Vers des outils formels de description des tâches orientés conception d'interfaces*, Rapport INRIA, n° 893, Septembre 88.

[Senach 90] B. SENACH : *Evaluation ergonomique des Interfaces Homme-Machine : une revue de la littérature.* ; Rapport INRIA n°1180, Mars 1990.

[Shneiderman 83] B. SHNEIDERMAN : *Direct manipulation : a step Beyond Programming Languages*, IEEE Computer, August 1983, pp. 57-69.

[Visser 90] W. VISSER : *Acquisition de connaissances: l'approche de la psychologie cognitive illustrée par le recueil d'expertise en conception* ; JAC 90 du PRC-IA, Lannion, 27 Avril 1990.

OUTILS DE PROGRAMMATION ET ENVIRONNEMENTS DE MANIPULATION

[Barston 84] D.R. BARSTON, H.E. SHROBE, E. SANDEWALL (Eds) : *Interactive programming environments* ; McGraw-Hill, 1983.

[Bates 85] J.L. BATES and R.L. CONSTABLE : *Proofs as programs* ; ACM Transactions on Programming and Systems, Vol 7, n° 1, January 1985, pp. 113-136.

[Borras 87] P. BORRAS, D. CLEMENT, T. DESPEYROUX, J. INCERPI, G. KHAN, B. LANG, V. PASCUAL : *CENTAUR : the system* ; Rapport Inria n° 777, Déc. 87.

[Burstall 77] R.M. BURSTALL, J. DARLINGTON : *A transformation system for developing recursive programs* ; Journal of the association for computing machinery, vol. 24, n° 1, January 1977, pp. 44-67.

[Conchon 85] A. CONCHON : *Projet CONCERTO : le poste de travail* ; l'Echo des Recherches, n° 119/120, 1er/2° trimestres 1985, pp. 49-56.

[Conchon 86] A. CONCHON, J. CAMACHO, A.M. RASSER : *Une session sur le poste de travail CONCERTO*, 3ème Colloque Génie Logiciel, Versailles, 27-30 mai 1986, pp. 57-68.

[Deslaugiers 85] M. DESLAUGIERS : *Conception et réalisation du module d'affichage d'un méta-décompilateur* ; Thèse de 3° Cycle d'Informatique, Université Paris VII, 26 Septembre 1985.

[De Millo 79] R.A. DE MILLO, R.J. LIPTON, A.J. PERLIS : *Social Processes and Proofs of Theorems and programs* ; CACM, vol 22, n° 5, Mai 1979, pp. 271-280.

[Donzeau-Gouge 83] V. DONZEAU-GOUGE, G. KAHN, B. LANG, B. MELESE, E. MORCOS : *Outline of a tool for document manipulation*, Proceedings of the IFIP 9th World Computer Congress, Paris, 19-23 september, 1983, pp. 615-620.

[Fetzer 88] J.H. FETZER : *Program verification: The very idea* ; CACM vol. 31, n°9, sept 1988, pp. 1048-63.

[Kaiser 88] G.E. KAISER, P.H. FEILER, F. JALILI, J.H. SCHLICHTER : *A retrospective on DOSE : an interpretive approach to structure editor generation* ; Software-practice and experience, Vol. 18(8), August 1988, pp. 733-748.

[Shapiro 86] S. SHAPIRO : *The Art of PROLOG Advanced Programming Techniques* ; MIT Press, 1986.

LANGAGES A OBJETS

[Cardelli 85] L. CARDELLI, P. WEGNER : *On understanding types, data abstraction, and polymorphism*, Computing Surveys, Vol 17, n° 4, December 1985, pp. 471-522.

[Carre 89] B. CARRE : *Méthodologie orientée objet pour la représentation des connaissances, concept de point de vue, de représentation multiple et évolutive d'objet* ; Thèse Univ. Lille, 31 janv. 1989.

[Ferber 89] J. FERBER : *Objets et agents: une étude des structures de représentation et de communications en Intelligence Artificielle* ; Thèse d'état Univ. Paris VI, 8 juin 1989.

[Lenat 90] D.B. LENAT, R.V. GUHA, K. PITTMAN, D. PRATT, M. SHEPHERD : *CYC : Toward programs with common sense* ; CACM vol 33, N° 8, Août 90, pp. 30-49.

[Masini 89] G. MASINI & al. : *Les langages à objets* ; Interéditions, Paris, 1989.

[Meyer 88] B. MEYER : *Cépage : toward computer-aided design of software*, The Journal of Systems and Software, 1988, pp. 419-429.

[Motschnig 90] R. MOTSCHNIG-PITRICK : *A Framework for the Support of a Common Structural Level for Software-, Data Base-, and Knowledge-Based Systems* ; J. Systems Software, N° 12, 1990, pp. 125-137.

[Prade 84] H. PRADE : *Modèles de raisonnement approché pour les Systèmes Experts;* AFCET-RFIA, Paris, t II, 25-27 janv. 1984, pp. 355-373.

[Vogel 88] C. VOGEL : *Génie Cognitif* ; Masson, 1988.

[Wolstencroft 89] J. WOLSTENCROFT : *Restucturing, Reminding and Repair: What's Missing from Models of Analogy* ; AICOM, vol. 2, n° 2, june 1989, pp. 58-71.

ANALYSE ET REPRESENTATION DES LANGAGES

[Bouhier 89] M. BOUHIER : *Génération de la sémantique d'une formule mathématique* ; Rapport IMAG n° RR783-M-, Juin 89.

[Cohen 87] J. COHEN, T.J. HICKEY : *Parsing and Compiling Using Prolog* ; ACM Transactions on Programming Languages and Systems, Vol. 9, n° 2, April 1987, pp. 125-163.

[Coulon 86] D. COULON, D. KAYSER : *Informatique et langage naturel : Présentation générale des méthodes d'interprétation des textes écrits* ; TSI vol. 5, n°2, 1986, pp. 103-128.

[Deransart 85] P. DERANSART, J. MALUSZYNSKI : *Relating logic programs and attribute grammars* ; J. logic programming, n°2, 1985, pp. 119-155.

[Lehnert 83] W.G. LEHNERT, M.G. DYER, P.N. JOHNSON, C.J. YANG, S. HARLEY :*"BORIS--An experiment in in-depth understanding of narratives"*, Artificial Intelligence, n° 20, 1983, pp. 15-62.

[Matsumoto 86] Y. MATSUMOTO, H. TANAKA, M. KIYONO : *BUP : A Bottom-Up Parsing System For Natural Languages* ; Dans : *Logic programming and its applications*, Ablex publishing, 1986.

[Pereira 80] F.C.N. PEREIRA, D.H.D. WARREN, *Definite clause grammar for language analysis*, Artificial Intelligence 13, 1980, pp. 231-278.

[Sabah 89] G. Sabah : *L'intelligence artificielle et le langage* ; Hermes, vol. 1 : *Représentation des connaissances* ; 2° ed. 1990 ; vol. 2 : *Processus de compréhension* ; 1989.

[Schank 77] R.G. SCHANK, R.P. ABELSON : *Scripts, Goals, Plans and Understanding* ; Lawrence Erlbaum, Wiley, 1977.

[Sowa 84] J.F. SOWA : *Conceptual structures: information processing in mind and machine* ; Addison-Wesley, 1984.

DESCRIPTIF DES ANNEXES

page de renvoi	intitulé de l'Annexe	page d' annexe
46	Le jeu du professeur Rufus Isaacs	A-1
47 à 49	Les produits cartésiens discrets : typologie non exhaustive	A-2
47 à 49	Exemples de représentations spatiales	A-4
47 à 49	Exemples de structures discrètes	A-8
49	Reconnaissance des "formes symboliques"	A-11
52, 53	Ergonomie du Langage Mathématique : expressivité, homogénéité, mnémonique	A-14
126	Parcours et habillage d'une preuve	A-17
129	Problème résolu par transformation explicite de problème	A-18
130	Exemple de stratégie de développement-identification	A-19
133,142	Influence du contexte de définition pour l'obtention d'un résultat	A-20
143	Présentation de démonstation proposée dans [Whitworth 1875]	A-21
143	idem, proposée par Paul Depasse pour satisfaire ses besoins de clarté et de rigueur (1989)	A-22
149	Preuve par récurrence (autre version)	A-24
197	La forme des résultats varie selon l'argument choisi	A-26
223	La résolution d'un problème entraîne des manipulations de formules très variées sans fil directeur apparent : analyse d'un exemple	A-27
260	Session de saisie d'une formule	A-28
266	Code Flavors associé à la figure p 267	A-29
271	Schéma général de traitement du langage (projet japonais de 5° génération)	A-33
272	Adaptation du langage par introduction de notations	A-34
275	Analyseur	A-39
285	La définition et l'usage d'une notation : exemples d'utilisation de Σ	A-48
ORAL	Tableau synoptique des transparents de la soutenance orale	A-52

Le jeu du Professeur Rufus Isaacs (p308 du livre de BERGE : Graphes et hypergraphes)

titre

légende : point important (commentaire)

• Complémentarité du texte et de la figure,
• Pluralité des informations présentes sur la figure.

EXEMPLE 4. X est l'ensemble des points (p, q) du plan à coordonnées entières ; les deux joueurs jouent en biffant à tour de rôle un point de X ; si x est le dernier point biffé, le joueur ayant le trait doit faire son choix soit sur une des perpendiculaires issues de x aux axes de coordonnées, soit sur leur bissectrice ; par exemple, après le sommet $x = (4, 1)$, le choix doit se faire entre les sommets (4, 0), (3, 1), (2, 1), (1, 1), (0, 1), (3, 0). Le premier joueur qui arrive à atteindre le sommet (0, 0) a gagné.

Plan discret

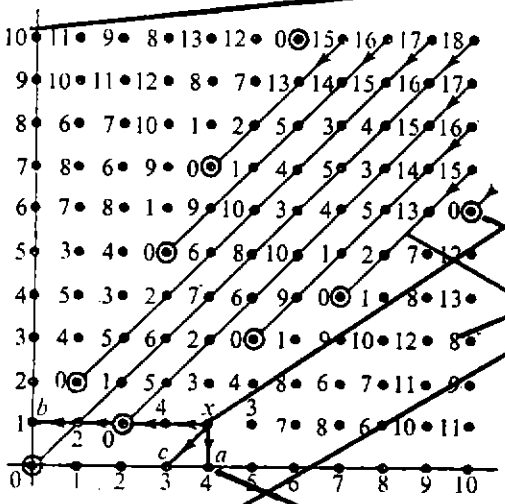


Fig. 14.8.

Possibilités de déplacement

Valeurs de la fonction d'évaluation du problème.

Mise en valeur des choix gagnants.

Signalisation des coups diagonaux gagnants.

Le graphe (X, Γ) est sans circuits, et par conséquent admet une fonction de Grundy $g(x)$ dont la valeur en chacun des sommets est indiquée sur la figure 14.8 ; les choix gagnants sont entourés d'un rond. Bien qu'ils forment un ensemble symétrique par rapport à la diagonale principale, on notera l'irrégularité de leur répartition.

(Nous devons ce jeu à l'obligeance du professeur Rufus Isaacs.)

Pragmatique de description : choix d'un exemple x respectant la lisibilité de la figure (par ailleurs, introduction de 3 cas : a, b, c non exploités dans le texte)

Théorème 9. *Si le graphe admet un noyau S, et si un joueur choisit un sommet dans S, ce choix lui assure le gain ou la nullité.*

En effet, si le joueur (A) choisit $x_1 \in S$, ou bien $\Gamma(x_1) = \varnothing$ et alors il a gagné la partie ; ou bien son adversaire sera contraint à choisir un sommet x_2

La Figure est ici introduite lorsque la description textuelle devient insuffisante

dans $X - S$, et alors, le coup suivant, le joueur (A) pourra choisir à nouveau un sommet x_3 dans S, et ainsi de suite. Si à un moment donné l'un des joueurs gagne en choisissant un sommet x_k tel que $\Gamma(x_k) = \varnothing$, on a $x_k \in S$, donc le joueur gagnant est nécessairement (A).

C. Q. F. D.

Ses propriétés synthétiques sont exploitées dans le texte.

La figure servira aussi de support d'illustration des abstractions introduites ultérieurement.

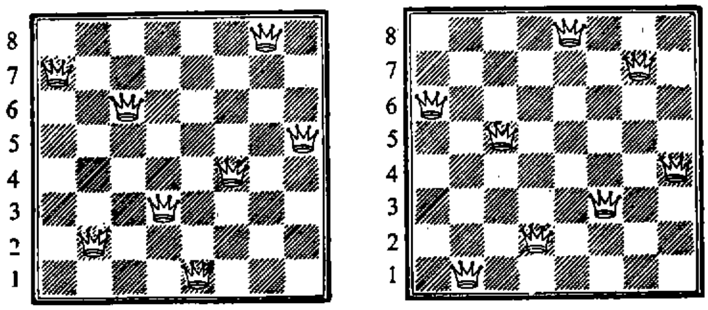
Une méthode fondamentale pour bien jouer consistera donc à calculer une fonction de Grundy $g(x)$, si elle existe ; on en déduit un noyau

$$S = \{ x \mid g(x) = 0 \}$$

pour le graphe considéré. Si le sommet initial x_0 est tel que $g(x_0) = 0$, le joueur (A) est dans une position critique, car son adversaire peut s'assurer le gain ou la nullité ; si au contraire $g(x_0) \neq 0$, le joueur (A) s'assurera le gain ou la nullité en choisissant un sommet x_1 tel que $g(x_1) = 0$.

A-1

Les Produits Cartésiens discrets : typologie non exhaustive



(72631485) (61528374)

[BERGE] *Fig.* **13.1.**

Echiquier : référence par numérotation des lignes et colonnes par ex : A7

Codage "équivalent" après mathématisation du problème des huit reines

et la loi de composition est donnée par la « table de multiplication » suivante :

	s_1	s_2	s_3	s_4	s_5	s_6
s_1	s_1	s_2	s_3	s_4	s_5	s_6
s_2	s_2	s_3	s_1	s_5	s_6	s_4
s_3	s_3	s_1	s_2	s_6	s_4	s_5
s_4	s_4	s_6	s_5	s_1	s_3	s_2
s_5	s_5	s_4	s_6	s_2	s_1	s_3
s_6	s_6	s_5	s_4	s_3	s_2	s_1

Table de multiplications : référence par nommage des lignes et colonnes

(on a adopté la convention suivante : pour calculer un produit xy à l'aide de cette table, on porte x en *ligne* et y en *colonne*. Exemple : $s_2s_4 = s_5$, $s_4s_3 = s_5$).

interprétation de la table

Cet exemple prouve l'existence de **groupes finis**, i.e. de groupes à un nombre fini d'éléments; il est clair d'ailleurs que $\mathfrak{S}(X)$ est fini dès que (et seulement si) l'ensemble X est fini.

On résumera ceci par le tableau : [BERGE, p 151]

Type du sommet	*épais*	*fin*	*mixte*	*inaccessible*
épais	arête épaisse			arête épaisse
fin		arête fine		arête fine
mixte				X
inaccessible	arête épaisse	arête fine	X	

Tableau de correspondance : référence par nommage des lignes et colonnes

Théorème 8. *Deux sommets épais ne peuvent être reliés que par une arête épaisse ; deux sommets fins ne peuvent être reliés que par une arête fine ; un sommet mixte et un sommet inaccessible ne peuvent être reliés par aucune arête ; un sommet épais et un sommet inaccessible ne peuvent être reliés que par une arête épaisse : un sommet fin et un sommet inaccessible ne peuvent être reliés que par une arête fine.*

description du tableau en langage de Phrases

 [Graham 89]

Table 256 Second-order Eulerian triangle.

n	$\left\langle\!\left\langle {n \atop 0} \right\rangle\!\right\rangle$	$\left\langle\!\left\langle {n \atop 1} \right\rangle\!\right\rangle$	$\left\langle\!\left\langle {n \atop 2} \right\rangle\!\right\rangle$	$\left\langle\!\left\langle {n \atop 3} \right\rangle\!\right\rangle$	$\left\langle\!\left\langle {n \atop 4} \right\rangle\!\right\rangle$	$\left\langle\!\left\langle {n \atop 5} \right\rangle\!\right\rangle$	$\left\langle\!\left\langle {n \atop 6} \right\rangle\!\right\rangle$	$\left\langle\!\left\langle {n \atop 7} \right\rangle\!\right\rangle$	$\left\langle\!\left\langle {n \atop 8} \right\rangle\!\right\rangle$
0	1								
1	1	0							
2	1	2	0						
3	1	8	6	0					
4	1	22	58	24	0				
5	1	52	328	444	120	0			
6	1	114	1452	4400	3708	720	0		
7	1	240	5610	32120	58140	33984	5040	0	
8	1	494	19950	195800	644020	785304	341136	40320	0

"Triangle de Pascal": référence de type $(a, f(a))$ ayant une présentation spécifique des axes et un contenu triangulaire

tous ces graphes de fonction ont des représentations très différentes en langage de figure! (sans parler des désignations)

Une position irréductible étant définie par un tas unique de $\bar{x}_1$ allumettes, la fonction g est déterminée par le tableau suivant : [BERGE p 312]

$\bar{x}_1$ =	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
$g(\bar{x}_1)$ =	0	0	1	0	2	1	0	2	1	0	2	1	3	2	1	3	2	4	3	0

Tableau de coordonnées point-valeur : référence par la coordonnée unidimensionnelle (notation ligne décrite; par ex. "$g(\bar{x}_1)$=")

Graphe de fonction entière bornée: référence par les coordonnées

cf. le jeu du Professeur Rufus Isaacs

Cette *écriture binaire* se forme comme l'écriture décimale, et la correspondance est facile à établir : [BERGE p 305]

écriture décimale	*écriture binaire*	*développement binaire*
0	0	(0)
1	1	(1)
2	10	(0, 1)
3	11	(1, 1)
4	100	(0, 0, 1)
5	101	(1, 0, 1)
6	110	(0, 1, 1)
7	111	(1, 1, 1)
8	1 000	(0, 0, 0, 1)
9	1 001	(1, 0, 0, 1)
10	1 010	(0, 1, 0, 1)
11	1 011	(1, 1, 0, 1)

Tableau informel : référence par le contenu

Matrices et Déterminants : référence par ligne et colonne

En vertu de la proposition 6 du théorème 13, pour que G soit une arborescence, il suffit de démontrer que le graphe G est connexe.

Supposons qu'il n'en soit pas ainsi ; on peut alors décomposer Δ_1 en deux matrices carrées B' et B'', de la façon suivante : [BERGE, lemme du Th. 20]

$$\Delta_1 = \begin{array}{c|cc} & S & T \\ \hline S & B' & 0 \\ T & 0 & B'' \end{array} = \text{Dét}\,(B').\text{Dét}\,(B'')\,.$$

un système de référence par nommage peut être rajouté

Exemples de Représentations Spatiales

Les représentations spatiales sont les plus difficiles à construire et à interpréter !

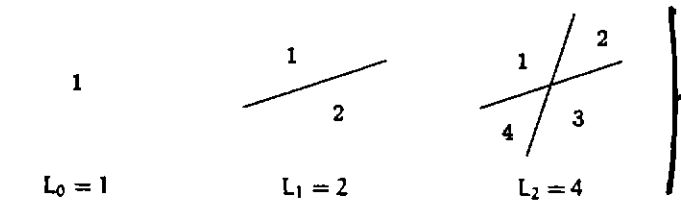
1.2 LINES IN THE PLANE

Our second sample problem has a more geometric flavor: How many slices of pizza can a person obtain by making n straight cuts with a pizza knife? Or, more academically: What is the maximum number L_n of regions

[Graham 89]

defined by n lines in the plane? This problem was first solved in 1826, by the Swiss mathematician Jacob Steiner [277].

Again we start by looking at small cases, remembering to begin with the smallest of all. The plane with no lines has one region; with one line it has two regions; and with two lines it has four regions:



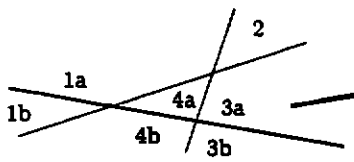
$L_0 = 1$ $L_1 = 2$ $L_2 = 4$

Les constructions possibles sont nombreuses et très peu d'éléments sont prédéfinis pour aider à les construire.

(Each line extends infinitely in both directions.)

Les signes employés nécessitent une interprétation plus générale que ce qu'ils sont sensés représenter au premier abord.

Sure, we think, $L_n = 2^n$; of course! Adding a new line simply doubles the number of regions. Unfortunately this is wrong. We could achieve the doubling if the nth line would split each old region in two; certainly it can split an old region in at most two pieces, since each old region is convex. (A straight line can split a convex region into at most two new regions, which will also be convex.) But when we add the third line—the thick one in the diagram below—we soon find that it can split at most three of the old regions, no matter how we've placed the first two lines:



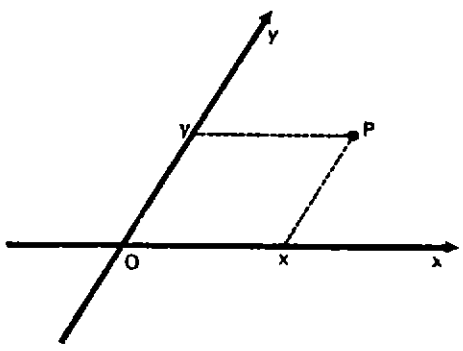
Pragmatique de la numérotation : une rupture se produit dans l'algorithme de numérotation qui doit être repensé dès L_3.

Thus $L_3 = 4 + 3 = 7$ is the best we can do.

[p 52, Godement 66]

Les coordonnées cartésiennes constituent le principal élément prédéfini utile pour les représentations spatiales

Le fait que l'opération précédente soit nommée en l'honneur de Descartes s'explique comme suit. Dans le « plan » de la Géométrie élémentaire, choisissons deux axes de coordonnées Ox, Oy et des unités de longueur sur ces axes; on peut alors définir l'abscisse et l'ordonnée de tout point P du plan; les désignant par x et y, il est naturel de ne pas faire de différence entre le point P et le couple (x, y); du reste, si P′ est un autre point, de coordonnées x' et y', la relation $P = P'$ équivaut évidemment à $x = x'$ et $y = y'$, i.e. à $(x, y) = (x', y')$. Désignant par $\mathbf{R}$ l'ensemble des nombres réels, on voit donc que le choix d'un système de coordonnées dans le plan permet d'assimiler le plan à l'ensemble



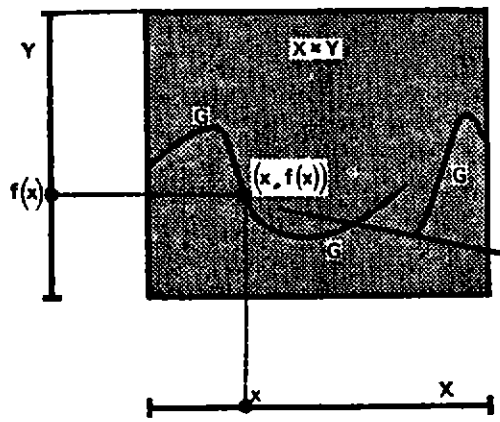
$$\mathbf{R} \times \mathbf{R} = \mathbf{R}^2$$

des couples de nombres réels — c'est ce qui justifie la référence à l'inventeur des systèmes de coordonnées.

utilisation des coordonnées cartésiennes (présentation inusuelle)

La condition (F 1) signifie que G est un graphe (n° 1); on dit que G est le **graphe de la fonction** f; d'après (F 2), pour tout $x \in X$ il existe un $z \in G$ tel que $x = pr_1(z)$; on a donc

$$pr_1(G) = X, \qquad pr_2(G) \subset Y.$$



La figure est implicitement commentée dans le texte.

Soit x un élément de X; l'unique élément y de Y tel que $(x, y) \in G$ s'appelle la **valeur de la fonction** f **en** x, et on utilise pour le désigner la notation

$$y = f(x);$$

il est alors clair que le graphe G de f est l'ensemble des couples de la forme $(x, f(x))$ où $x \in X$, ce qui est conforme à l'idée intuitive qu'on se fait d'une fonction.

L'interprétation de la figure est mise en évidence par des signes conventionnels détaillant les liens présents dans la figure.

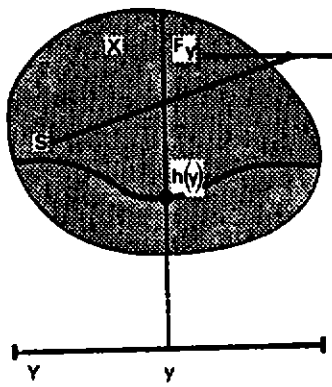
THÉORÈME 4. *Soit* $f: X \to Y$ *une application. Les conditions suivantes sont équivalentes:*

a) f *est surjective;*

b) il existe une application $h: Y \to X$ *telle que* $f \circ h$ *soit l'application identique de* Y *sur* Y.

Si *b)* est remplie, on a $f(h(y)) = y$ pour tout $y \in Y$, donc tout $y \in Y$ est de la forme $f(x)$, et f est donc bien surjective.

[Godement 66 p 63]

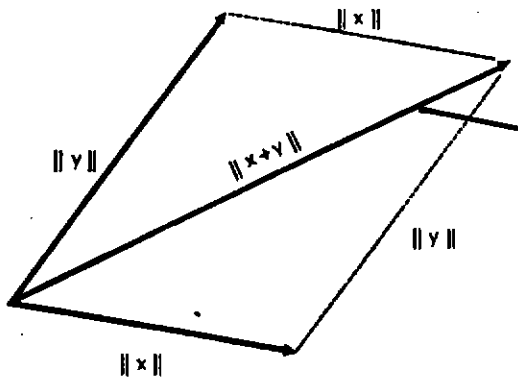


Le rédacteur exploite la familiarité de l'interprétation de ce type de figure pour introduire des symboles non définis dans le texte. L'interprétation de la figure devient laborieuse...

Dans le cas du produit scalaire usuel $(x|y)$, il est clair que $\|x\|$ n'est autre que la *longueur* du vecteur x. Le Corollaire 2 démontre donc le résultat suivant :

COROLLAIRE 3. *Dans un triangle, la longueur de chaque côté est inférieure à la somme des longueurs des deux autres.*

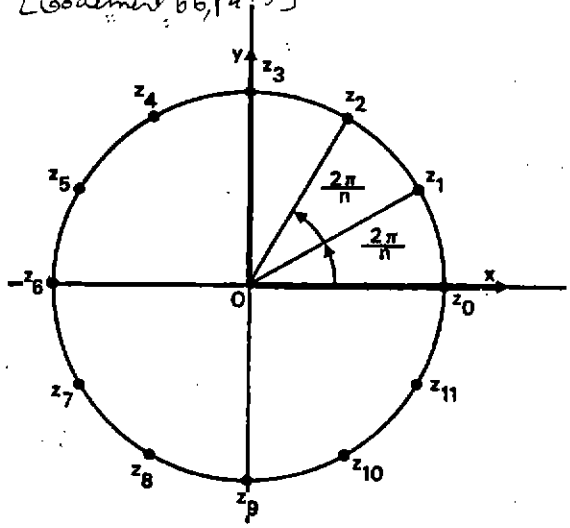
De nombreux objets mathématiques ont une interprétation sous forme de figure géométrique.



L'interprétation des vecteurs est plus simple (et plus automatisable) que celle des graphes de fonction

Lorsque K = C, les racines n^e de l'unité sont données par la formule

[Godement 66, p4 19]



$$z_k = \cos(2k\pi/n) + i.\sin(2k\pi/n) \quad (0 \leqslant k \leqslant n-1),$$

autrement dit sont représentées, dans le plan complexe, par les sommets d'un polygone régulier de n côtés inscrit dans le cercle unité (cf. la figure ci-contre). En effet, la formule de Moivre (§ 9, n° 6) montre que

$$z_k^n = \cos(2k\pi n/n) + i.\sin(2k\pi n/n) = 1,$$

de sorte que la formule ci-dessus représente n racines, deux à deux distinctes, de l'équation (8); celle-ci étant de degré n ne saurait en posséder d'autres.

Les complexes ont une représentation géométrique simple

Ici encore, la diversité des graphes de fonctions et de leur présentation apparaît clairement

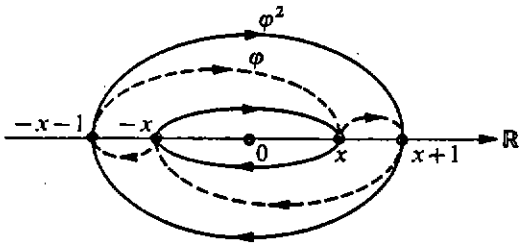


Fig. 3.6. — Deux composantes connexes du graphe $H = (X, \varphi^2)$ (où $\varphi^2(x) = -x$).

[BERGE; p38]

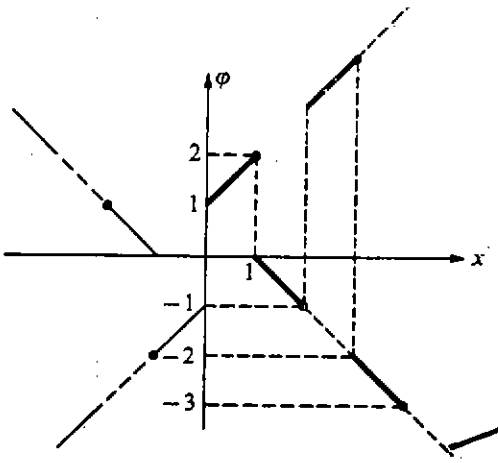


Fig. 3.7. — Fonction φ telle que $\varphi^2(x) = -x$.

Deux modes pour présenter des ensembles infinis: en compréhension et en extension.

Cette figure tire part de généralisations implicites pour anticiper l'alternance à l'infini.

68. Volume engendré par une surface plane fermée tournant autour d'un axe. — Soit une courbe *fermée* C située dans le plan de la figure. Supposons que cette courbe tourne autour de l'axe des x et calculons

[Lavrentiev, p 182]

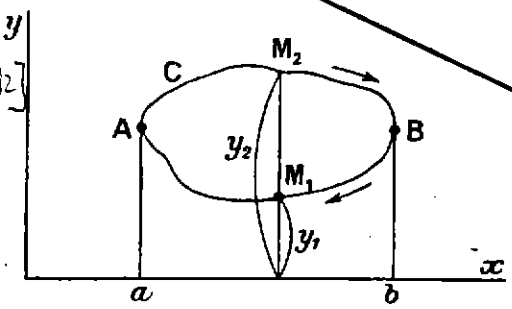
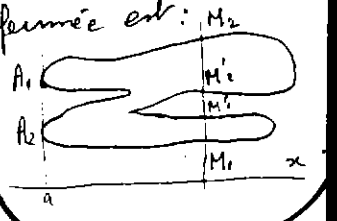


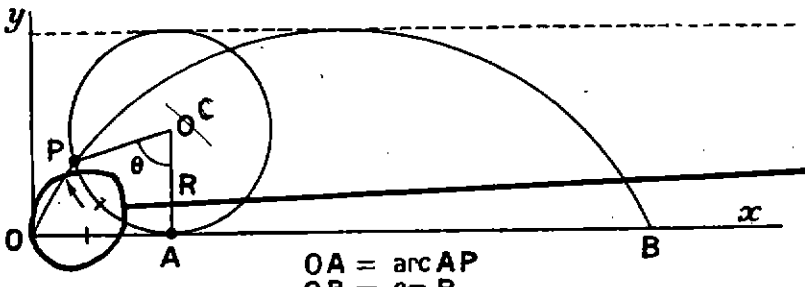
FIG. 58.

le volume qui sera engendré par cette surface S (fig. 58). Traçons les tangentes extrêmes à la courbe, en A et en B, et menons une parallèle à l'axe des y, d'abscisse x; elle coupe la courbe en deux points M_1 et M_2 dont les ordonnées sont y_1 et y_2.

Les figures (voire le raisonnement) s'appuyer sur des exemples typiques qui ne sont pas aussi généraux que les mots le laissent entendre. Ici, une autre courbe fermée est:



La dynamique ou une dimension supérieure à celle du support introduisent des conventions pragmatiques de description et d'interprétation encore plus sophistiquées...



$OA = \text{arc}\,AP$
$OB = 2\pi R$

[Quinet 62, p 136]

FIG. 28 *a*.

position instantanée et marqueurs d'évolution dynamique à interpréter

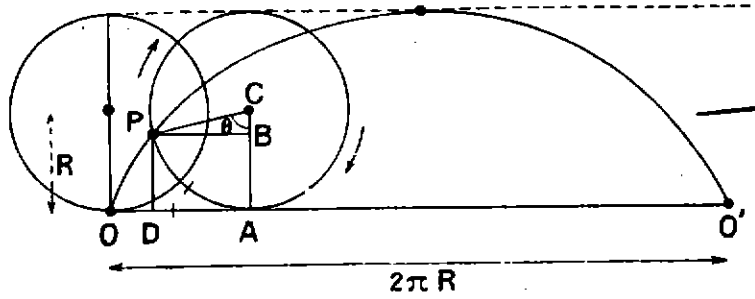


FIG. 28 *b*.

Superposition de deux instantanés qu'il faut analyser et différencier. La complexité du choix des symboles et des traits de construction à représenter est analysé dans l'exemple de la parabole (ch I §4.2)

La représentation 2.D d'un objet 3.D passe par des coupes ou des perspectives

RECHERCHE DES CENTRES DE GRAVITÉ [Quinet 62, p 205]

C'est ici le cas du tore, engendré par une circonférence de rayon *r* qui tourne autour d'un axe situé dans son plan, la distance de son centre à l'axe, soit R, étant plus grande que *r* (fig. 77).

La longueur de la courbe génératrice étant $2\pi r$, le théorème nous donne aussitôt

$$S = 2\pi r \cdot 2\pi R = 4\pi^2 R r.$$

Une coupe représentative est à choisir avec minutie: un mécanisme pertinent de génération de coupes est à étudier pour présenter une large classe d'objets 3-D. La coupe vient préciser une description globale de l'objet

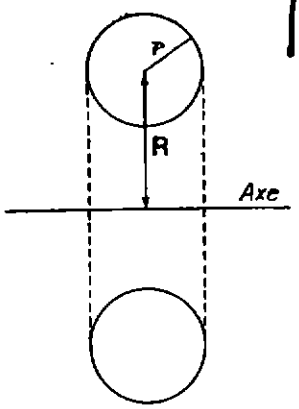


FIG. 77.

[Quinet 62, p 202]

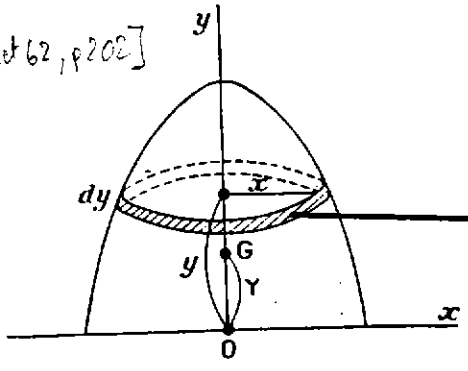


FIG. 73.

Une (bonne) perspective permet une description globale d'un objet 3.D moyennant des conventions d'interprétation comme l'expression du volume et des faces cachées.

Exemples de structures discrètes

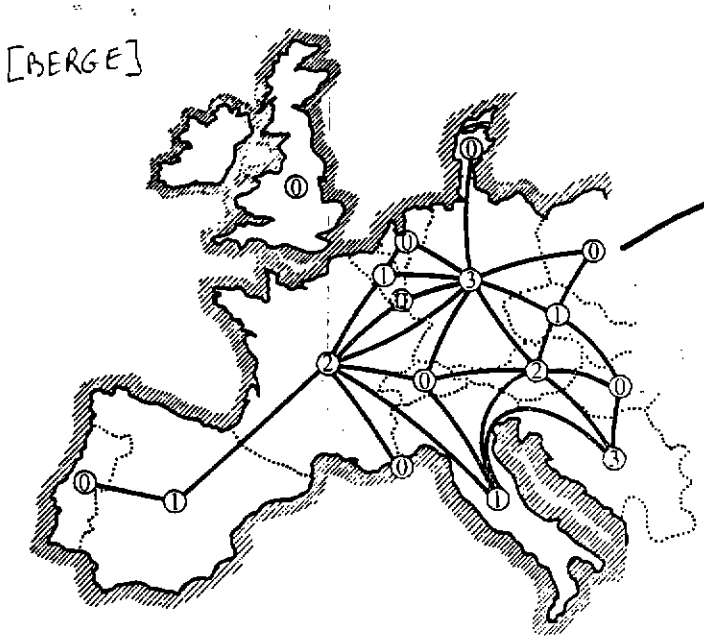


Fig. **15.1.** — Carte 4-chromatique. Le choix des couleurs 0, 1, 2, 3 correspond aux valeurs d'une fonction de Grundy du graphe symétrique planaire représenté en traits gras.

Les structures discrètes sont souvent issues d'autres représentations avec lesquelles elles peuvent se superposer : ici le cas du coloriage d'une carte.

La difficulté des structures discrètes réside dans le choix du graphe à représenter et dans celui de la disposition spatiale des nœuds et des arcs.

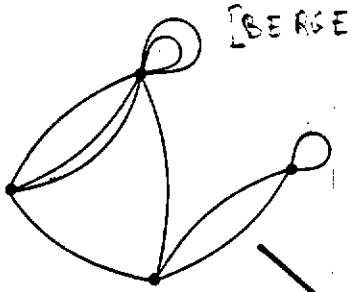


Fig. **1.3.** — Multigraphe.

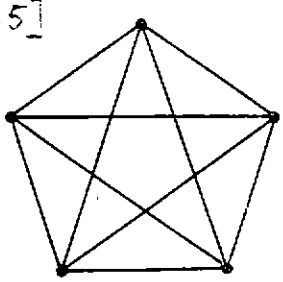


Fig. **1.4.** — Graphe complet K_5.

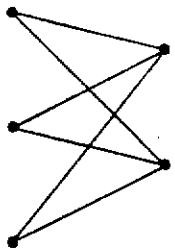


Fig. **1.5.** — Graphe biparti-complet $K_{3,2}$.

L'analyse de ces 3 figures nous incite à rechercher des règles pragmatiques de disposition des nœuds et de choix des arcs (par ex. présence de boucles et d'arcs multiples dès que cela est possible)

De tels schémas, graphes ou multigraphes, se rencontrent partout sous des noms divers : sociogrammes (psychologie), simplexes (topologie), circuits électriques (physique), diagrammes d'organisation (économie), réseaux de communication, arbres généalogiques, etc. ; il est parfois surprenant que des disciplines aussi variées aient à utiliser des théorèmes communs, et le but initial de la théorie des graphes était de forger un outil mathématique s'appliquant tant aux sciences du comportement, à la théorie de l'information, aux jeux, aux réseaux de transport, qu'à la théorie des ensembles ou à d'autres domaines purement théoriques.

Chaque discipline va apporter en outre ses conventions pragmatiques de présentation d'un graphe, mieux adaptée à l'exploitation qu'elle en fait...

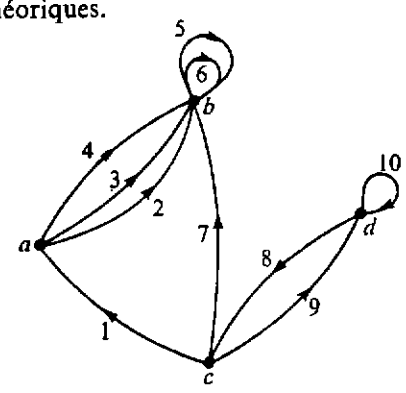


Fig. **1.1.** — Un 3-graphe d'ordre 4.

Un même graphe peut être décoré de multiples façons, plusieurs algorithmes de numérotation existent, etc. : une automatisation réaliste du tracé d'une figure comprend aussi ces aspects pragmatiques

[BERGE p 34]

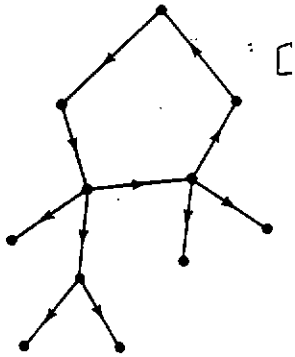


Fig. 3.4. — Graphe injectif.

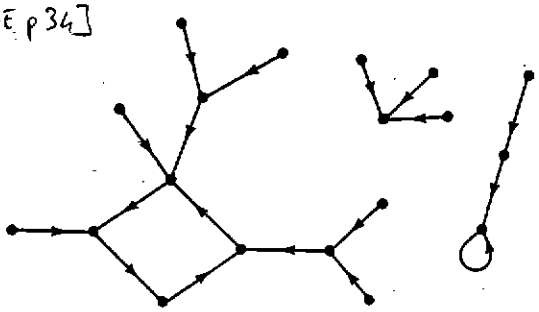


Fig. 3.5. — Graphe fonctionnel.

Le choix d'un exemple prototypique de graphe doit être suffisamment riche pour évoquer les possibles, mais pas trop afin d'être "lisible". Ce dilemme rejoint celui de l'usage des "..." dans une généralisation : $e^x = 1 + x + \frac{x^2}{2!} + \cdots$

[BERGE]

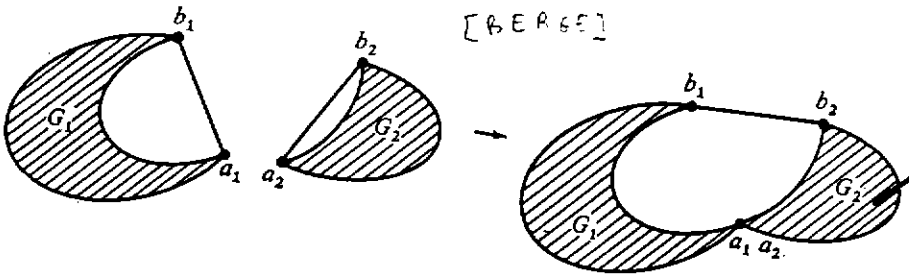


Fig. 15.10.

Lorsque certaines parties du graphe sont quelconques, une convention de langage permet de représenter une telle figure.

[BERGE]

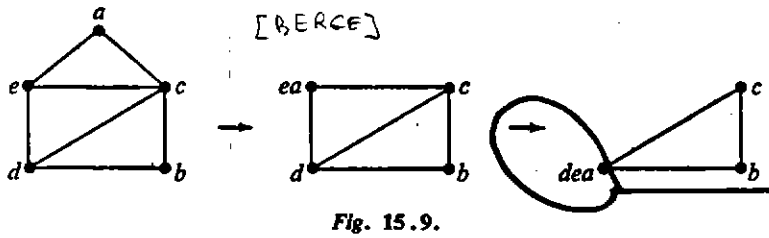


Fig. 15.9.

Les opérations de transformation de graphe viennent compliquer la pragmatique de présentation des figures

De nouvelles règles de présentation apparaissent

[BERGE]

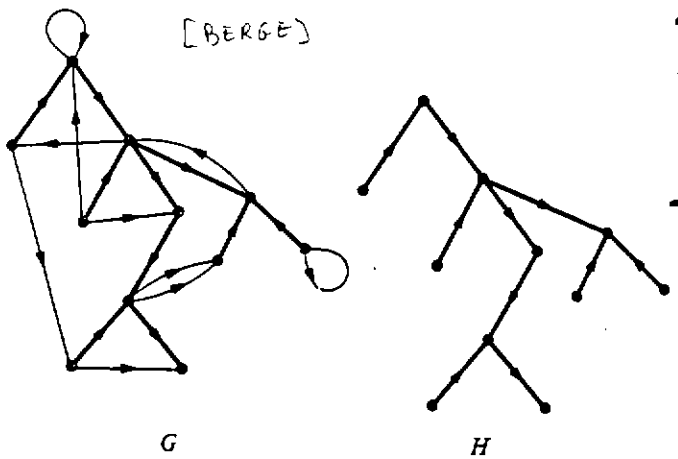


Fig. 3.1.

L'extraction du sous-graphe se fait en deux temps :
- marquage en gras des arêtes
- isolement du sous-graphe extrait

La reconnaissance visuelle est sollicitée pour vérifier que ces deux sous-graphes sont bien identiques.

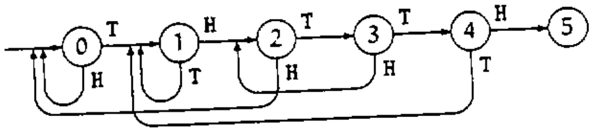
[Graham 89]

La diversité des besoins fait émerger de nombreux langages de Figures différents mais adaptés aux objets mathématiques à représenter.

Now let's try a more intricate experiment: We will flip coins until the pattern THTTH is first obtained. The sum of winning positions is now

$$S = \text{THTTH} + \text{HTHTTH} + \text{TTHTTH} + \text{HHTHTTH} + \text{HTTHTTH} + \text{THTHTTH} + \text{TTTHTTH} + \cdots;$$

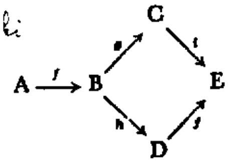
this sum is more difficult to describe than the previous one. If we go back to the method by which we solved the domino problems in Chapter 7, we can obtain a formula for S by considering it as a "finite state language" defined by the following "automaton":

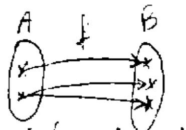


Un langage de figure adapté est plus explicite qu'un long discours

Au lieu de dire « soit *f* une application de A dans B », on emploiera souvent les phrases suivantes: « soit une application $f: A \to B$ » ou même « soit $f: A \to B$ ». Pour faciliter la lecture d'un raisonnement où interviennent plusieurs applications, on fera usage de *diagrammes* tels que

Th. des Ens. de Bourbaki p. E II.14





"pastiche scolaire"

où un groupe de signes tel que $A \xrightarrow{f} B$ doit s'interpréter comme signifiant que *f* est une application de A dans B.

th. des Ens de Bourbaki p. E IV.15

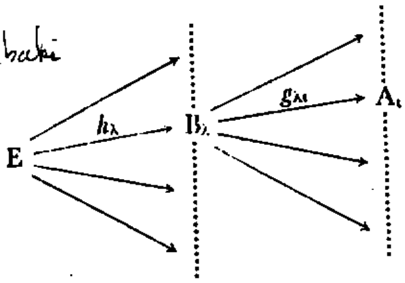
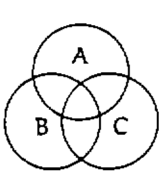


Fig. 1

5 A "Venn diagram" with three overlapping circles is often used to illustrate the eight possible subsets associated with three given sets:



[Graham 89, exo p.17]

Le diagramme de Venn, malgré ses imperfections, s'est imposé car il est utile dans [illegible]

Can the sixteen possibilities that arise with four given sets be illustrated by four overlapping circles?

$$a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cfrac{1}{a_4 + \cfrac{1}{a_5 + \cfrac{1}{a_6 + \cfrac{1}{a_7}}}}}}}$$

[Graham 89, p287]

La disposition spatiale des fractions continues, des divisions ou des preuves par [illegible] est assimilée à un langage de Figure particulier, présent[ant] un raisonnement qui lui est propre...

Reconnaissance des "formes symboliques"

Une des caractéristiques des Langages de Figures est d'exploiter plus que les autres les capacités de perception. C'est pourquoi, la connaissance des conventions et la reconnaissance des "Formes symboliques" sont deux éléments essentiels des Langages de Figures. Pour les produits cartésiens discrets, cette généralisation requise est matérialisée par les points de suspension ...

Ces "..." n'ont pas la même interprétation!

Cette convention de ne mettre qu'un seul zéro permet d'interpréter correctement les autres symboles

[Godement 66, p 438]

$$U = \begin{pmatrix} \lambda_1 & & & & 0 \\ & \ddots & & & \\ & & \lambda_1 & & \\ & & & \ddots & \\ & & & & \lambda_q \\ 0 & & & & & \ddots \\ & & & & & & \lambda_q \end{pmatrix}$$

avec s_i termes diagonaux égaux à λ_i; le polynôme caractéristique de U, i.e. de u, est donc

$$p_u(X) = (\lambda_1 - X)^{s_1} \dots (\lambda_q - X)^{s_q};$$

La matrice n'a pas besoin d'être complètement décrite pour être comprise.

l'évidence de ce résultat traduit la clarté de la reconnaissance des formes et de leur interprétation

¶ 49. Soient E un espace vectoriel réel de dimension finie, f une forme bilinéaire symétrique définie positive sur E, et u un automorphisme de f. Montrer qu'il existe une base de E, orthonormale relativement à f, et par rapport à laquelle la matrice de u est de la forme

[Godement 66, p 646]

$$\begin{pmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & 1 & & & & \\ & & & -1 & & & \\ & & & & \ddots & & \\ & & & & & -1 & \\ & & & & & & \begin{matrix} \cos\varphi_1 & -\sin\varphi_1 \\ \sin\varphi_1 & \cos\varphi_1 \end{matrix} \\ & & & & & & & \ddots \\ & & & & & & & & \begin{matrix} \cos\varphi_h & -\sin\varphi_h \\ \sin\varphi_h & \cos\varphi_h \end{matrix} \end{pmatrix}$$

La reconnaissance des formes n'est pas toujours très simple lorsque les termes se compliquent...

où les φ_i sont des nombres réels non multiples de π.

Exemple 2. Prenons pour X une matrice **triangulaire**, i.e. de la forme

La forme est explicitement utilisée dans les langages de Figures.

p 32-1

$$X = \begin{pmatrix} \alpha_{11} & \alpha_{21} & \alpha_{31} & \dots & \alpha_{n1} \\ 0 & \alpha_{22} & \alpha_{32} & \dots & \alpha_{n2} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & \alpha_{nn} \end{pmatrix};$$

Cette formule est un cas particulier de la suivante. Soient $n_1, \dots, n_p$ des entiers strictement positifs, et considérons une matrice de la forme

$$X = \begin{pmatrix} A_{11} & A_{21} & \dots & A_{p1} \\ 0 & A_{22} & \dots & A_{p2} \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & A_{pp} \end{pmatrix}$$

La forme décrite fait appel à l'abstraction du lecteur car elle n'est pas triangulaire lorsque les A_{ij} sont dépliés.

on a

[Godement 66, p449] $u_i = \lambda_i + v_i$

où v_i est un endomorphisme *nilpotent* de E_i; il suffit donc de montrer qu'il existe une base de E_i par rapport à laquelle la matrice de v_i est un tableau diagonal de matrices réduites; pour cela, il suffit d'appliquer le Théorème 3 à v_i et à E_i, en formant une matrice réduite toutes les fois qu'on a une « chaîne » de coefficients v_i égaux à 1. Par exemple, la matrice

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

s'écrit

$$\begin{pmatrix} U_1 & 0 & 0 \\ 0 & U_2 & 0 \\ 0 & 0 & U_3 \end{pmatrix}$$

avec

$$U_1 = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}, \quad U_2 = (0),$$

$$U_3 = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}.$$

Lorsque la généralisation possible est difficile ou trop nombreuse, des méthodes permettent au mathématicien d'introduire de bonnes abstractions pour la reconnaissance des formes.

Avant d'énoncer le résultat final de ce §, introduisons la définition suivante : on appelle **matrice réduite** (ou **matrice de Jordan**) à coefficients dans un corps commutatif K toute matrice carrée à coefficients dans K qui est de la forme

[Godement 66, p448]
$$\begin{pmatrix} \lambda & 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & \lambda & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & \lambda & 1 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & \dots & \lambda & 1 \\ 0 & 0 & 0 & 0 & \dots & 0 & \lambda \end{pmatrix}$$

autrement dit dont tous les termes diagonaux sont égaux, qui comporte des 1 immédiatement au-dessus de la diagonale, et des 0 partout ailleurs.

Par exemple,

$$(\lambda), \quad \begin{pmatrix} \lambda & 1 \\ 0 & \lambda \end{pmatrix}, \quad \begin{pmatrix} \lambda & 1 & 0 \\ 0 & \lambda & 1 \\ 0 & 0 & \lambda \end{pmatrix}$$

sont des matrices réduites d'ordres 1, 2, 3 respectivement.

Une définition de forme peut s'exprimer conjointement en langage de Figures ou en une description à l'aide d'un langage de Phrases (son interprétation littérale). La supériorité de l'usage de figures vient de la reconnaissance des formes...

p320
$$f(x_1, \dots, x_{n-1}) = \sum_{j=1}^{j=n} (-1)^{i+j} \alpha_j \cdot \begin{vmatrix} \xi_{11} & \dots & \xi_{n-1,1} \\ \dots & \dots & \dots \\ \xi_{1,j-1} & \dots & \xi_{n-1,j-1} \\ \xi_{1,j+1} & \dots & \xi_{n-1,j+1} \\ \dots & \dots & \dots \\ \xi_{1,n} & \dots & \xi_{n-1,n} \end{vmatrix}$$

Le contenu déforme les formes, puisque voici deux présentations d'un déterminant de même "forme"!

p320
$$f(x_1, \dots, x_{n-1}) = \begin{vmatrix} \xi_{11} & \dots & \xi_{i-1,1} & \alpha_1 & \xi_{i1} & \dots & \xi_{n-1,1} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \xi_{1n} & \dots & \xi_{i-1,n} & \alpha_n & \xi_{in} & \dots & \xi_{n-1,n} \end{vmatrix}$$

p550 $f'(y_1, \dots, y_p, x_1, \dots, x_q) = f[{}^t u(y_1), \dots, {}^t u(y_p), u^{-1}(x_1), \dots, u^{-1}(x_q)]$

La reconnaissance des Formes est aussi utilisée dans les langages de Figures (cette ligne s'apparente à la Figure ci-dessus)

[Graham 89]

6.5 BERNOULLI NUMBERS

The next important sequence of numbers on our agenda is named after Jakob Bernoulli (1654–1705), who discovered curious relationships while working out the formulas for sums of mth powers [22]. Let's write

$$S_m(n) = 0^m + 1^m + \cdots + (n-1)^m = \sum_{k=0}^{n-1} k^m = \sum_0^n x^m \, \delta x. \quad (6.77)$$

(Thus, when $m > 0$ we have $S_m(n) = H_{n-1}^{(-m)}$ in the notation of generalized harmonic numbers.) Bernoulli looked at the following sequence of formulas and spotted a pattern:

$$\begin{aligned}
S_0(n) &= n \\
S_1(n) &= \tfrac{1}{2}n^2 - \tfrac{1}{2}n \\
S_2(n) &= \tfrac{1}{3}n^3 - \tfrac{1}{2}n^2 + \tfrac{1}{6}n \\
S_3(n) &= \tfrac{1}{4}n^4 - \tfrac{1}{2}n^3 + \tfrac{1}{4}n^2 \\
S_4(n) &= \tfrac{1}{5}n^5 - \tfrac{1}{2}n^4 + \tfrac{1}{3}n^3 - \tfrac{1}{30}n \\
S_5(n) &= \tfrac{1}{6}n^6 - \tfrac{1}{2}n^5 + \tfrac{5}{12}n^4 - \tfrac{1}{12}n^2 \\
S_6(n) &= \tfrac{1}{7}n^7 - \tfrac{1}{2}n^6 + \tfrac{1}{2}n^5 - \tfrac{1}{6}n^3 + \tfrac{1}{42}n \\
S_7(n) &= \tfrac{1}{8}n^8 - \tfrac{1}{2}n^7 + \tfrac{7}{12}n^6 - \tfrac{7}{24}n^4 + \tfrac{1}{12}n^2 \\
S_8(n) &= \tfrac{1}{9}n^9 - \tfrac{1}{2}n^8 + \tfrac{2}{3}n^7 - \tfrac{7}{15}n^5 + \tfrac{2}{9}n^3 - \tfrac{1}{30}n \\
S_9(n) &= \tfrac{1}{10}n^{10} - \tfrac{1}{2}n^9 + \tfrac{3}{4}n^8 - \tfrac{7}{10}n^6 + \tfrac{1}{2}n^4 - \tfrac{3}{20}n^2 \\
S_{10}(n) &= \tfrac{1}{11}n^{11} - \tfrac{1}{2}n^{10} + \tfrac{5}{6}n^9 - n^7 + n^5 - \tfrac{1}{2}n^3 + \tfrac{5}{66}n
\end{aligned}$$

Can you see it too? The coefficient of n^{m+1} in $S_m(n)$ is always $1/(m+1)$. The coefficient of n^m is always $-1/2$. The coefficient of n^{m-1} is always ... let's see ... $m/12$. The coefficient of n^{m-2} is always zero. The coefficient of n^{m-3} is always ... let's see ... hmmm ... yes, it's $-m(m-1)(m-2)/720$. The coefficient of n^{m-4} is always zero. And it looks as if the pattern will continue, with the coefficient of n^{m-k} always being some constant times $m^{\underline{k}}$.

That was Bernoulli's discovery. In modern notation we write the coefficients in the form

$$\begin{aligned}
S_m(n) &= \frac{1}{m+1}\left(B_0 n^{m+1} + \binom{m+1}{1} B_1 n^m + \cdots + \binom{m+1}{m} B_m n\right) \\
&= \frac{1}{m+1}\sum_{k=0}^{m} \binom{m+1}{k} B_k n^{m+1-k}.
\end{aligned} \quad (6.78)$$

[Quinet 62, p104] $e^{-x^2} = 1 - \frac{x^2}{1!} + \frac{x^4}{2!} - \frac{x^6}{3!} + \cdots$

En intégrant terme à terme on obtient

$$I = x - \frac{x^3}{3} + \frac{1}{2!}\frac{x^5}{5} - \frac{1}{3!}\frac{x^7}{7} + \cdots$$

44. Intégrales de la forme $\int x^m (a + bx^n)^p dx$ *(avec m et n entiers, n positif et p quelconque).* [Quinet 62, p104]

On intègrera par parties, les calculs étant un peu longs si l'exposant m est élevé. On peut quelquefois aussi intégrer par substitution.

Exemple :

1° *Calculer* $\int \frac{x^3 dx}{\sqrt{1-x^2}}$ ou $\int x^3 (1-x^2)^{-\frac{1}{2}} dx.$

Les Langages de Formules utilisent une reconnaissance des formes à 1 dimension, ou à plusieurs dimensions par une disposition reconstituant un produit cartésien discret

L'exploitation de capacités de généralisation sont un facteur important de la créativité et de la découverte en mathématiques

La forme générale est préparée par la présentation adoptée, ici provenance, nombre et disposition des termes (pourquoi 4 termes, pourquoi pas $\frac{x^7}{3!7}$ ou $\frac{1}{7}\frac{x^3}{3}$)

Les "formes" sont alors utilisées pour décrire des familles de formules, le langage lui-même se prête alors à une mathématisation!

Ergonomie du Langage Mathématique : expressivité, homogénéité, mnémonique

definition where the factors go up and up:

$$x^{\overline{m}} = \overbrace{x(x+1)\dots(x+m-1)}^{m \text{ factors}}, \qquad \text{integer } m \geq 0. \tag{2.44}$$

When $m = 0$, we have $x^{\underline{0}} = x^{\overline{0}} = 1$, because a product of no factors is conventionally taken to be 1 (just as a sum of no terms is conventionally 0).

The quantity $x^{\underline{m}}$ is called "x to the m falling," if we have to read it aloud; similarly, $x^{\overline{m}}$ is "x to the m rising." These functions are also called *falling factorial powers* and *rising factorial powers*, since they are closely related to the factorial function $n! = n(n-1)\dots(1)$. In fact, $n! = n^{\underline{n}} = 1^{\overline{n}}$.

Several other notations for factorial powers appear in the mathematical literature, notably "Pochhammer's symbol" $(x)_m$ for $x^{\overline{m}}$ or $x^{\underline{m}}$; notations like $x^{(m)}$ or $x_{(m)}$ are also seen for $x^{\underline{m}}$. But the underline/overline convention is catching on, because it's easy to write, easy to remember, and free of redundant parentheses.

Mathematical terminology is sometimes crazy: Pochhammer [233] actually used the notation $(x)_m$ for the binomial coefficient $\binom{x}{m}$, not for factorial powers.

Falling powers $x^{\underline{m}}$ are especially nice with respect to Δ. We have

$$\begin{aligned}\Delta(x^{\underline{m}}) &= (x+1)^{\underline{m}} - x^{\underline{m}} \\ &= (x+1)x\dots(x-m+2) \;-\; x\dots(x-m+2)(x-m+1) \\ &= m\,x(x-1)\dots(x-m+2),\end{aligned}$$

hence the finite calculus has a handy law to match $D(x^m) = mx^{m-1}$:

$$\Delta(x^{\underline{m}}) = m x^{\underline{m-1}}. \tag{2.45}$$

Now we're almost ready for the punch line. Infinite calculus also has *definite* integrals: If $g(x) = Df(x)$, then

$$\int_a^b g(x)\,dx = f(x)\Big|_a^b = f(b) - f(a).$$

Therefore finite calculus — ever mimicking its more famous cousin — has definite *sums*: If $g(x) = \Delta f(x)$, then

$$\sum_a^b g(x)\,\delta x = f(x)\Big|_a^b = f(b) - f(a). \tag{2.47}$$

$$= \frac{1}{32}\int \left[e^{5jx} + 5e^{3jx} + 10e^{jx} + 10e^{-jx} + 5e^{-3jx} + e^{-5jx} \right] dx.$$

[Cousinet 62, p98]

groupons les termes équidistants du milieu

l'expressivité consiste ici en l'explicitation d'une opération sur les termes

L'activité mathématique procède par groupement, classification et association. Aussi, le langage ou des procédés métalangagiers doivent faciliter le déroulement de cette activité

¶ 7. Démontrer les relations suivantes : [Godement 66, p514]

$$2^{2n}\cos^{2n}t = 2\cos 2nt + 2\binom{2n}{1}\cos(2n-2)t + \dots + 2\binom{2n}{n-1}\cos 2t + \binom{2n}{n},$$

$$2^{2n}\cos^{2n+1}t = \cos(2n+1)t + \binom{2n+1}{1}\cos(2n-1)t + \dots + \binom{2n+1}{n}\cos t,$$

$$2^{2n}\sin^{2n}t = 2\sum_{k=0}^{k=n-1}(-1)^{n+k}\binom{k}{2n}\cos 2(n-k)t + \binom{n}{2n},$$

la présentation avec ... est meilleure pour la compréhension, mais celle avec Σ favorise les opérations de calcul. L'expressivité dépend de l'objectif courant

[Graham 89, p24]

People are often tempted to write

$$\sum_{k=2}^{n-1} k(k-1)(n-k) \qquad \text{instead of} \qquad \sum_{k=0}^{n} k(k-1)(n-k)$$

because the terms for $k = 0$, 1, and n in this sum are zero. Somehow it seems more efficient to add up $n-2$ terms instead of $n+1$ terms. But such temptations should be resisted; efficiency of computation is not the same as efficiency of understanding! We will find it advantageous to keep upper and lower bounds on an index of summation as simple as possible, because sums can be manipulated much more easily when the bounds are simple. Indeed, the form $\sum_{k=2}^{n-1}$ can even be dangerously ambiguous, because its meaning is not at all clear when $n = 0$ or $n = 1$ (see exercise 1). Zero-valued terms cause no harm, and they often save a lot of trouble.

homogénéité par simplicité et uniformité des bornes

[Graham 89, p251]

$$\left\{ {n+1 \atop m+1} \right\} = \sum_k \binom{n}{k} \left\{ {k \atop m} \right\}. \tag{6.15}$$

$$\left[{n+1 \atop m+1} \right] = \sum_k \left[{n \atop k} \right] \binom{k}{m}. \tag{6.16}$$

homogénéité par similarité

[Graham 89, p34]

$$\sum_j \sum_k a_{j,k}(P(j,k)) = \sum_{P(j,k)} a_{j,k} = \sum_k \sum_j a_{j,k}(P(j,k)). \tag{2.27}$$

homogénéité par simplification des règles de calcul

[Graham 89, p216]

5.6 HYPERGEOMETRIC TRANSFORMATIONS

It should be clear by now that a database of known hypergeometric closed forms is a useful tool for doing sums of binomial coefficients. We simply convert any given sum into its canonical hypergeometric form, then look it up in the table. If it's there, fine, we've got the answer. If not, we can add it to the database if the sum turns out to be expressible in closed form. We might also include entries in the table that say, "This sum does not have a simple closed form in general."

homogénéité par méthode de calcul telle la recherche d'une forme exacte (closed form)

Les Mathématiques sont confrontées à une autre forme d'ergonomie (déjà initiée par le calcul) qui vise une standardisation et une exploitation informatique. De tels outils ne font que rendre de nouveaux problèmes abordables... le défi majeur restant l'assistance à l'oralité

homogénéité par intégration des phases et formules

p179] $Q_n = R_{2^n} = \begin{cases} R_1 = 1, & \text{if } n = 0; \\ R_2 = 0, & \text{if } n \text{ is odd}; \\ R_4 = -1, & \text{if } n > 0 \text{ is even.} \end{cases}$

$$R_m = \begin{cases} 1 \\ 1 \\ 0 \\ -1 \\ -1 \\ 0 \end{cases} \quad \text{if } m \bmod 6 = \begin{cases} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{cases}.$$

découpage en cas et même factorisation des conditions de validité

9 Egyptian mathematicians in 1800 B.C. represented rational numbers between 0 and 1 as sums of unit fractions $1/x_1 + \cdots + 1/x_k$, where the x's were distinct positive integers. For example, they wrote $\frac{1}{3} + \frac{1}{15}$ instead of $\frac{2}{5}$. Prove that it is always possible to do this in a systematic way: If $0 < m/n < 1$, then [Graham 89, p94]

$$\frac{m}{n} = \frac{1}{q} + \left\{\text{representation of } \frac{m}{n} - \frac{1}{q}\right\}, \qquad q = \left\lceil \frac{n}{m} \right\rceil.$$

formules et phrases deviennent ici complètement mélangées.

(This is *Fibonacci's algorithm*, due to Leonardo Fibonacci, A.D. 1202.)

So far the notations we've been discussing are quite standard, but now we are about to make a radical departure from tradition. Kenneth Iverson introduced a wonderful idea in his programming language APL [161, page 11], and we'll see that it greatly simplifies many of the things we want to do in this book. The idea is simply to enclose a true-or-false statement in parentheses, and to say that the result is 1 if the statement is true, 0 if the statement is false. For example,

$$(p \text{ prime}) = \begin{cases} 1, & \text{if } p \text{ is a prime number;} \\ 0, & \text{if } p \text{ is not a prime number.} \end{cases}$$

Iverson's convention allows us to express sums with no constraints whatever on the index of summation, because we can rewrite (2.4) in the form

$$\sum_k a_k(P(k)). \qquad \text{[Graham 89, p24]} \tag{2.5}$$

Hey: The "Kronecker delta" that I've seen in other books (I mean δ_{kn}, which is 1 if $k = n$, 0 otherwise) is just a special case of Iverson's convention: We can write $(k = n)$ instead.

insertion des prédicats de validité dans la formule !

If $P(k)$ is false, the term $a_k(P(k))$ is zero, so we can safely include it among the terms being summed. This makes it easy to manipulate the index of summation, because we don't have to fuss with boundary conditions.

A slight technicality needs to be mentioned: Sometimes a_k isn't defined for all integers k. We get around this difficulty by assuming that $(P(k))$ is "very strongly zero" when $P(k)$ is false; it's so much zero, it makes $a_k(P(k))$ equal to zero even when a_k is undefined.

positive integers m and n, with $m \perp n$, we can find the position of m/n in the Stern–Brocot tree by "binary search" as follows: [Graham 89, p121]

```
S := I;
while m/n ≠ f(S) do
    if m/n < f(S) then (output(L); S := SL)
                  else (output(R); S := SR).
```

intégration des phrases et formule par formalisation de méthodes autrefois décrites en Phrases (c-à-d. extension de l'application des Langages de Formules)

Parcours et habillage d'une preuve

[Solow 82]

p8: **Example 1.** If the right triangle XYZ with sides of lengths x and y, and hypotenuse of length z, has an area of $z^2/4$, then the triangle XYZ is isosceles (see Figure 2).

ÉNONCÉ

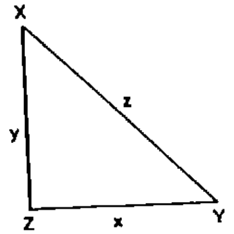


Fig. 2. The right triangle XYZ.

Finally, you should realize that, in general, it will not be practical to write down the entire thought process that goes into a proof, for this would require far too much time, effort, and space. Rather, a highly condensed version is usually presented and often makes little or no reference to the backward process. For the problem above it might go something like this.

PREUVES

Proof of Example 1. From the hypothesis and the formula for the area of a right triangle, the area of $XYZ = xy/2 = z^2/4$. By the Pythagorean theorem, $(x^2 + y^2) = z^2$, and on substituting $(x^2 + y^2)$ for z^2 and performing some algebraic manipulations one obtains $(x - y)^2 = 0$. Hence $x = y$ and the triangle XYZ is isosceles.// (The "//" or some equivalent symbol is usually used to indicate the end of the proof. Sometimes the letters Q.E.D. are used as they stand for the Latin words *quod erat demonstrandum*, meaning "which was to be demonstrated.")

Sometimes the shortened proof will be partly backward and partly forward. For example:

Proof of Example 1. The statement will be proved by establishing that $x = y$, which in turn is done by showing that $(x - y)^2 = (x^2 - 2xy + y^2) = 0$. But the area of the triangle is $(1/2)xy = (1/4)z^2$, so that $2xy = z^2$. By the Pythagorean theorem, $z^2 = (x^2 + y^2)$ and hence $(x^2 + y^2) = 2xy$, or $(x^2 - 2xy + y^2) = 0$, as required.//

Table 2. Proof of Example 1

Statement	Reason
A : Area of XYZ is $z^2/4$	Given
A1: $xy/2 = z^2/4$	Area = (base)(height)/2
A2: $x^2 + y^2 = z^2$	Pythagorean theorem
A3: $xy/2 = (x^2 + y^2)/4$	Substitute A2 into A1
A4: $x^2 - 2xy + y^2 = 0$	Algebra
A5: $(x - y)^2 = 0$	Factoring A4
B2: $(x - y) = 0$	Take square root in A5
B1: $x = y$	Add y to both sides of B2
B : XYZ is isosceles	Since B1 is true

The proof can also be written entirely from the backward process. Although this version is slightly unnatural, nonetheless, it is worth seeing.

Proof of Example 1. To reach the conclusion, it will be shown that $x = y$ by verifying that $(x - y)^2 = (x^2 - 2xy + y^2) = 0$, or equivalently, that $(x^2 + y^2) = 2xy$. This can be established by showing that $2xy = z^2$, for the Pythagorean theorem states that $(x^2 + y^2) = z^2$. In order to see that $2xy = z^2$, or equivalently, that $(1/2)xy = (1/4)z^2$, note that $(1/2)xy$ is the area of the triangle and it is equal to $(1/4)z^2$ by hypothesis, thus completing the proof.//

Proofs found in research articles are often very condensed, giving little more than a hint of how to do the proof. For example:

Proof of Example 1. The hypothesis together with the Pythagorean theorem yield $(x^2 + y^2) = 2xy$, hence $(x - y)^2 = 0$. Thus the triangle is isosceles as required.//

Problème résolu par transformation explicite de problème

6. *Polynômes à coefficients dans un anneau d'intégrité* [Godement 66, p360]

Nous allons établir le résultat suivant :

Les transformations de problème ici utilisées restent à modéliser

THÉORÈME 1. *Soit* K *un anneau d'intégrité commutatif; alors, pour tout entier* n, *l'anneau* $K[X_1, \ldots, X_n]$ *est intègre. En outre, quels que soient* $f, g \in K[X_1, \ldots, X_n]$, *on a*

$$d^o(fg) = d^o(f) + d^o(g).$$

Pour montrer que $K[X_1, \ldots, X_n]$ est un anneau d'intégrité, on tient compte de la relation

$$K[X_1, \ldots, X_n] = K[X_1, \ldots, X_{n-1}][X_n];$$

celle-ci permet, en raisonnant par récurrence (*) sur n, de se ramener au cas où $n = 1$.

Soient alors

$$f = a_0 + \cdots + a_p X^p, \qquad a_p \neq 0$$
$$g = b_0 + \cdots + b_q X^q, \qquad b_q \neq 0$$

deux polynômes non nuls à une variable, à coefficients dans K. Il est clair que fg contient un seul terme de degré $p + q$, à savoir

$$a_p b_q X^{p+q};$$

(*) Il est fréquent de procéder ainsi dans la théorie des polynômes à plusieurs indéterminées — mais on ne peut le faire que si l'on admet des polynômes à coefficients dans un *anneau* quelconque, car, même si K est un corps, l'anneau $K[X_1, \ldots, X_{n-1}]$ *n'est pas* un corps.

comme K est intègre, on a $a_p b_q \neq 0$, et par suite $fg \neq 0$. Ceci établit la première assertion du Théorème.

Pour établir la seconde (dans le cas général de plusieurs variables) on écrit

$$f = f_0 + \cdots + f_p, \qquad f_p \neq 0$$
$$g = g_0 + \cdots + g_q, \qquad g_q \neq 0;$$

il est clair que la partie homogène de degré total $p + q$ de fg est $f_p g_q$; or comme on sait déjà que $K[X_1, \ldots, X_n]$ est un anneau d'intégrité, on a $f_p g_q \neq 0$; donc f est de degré total $p + q$, ce qui achève la démonstration.

De façon générale, la plupart des démonstrations de livres mathématiques ne sont pas directement modélisables à l'heure actuelle faute d'analyses suffisantes...

Remarque 4. A vrai dire nous n'avons démontré la relation

$$d^o(fg) = d^o(f) + d^o(g)$$

que dans le cas où f et g sont non nuls. Si $f = 0$, cette relation s'écrit $-\infty = -\infty + d^o(g)$, et résulte des règles de calcul dont on a convenu de se servir au n° 2 en ce qui concerne le symbole $-\infty$.

Remarque 5. Il est clair que, si K n'est pas intègre, $K[X_1, \ldots, X_n]$ ne peut pas l'être non plus. En outre, dans ce cas, la relation

$$d^o(fg) = d^o(f) + d^o(g)$$

peut aussi être en défaut : choisir dans K deux éléments a, b non nuls tels que $ab = 0$, et prendre

$$f = aX, \qquad g = bX;$$

on a

$$d^o(fg) = -\infty, \qquad d^o(f) + d^o(g) = 2.$$

Exemple de stratégie de développement - identification

Theorem: Let **G** be a separable abelian locally compact group acting continuously on a compact space Ω and let σ be a 2-cocycle on **G**. Then we define on the space $C_c(G\times\Omega)$ a bilinear product $*$ and an involution * by the formulas:

(i) $a*b(x,\omega) = \int a(y,\omega)\ b(-y+x,-y.\omega)\ \sigma(y,-y+x)\ dy$

(ii) $a^*(x,\omega) = a(-x,-x.\omega)^*\ \sigma(x,-x)^*$

$C_c(G\times\Omega)$ is such endowed with the structure of involutive algebra.

Démonstration:

Let us show the associativity of the product:

$a*b(x,\omega) = \int a(y,\omega)\ b(-y+x,-y.\omega)\ \sigma(y,-y+x)\ dy$

$[(a*b)*c](x,\omega) = \int (a*b)(z,\omega)\ c(-z+x,-z.\omega)\ \sigma(z,-z+x)\ dz$

$[(a*b)*c](x,\omega) = \int\int a(y,\omega)\ b(-y+z,-y.\omega)\ \sigma(y,-y+z)\ c(-z+x,-z.\omega)\ \sigma(z,-z+x)\ dy\ dz$

$[(a*b)*c](x,\omega) = \int\int a(y,\omega)\ b(z,-y.\omega)\ \sigma(y,z)\ c(-z-y+x,-z-y.\omega)\ \sigma(y+z,-z-y+x)\ dy\ dz$

$[a*(b*c)](x,\omega) = \int a(y,\omega)\ (b*c)(-y+x,-y.\omega)\ \sigma(y,-y+x)\ dy$

$[a*(b*c)](x,\omega) = \iint a(y,\omega)\ b(z,-y.\omega)\ c(-z-y+x,-z-y.\omega)\ \sigma(z,-z-y+x)\ \sigma(y,-y+x)\ dz\ dy$

By applying the chain-rule, we get:

$\sigma(y,z)\ \sigma(y+z,-z-y+x) = \sigma(z,-z-y+x)\ \sigma(y,-y+x)$

which achieves the verification of the associativity.

[Graham 89] : Influence du contexte de définition pour l'obtention d'un résultat

p6 To evaluate S_n we can use a trick that Gauss reportedly came up with in 1786, when he was nine years old [73]:

$$\begin{array}{rcccccccc} S_n &=& 1 &+& 2 &+& 3 &+\cdots+& (n-1) + n \\ +S_n &=& n &+& (n-1) &+& (n-2) &+\cdots+& 2 + 1 \\ \hline 2S_n &=& (n+1) &+& (n+1) &+& (n+1) &+\cdots+& (n+1) + (n+1) \end{array}$$

We merely add S_n to its reversal, so that each of the n columns on the right sums to $n+1$. Simplifying,

$$S_n = \frac{n(n+1)}{2}, \qquad \text{for } n \geqslant 0. \tag{1.5}$$

Le résultat est l'objectif principal. Une représentation intuitive est utilisée pour sa découverte. Le procédé peut être généralisé à d'autres types de sommes.

---

p7 As experts, we might be satisfied with this derivation and consider it a proof, ~~even though we waved our hands a bit when doing the unfolding and reflecting~~. But students of mathematics should be able to meet stricter standards; so it's a good idea to construct a rigorous proof by induction.

La rigueur du résultat est l'objectif principal. Le contexte de définition exploite alors des outils comme une suite récurrente et les preuves par récurrence.

---

p30 Gauss's trick in Chapter 1 can be viewed as an application of these three basic laws. Suppose we want to compute the general sum of an *arithmetic progression*,

$$S = \sum_{0\leqslant k\leqslant n} (a+bk).$$

By the commutative law we can replace k by $n-k$, obtaining

$$S = \sum_{0\leqslant n-k\leqslant n} (a+b(n-k)) = \sum_{0\leqslant k\leqslant n} (a+bn-bk).$$

These two equations can be added by using the associative law:

$$2S = \sum_{0\leqslant k\leqslant n} \big((a+bk)+(a+bn-bk)\big) = \sum_{0\leqslant k\leqslant n} (2a+bn).$$

And we can now apply the distributive law and evaluate a trivial sum:

$$2S = (2a+bn)\sum_{0\leqslant k\leqslant n} 1 = (2a+bn)(n+1).$$

Dividing by 2, we have proved that

$$\sum_{k=0}^{n} (a+bk) = (a+\tfrac{1}{2}bn)(n+1). \tag{2.18}$$

Le Σ est introduit pour formaliser la [illegible] [illegible] preuve est [illegible] ce nouveau contexte de définition. L'auteur traite le cas le plus général qui préserve la preuve. L'automatisation de cette généralisation pose problème car il faut savoir construire une preuve pour pouvoir contrôler les possibilités.

---

p50 At this point a few of us are probably starting to wonder what all these parallels and analogies buy us. Well for one, definite summation gives us a simple way to compute sums of falling powers: The basic laws (2.45), (2.47), and (2.48) imply the general law

$$\sum_{0\leqslant k<n} k^{\underline{m}} = \frac{k^{\underline{m+1}}}{m+1}\bigg|_0^n = \frac{n^{\underline{m+1}}}{m+1}, \qquad \text{for integers } m, n \geqslant 0. \tag{2.50}$$

In particular, when $m = 1$ we have $k^{\underline{1}} = k$, so the principles of finite calculus give us an easy way to remember the fact that

$$\sum_{0\leqslant k<n} k = \frac{n^{\underline{2}}}{2} = n(n-1)/2.$$

Le résultat est maintenant un cas particulier d'abstractions plus élevées. Dans ce contexte de définition, $\sum_{0\leqslant k\leqslant n} (a^{k^{\underline{m}}} + b)$ serait une généralisation évidente sans intérêt. Dans ce cadre très formel, même (2.50) n'est qu'une utilisation de résultats (2.45) (2.47) et (2.48)

---

p161 Certain sums that we did in Chapters 1 and 2 were actually special cases of (5.10), or disguised versions of this identity. For example, the case $m = 1$ gives the sum of the nonnegative integers up through n:

$$\binom{0}{1}+\binom{1}{1}+\cdots+\binom{n}{1} = 0+1+\cdots+n = \frac{(n+1)n}{2} = \binom{n+1}{2}.$$

Ce contexte de définition traite des coefficients du binôme. Le résultat (5.10) est obtenu de façon intuitive à partir de $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$. Il est équivalent à (2.50) précédent.

NOTE ON A UNIVERSAL ANALYTICAL LANGUAGE.

(*Extract from a letter from Professor Mansion to Mr. Glaisher*).

.....PERMIT me to communicate to you an idea which I have long had in my mind concerning a sort of universal analytical language. I use in writing notations sufficiently expressive to enable me to dispense with all the text in many analytical questions.

Example. Let it be required to integrate

$$\frac{d^2u}{dx^2} - a^2 \frac{d^2u}{dy^2} = 0,$$

subject to the conditions

$$u = F(y), \quad \frac{du}{dx} = f(y), \text{ when } x = 0.$$

Put $$\frac{du}{dx} - a\frac{du}{dy} = \beta,$$

the equation then becomes

$$\frac{dv}{dx} + a\frac{dv}{dy} = 0,$$

whence $$v = \phi(y - ax),$$

and therefore $$\frac{du}{dx} - a\frac{du}{dy} = \phi(y - ax).$$

The integral of this last equation is

$$u = \psi(y + ax) + \chi(y - ax),$$

whence we have

$$\frac{du}{dx} = a\psi'(y + ax) - a\chi'(y - ax).$$

When $x = 0$, $$Fy = \psi y + \chi y,$$

$$fy = a\psi' y - a\chi' y.$$

The last equation gives

$$\frac{1}{a}\int fy\,dy = \psi y - \chi y,$$

whence $$\psi y = \tfrac{1}{2}Fy + \frac{1}{2a}\int fy\,dy,$$

$$\chi y = \tfrac{1}{2}Fy - \frac{1}{2a}\int fy\,dy,$$

&c., &c.

Translation:

$$\frac{d^2u}{dx^2} - a^2\frac{d^2u}{dy^2} = 0 \quad \text{..... (1),}$$

*$S(1)$ $$\frac{du}{dx} - a\frac{du}{dy} = v \quad \text{.....(2),}$$

$(1')$ $$\frac{dv}{dx} + a\frac{dv}{dy} = 0 \quad \text{.....(3),}$$

$\int(3)$ $$v = \phi(y - ax) \quad \text{..... (4),}$$

$(4) + (2)$ $$\frac{du}{dx} - a\frac{du}{dy} = \phi(y - ax) \quad \text{..... (5),}$$

$\int(5)$ $$u = \psi(y + ax) + \chi(y - ax) \quad \text{..... .... (6),}$$

$D_x(6)$ $$\frac{du}{dx} = a\psi'(y + ax) - a\chi'(y - ax) \quad \text{.....(7),}$$

$S(6)$ & (7) $x = 0$, $u = Fy$, $\frac{du}{dx} = fy$,

$(6')$ $$Fy = \psi y + \chi y \quad \text{.....(8),}$$

$(7')$ $$fy = a\psi' y - a\chi' y \quad \text{.....(9),}$$

$\frac{1}{a}\int(9)$ $$\frac{1}{a}\int fy\,dy = \psi y - \chi y \quad \text{.....(10),}$$

$\frac{(8)+(10)}{2}$ $$\psi y = \tfrac{1}{2}Fy + \frac{1}{2a}\int fy\,dy,$$

&c. &c."

Gauss certainly used analogous notations, I saw at Göttingen several of his MSS., and in particular that of his memoir on hypergeometrical series. There was no text at all, but only equations.....

PAUL MANSION.

Ghent, *February* 7, 1875.

---

* Substituendo in (1).

Présentation de démonstration proposée par Paul Depasse pour satisfaire ses besoins de clarté et de rigueur (1989)

Annexe: proofs of P3, P4 and T

The semantics of our proofs allows a syntax we think it makes them more readable:
Example:

P	A	>	(by M)
	B	=	(by N)
	C.		

is the (P-labelled) proof that A > C. This proof needs the results (above) proved by M and N. If a symbol such as ">" or "=" is not followed by "(by ...)" that is because the logical reason for which we wrote this symbol is obvious. Example:

P'	A'	>	(by M')
	3	>	
	2.		

Proof of P3:

	$2^{(x+2)}$	=		
	$4(2^x)$	$\geq$		
	$4(x+1)$	>		
	$4x+3$	$\geq$		
	$x+3$.			Hence:
	$2^{(x+2)}(y+1)-1$	>	$(x+3)(y+1)-1$	i.e, (by P1, P2)
P3.0	$F_2(x+1,\ y)$	>	$G_2(x+1,\ y)$	
P3.n	Let $\forall(x,\ y), F_{n+2}(x+1,\ y)$	>	$G_{n+2}(x+1,\ y)$.	Then:
	$F_{n+3}(0+1,\ 0)$	=		(by R1")
	$F_{n+2}(F_{n+2}(0,\ 0),\ F_{n+2}(0,\ 0))$	=		(by R3)
	$F_{n+2}(1,\ 1)$	>		(by P3.n)
	$G_{n+2}(1,\ 1)$	>		(by R9)
	$G_{n+2}(1,\ 0)$	=		(by R7)
	2	=		(by R7)
	$G_{n+3}(1,\ 0)$	=		
P3.n+1.0.0	Hence $F_{n+3}(0+1,\ 0)$	>		$G_{n+3}(0+1,\ 0)$
P3.n+1.0.y	Let $F_{n+3}(0+1,\ y)$	>	$G_{n+3}(0+1,\ y)$	Then:
	$F_{n+3}(1,\ y+1)$	=		(by R1")
	$F_{n+2}(F_{n+2}(y+1,\ y+1),\ F_{n+2}(y+1,\ y+1))$	>		(by P3.n, R5)
	$F_{n+2}(G_{n+2}(y+1,\ y+1),\ F_{n+2}(y+1,\ y+1))$	>		(by R5', R5)
	$F_{n+2}(G_{n+2}(y+1,\ y+1),\ y+1)$	>		(by R9", P3.n)
	$G_{n+2}(G_{n+2}(y+1,\ y+1),\ y+1)$	=		(by R10)
	$G_{n+3}(1,\ y+1)$			
P3.n+1.0.y+1	Hence $F_{n+3}(0+1,\ y+1)$	>	$G_{n+3}(0+1,\ y+1)$	
P3.n+1.0	Hence by I.S upon y, $F_{n+3}(0+1,\ \bullet)$	>	$G_{n+3}(0+1,\ \bullet)$.	
P3.n+1.x	Let $F_{n+3}(x+1,\ \bullet)$	>	$G_{n+3}(x+1,\ \bullet)$	Then:
	$F_{n+3}(x+2,\ y)$	=		(by D21)
	$f_{n+2}^{(x+3)}(y)$	=		(by D8)
	$f_{n+2}(f_{n+2}^{(x+2)}(y))$	=		(by D21)
	$f_{n+2}(F_{n+3}(x+1,\ y))$	=		(by R1')
	$F_{n+2}(F_{n+3}(x+1,\ y),\ F_{n+3}(x+1,\ y))$	>		(by P3.n+1.x, R5)
	$F_{n+2}(G_{n+3}(x+1,\ y),\ F_{n+3}(x+1,\ y))$	>		(by R5', R5)
	$F_{n+2}(G_{n+3}(x+1,\ y),\ y)$	>		(by R9", P3.n)
	$G_{n+2}(G_{n+3}(x+1,\ y),\ y)$	=		(by D24)
	$G_{n+3}(x+2,\ y)$			
P3.n+1	Hence by I.S upon x, $\forall(x,\ y) \in \mathbb{N}^2$, $F_{n+3}(x+1,\ y) > G_{n+3}(x+1,\ y)$.			
P3	Hence by I.S upon n, P3 is proved.			

Proof of P4:

P4.0	G_{0+1}	$>$	A_{0+1}	(by R8, D25)
P4.n	Let G_{n+1}	$>$	A_{n+1}, on $\mathbb{N}^2\backslash\{(0,\ 0)\}$. Then:	
	Prove, by induction upon $s \equiv x + y$, that G_{n+2}	$>$	A_{n+2} on $\mathbb{N}^2\backslash\{(0,\ 0)\}$.	
P4.n+1.1.0	$G_{n+2}(0,\ 1)$	$=$		(by D23)
	$G_{n+1}(1,\ 1)$	$>$		(by R9)
	$G_{n+1}(1,\ 0)$	$=$		(by R7)
	2	$>$		
	1	$\geq$		(by D26, D27)
	$A_{n+2}(0,\ 1)$			
P4.n+1.1.1	$G_{n+2}(1,\ 0)$	$=$		(by R7)
	2	$>$		
	0	$=$		(by R12)
	$A_{n+2}(0,\ 1)$			
P4.n+1.1	Hence, if $x + y = 1$, $G_{n+2}(x,\ y)$	$>$	$A_{n+2}(x,\ y)$.	
P4.n+1.s	Let, if $x + y \leq s$ $(s \geq 1)$, $G_{n+2}(x,\ y)$	$>$	$A_{n+2}(x,\ y)$.	Then:
P4.n+1.s+1.0	$G_{n+2}(0,\ s+1)$	$>$		(by R9)
	$G_{n+2}(0,\ 1)$	$>$		(by P4.n+1.1.0)
	2	$>$		
	1	$\geq$		(by D26, D27)
	$A_{n+2}(0,\ s+1)$			
P4.n+1.s+1.x	Let $x \in]0\ \ s]$.			Then:
	$G_{n+2}(x,\ s+1-x)$	$=$		(by D24)
	$G_{n+1}(G_{n+2}(x-1,\ s+1-x),\ s+1-x)$	$>$		(by P4.n+1.s, R9)
	$G_{n+1}(A_{n+2}(x-1,\ s+1-x),\ s+1-x)$	$>$		(by P4.n)
	$A_{n+1}(A_{n+2}(x-1,\ s+1-x),\ s+1-x)$	$=$		(by D28)
	$A_{n+2}(x,\ s+1-x)$			
P4.n+1.s+1.s+1	$G_{n+2}(s+1,\ 0)$	$>$		(by R13)
	$A_{n+2}(s+1,\ 0)$			
P4.n+1.s+1	Hence, if $x + y \leq s + 1$ $(s \geq 1)$, $G_{n+2}(x,\ y)$	$>$	$A_{n+2}(x,\ y)$.	
P4.n+1	Hence, by induction upon s, G_{n+2}	$>$	A_{n+2}, on $\mathbb{N}^2\backslash\{(0,\ 0)\}$.	
P4	Hence, by induction upon n, P4 is proved.			

Proof of T:

T.0	$f_{0+1}(x+1)$	$=$	(by R4')
	$2(x+1)+1$	$>$	
	$(x+1)+(x+1)$	$=$	(by D25)
	$A_1(x+1,\ x+1)$	$=$	(by D29)
	$a_1(x+1)$		
T.$\mathbb{N}^*$	$f_{n+2}(x+1)$	$=$	(by R1')
	$F_{n+2}(x+1,\ x+1)$	$>$	(by P3)
	$G_{n+2}(x+1,\ x+1)$	$>$	(by P4)
	$A_{n+2}(x+1,\ x+1)$	$=$	(by D29)
	$a_{n+2}(x+1)$		
T	Obvious		(by T.0, T.$\mathbb{N}^*$)

Preuve par récurrence (autre version)

Connaissances et opérations générales décrivant la mise en œuvre d'une preuve par récurrence

INDUCTION [Solow 82, p 53]

Connaissance à savoir exploiter et utiliser pour un cas simple de preuve par récurrence ou retenir le schéma de preuve de la section 2.4. avec P(1) au lieu de P(0)

To repeat, a proof by induction consists of two steps: I The first step is to verify that the statement P(1) is true. To do so, you simply replace n everywhere by 1. Usually, to verify that the resulting statement is true, you will only have to do some minor rewriting.

II The second step is much more challenging. It requires that you reach the conclusion that P(n+1) is true by using the assumption that P(n) is true. There is a very standard way of doing this. Begin by writing down the statement P(n+1). Since you are allowed to assume that P(n) is true and you want to conclude that P(n+1) is true, you should somehow try to rewrite the statement P(n+1) in terms of P(n)—as will be illustrated in a moment—for then you will be able to make use of the assumption that P(n) is true. On establishing that P(n+1) is true, the proof will be complete.

Example 6. For every integer $n \geq 1$, $\sum_{k=1}^{n} k = n(n+1)/2$ — Enoncé textuel

Outline of proof. When you are using the method of induction, it is helpful to write down the statement P(n), in this case: — procédé de désignation indispensable aux manipulations de transformation de problème

verifier $P(n)$: $\sum_{k=1}^{n} k = n(n+1)/2$ — Problème initial

I The first step in a proof by induction is to verify P(1). Replacing n everywhere by 1 in P(n), you obtain — Transformation de problème

verifier $P(1)$: $\sum_{k=1}^{1} k = 1(1+1)/2$ — Problème étape I

With a small amount of rewriting, it is easy to verify this statement since — Preuve étape I

$$\sum_{k=1}^{1} k = 1 = 1(1+1)/2$$

This step is often so easy that it is virtually omitted in the condensed proof by simply saying "The statement is clearly true for n = 1." — évaluation de difficulté et forme textuelle associée à une preuve triviale pour l'étape I

II The second step is more involved. You must use the assumption that P(n) is true to reach the conclusion that P(n+1) is true. The best way to proceed is to write down the statement P(n+1) by carefully replacing n with (n+1) everywhere in P(n), and rewriting a bit, if necessary. In this case — Problème étape II non instancié — transformation de problème

P(n+1): $\sum_{k=1}^{n+1} k = (n+1)[(n+1)+1]/2 = (n+1)(n+2)/2$

début de la preuve type II

To reach the conclusion that P(n+1) is true, begin with the left side of the equality in P(n+1) and try to make it look like the right side. In so doing, you should use the information in P(n) by relating the left side of the equality in P(n+1) to the left side of the equality in P(n), for then you will be able to use the right side of the equality in P(n). In this example,

Pour <But> lancer <tâche> en utilisant <relation> avec
- <tâche d'introduction>
- <tâche d'utilisation>
- aviser !

$$\sum_{k=1}^{n+1} k = \left(\sum_{k=1}^{n} k\right) + (n+1)$$

mise en route sur l'exemple

Now you can use the assumption that P(n) is true by replacing

$$\sum_{k=1}^{n} k$$

with n(n+1)/2, obtaining

$$\sum_{k=1}^{n+1} k = [n(n+1)/2] + (n+1)$$

All that remains is a bit of algebra to rewrite [n(n+1)/2 + (n+1)] as [(n+1)(n+2)/2], thus obtaining the right side of the equality in P(n+1). The algebraic steps are:

chercher à mettre sous forme N(N+1)/2

$$\begin{aligned} n(n+1)/2 + (n+1) &= (n^2 + n)/2 + 2(n+1)/2 \\ &= (n^2 + 3n + 2)/2 \\ &= (n+1)(n+2)/2 \end{aligned}$$

fin de preuve différente de la notre

In summary,

$$\begin{aligned} 1+\ldots+(n+1) &= (1+\ldots+n) + (n+1) \\ &= n(n+1)/2 + (n+1) \\ &= (n^2 + 3n + 2)/2 \\ &= (n+1)(n+2)/2 \end{aligned}$$

étape omise !

le récapitulatif fait disparaître le Σ et une étape.

Your ability to relate P(n+1) to P(n) so as to use the induction hypothesis that P(n) is true will determine the success of the proof by induction. If you are unable to relate P(n+1) to P(n), then you might wish to consider a different proof technique.

cheminement stratégique basé sur "l'étape clé" (critère d'importance)

Proof of Example 6. I The statement is clearly true for n = 1. II Assume that it is true for n (i.e., that 1+...+n = n(n+1)/2). Then

anticipation

$$\begin{aligned} 1+\ldots+(n+1) &= (1+\ldots+n) + (n+1) \\ &= n(n+1)/2 + \boxed{2(n+1)/2} \\ &= (n^2 + 3n + 2)/2 \text{ ou } = (n+2)(n+1)/2 \\ &= (n+1)(n+2)/2 \end{aligned}$$

démonstration finale reproductible seulement après reconstitution de la démarche précédente
- pourquoi n=1?
- où est le Σ ?
- qui est P(n)? etc.

which is P(n+1).//

La forme des résultats varie selon l'argument choisi

(p17) $I = \int \frac{du}{u} = \operatorname{Log} u = \operatorname{Log} \operatorname{tg} \frac{x}{2} + C.$ [Quinet 62]

1) Si on avait à calculer $\int \frac{dx}{\cos x}$, on l'écrirait

$$\int \frac{dx}{\cos x} = \int \frac{dx}{\sin\left(\frac{\pi}{2} - x\right)} = \int \frac{-dz}{\sin z} = -\operatorname{Log} \operatorname{tg} \frac{z}{2} = \boxed{-\operatorname{Log} \operatorname{tg}\left(\frac{\pi}{4} - \frac{x}{2}\right) + C.} \quad (1)$$

2) *Autre procédé pour calculer* $\int \frac{dx}{\cos x}$.

Multiplions le numérateur et le dénominateur par $\frac{1}{\cos x} + \operatorname{tg} x$:

$$\int \frac{\left(\frac{1}{\cos x} + \operatorname{tg} x\right)dx}{\cos x\left(\frac{1}{\cos x} + \operatorname{tg} x\right)} = \int \frac{\left(\frac{1}{\cos^2 x} + \frac{\sin x}{\cos^2 x}\right)}{\operatorname{tg} x + \frac{1}{\cos x}} dx,$$

et posons

$$\operatorname{tg} x + \frac{1}{\cos x} = u \qquad \text{d'où} \qquad du = \left(\frac{1}{\cos^2 x} + \frac{\sin x}{\cos^2 x}\right)dx.$$

L'intégrale devient

$$\int \frac{du}{u} = \operatorname{Log} u = \boxed{\operatorname{Log}\left(\operatorname{tg} x + \frac{1}{\cos x}\right) + C.} \quad (2)$$

2° *Calculer* $\int \frac{dx}{\cos x}$. (p88)

3) ***Premier procédé :*** on écrit

$$I = \int \frac{dx}{\sin\left(\frac{\pi}{2} - x\right)} = \text{ aussi à } \int \frac{dx}{\sin\left(\frac{\pi}{2} + x\right)} = \int \frac{du}{\sin u}$$

$$= \operatorname{Log} \operatorname{tg} \frac{u}{2} = \boxed{\operatorname{Log} \operatorname{tg}\left(\frac{\pi}{4} + \frac{x}{2}\right) + C.} \quad (3)$$

4) ***Deuxième procédé :*** on a

$$I = \int \frac{\frac{2dt}{1+t^2}}{\frac{1-t^2}{1+t^2}} = \int \frac{2dt}{1-t^2}.$$

Décomposons donc cette fraction rationnelle

$$\frac{2}{1-t^2} = \frac{2}{(1+t)(1-t)} \equiv \frac{A}{1+t} + \frac{B}{1-t}$$

d'où

$$2 \equiv A(1-t) + B(1+t)$$

pour	$t = 1$	il reste	$2 = 2B$	$B = 1$
pour	$t = -1$	»	$2 = 2A$	$A = 1.$

On aura

$$I = \int \frac{dt}{1+t} + \int \frac{dt}{1-t} = \operatorname{Log}(1+t) - \operatorname{Log}(1-t)$$

$$= \operatorname{Log} \frac{1+t}{1-t} = \boxed{\operatorname{Log} \frac{1 + \operatorname{tg} \frac{x}{2}}{1 - \operatorname{tg} \frac{x}{2}} + C.} \quad (4)$$

Seconde : argument résultat

La résolution d'un problème entraîne des manipulations de formules très variées sans fil directeur apparent : analyse d'un exemple.

Method 4: Replace sums by integrals.

p45 People who have been raised on calculus instead of discrete mathematics tend to be more familiar with $\int$ than with $\sum$, so they find it natural to try changing $\sum$ to $\int$. One of our goals in this book is to become so comfortable with $\sum$ that we'll think $\int$ is more difficult than $\sum$ (at least for exact results). But still, it's a good idea to explore the relation between $\sum$ and $\int$, since summation and integration are based on very similar ideas.

In calculus, an integral can be regarded as the area under a curve, and we can approximate this area by adding up the areas of long, skinny rectangles that touch the curve. We can also go the other way if a collection of long, skinny rectangles is given: Since $\square_n$ is the sum of the areas of rectangles whose sizes are 1×1, 1×4, ..., $1 \times n^2$, it is approximately equal to the area under the curve $f(x) = x^2$ between 0 and n.

Le remplacement de $\sum$ par $\int$ est une méthode explicitement définie ici.

Problem 5: An infinite sum.

We turn now to an asymptotic question posed by Solomon Golomb [122]: What is the approximate value of

$$S_n = \sum_{k\ge 1} \frac{1}{kN_n(k)^2} \quad \text{I} \tag{9.51}$$

where $N_n(k)$ is the number of digits required to write k in radix n notation?

3 façons d'écrire S_n :
- saisie exhaustive de l'information
- récupération de $\sum \frac{1}{k N_n(k)^2}$ dans un problème précédent
- réutilisation de $\sum \frac{1}{k^2}$ et insertion graphique de H_n

essai A

First let's try again for a ballpark estimate. The number of digits, $N_n(k)$, II is approximately $\log_n k = \log k / \log n$, III so the terms of this sum are roughly IV $(\log n)^2/k(\log k)^2$. Summing on k gives $\approx (\log n)^2 \sum_{k\ge 2} 1/k(\log k)^2$, V and this sum converges to a constant value because it can be compared to the integral

$$\int_2^\infty \frac{dx}{x(\ln x)^2} \text{ VII} = -\frac{1}{\ln x}\Big|_2^\infty \text{ VIII} = \frac{1}{\ln 2} \text{ IX}$$

Therefore we expect S_n to be about $C(\log n)^2$. X

encheînement et effectif essai A :
→ décomposition selon les raisons d'intérêt de la 1re approximation
- I écrire S_n
  - II définir $N_n(k)$
  - III l'approximer par $\log_n k$
  - IV déplier $\log_n(k)$
  - V remonter d'un cran pour obtenir $\frac{1}{k N_n(k)^2}$ zone d'intérêt
  - VI remonter à S_n
  - VII remplacer $\sum \to \int$
  - VIII une étape de calcul
  - IX calcul total
  - X conclusion essai A

essai B

Hand-wavy analyses like this are useful for orientation, but we need better estimates to solve the problem. One idea is to express $N_n(k)$ exactly:

$$N_n(k) = \lfloor \log_n k \rfloor + 1. \tag{9.52}$$

Thus, for example, k has three radix n digits when $n^2 \le k < n^3$, and this happens precisely when $\lfloor \log_n k \rfloor = 2$. It follows that $N_n(k) > \log_n k$, hence $S_n = \sum_{k\ge 1} 1/kN_n(k)^2 < 1 + (\log n)^2 \sum_{k\ge 2} 1/k(\log k)^2$.

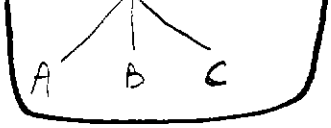
Proceeding as in Problem 1, we can try to write $N_n(k) = \log_n k + O(1)$ and substitute this into the formula for S_n. The term represented here by $O(1)$ is always between 0 and 1, and it is about $\frac{1}{2}$ on the average, so it seems rather well-behaved. But still, this isn't a good enough approximation to tell us about S_n; it gives us zero significant figures (that is, high relative error) when k is small, and these are the terms that contribute the most to the sum. We need a different idea.

essai C

The key (as in Problem 4) is to use our manipulative skills to put the sum into a more tractable form, before we resort to asymptotic estimates. We can introduce a new variable of summation, $m = N_n(k)$:

$$\begin{aligned} S_n &= \sum_{k,m\ge 1} \frac{(m = N_n(k))}{km^2} \\ &= \sum_{k,m\ge 1} \frac{(n^{m-1} \le k < n^m)}{km^2} \\ &= \sum_{m\ge 1} \frac{1}{m^2}\left(H_{n^m-1} - H_{n^{m-1}-1}\right). \end{aligned}$$

3 essais de résolution du problème



Session de saisie d'une formule

Touche appuyée	affichage sur l'écran
	_
s	s_
i	si_
n	sin(_)
^	$\sin^{-}(\,)$
2	$\sin^{2}-(\,)$
[return]	$\sin^{2}(\_)$
a	$\sin^{2}(a\_)$
+	$\sin^{2}(a+\_)$
b	$\sin^{2}(a+b\_)$
/	$\sin^{2}\left[a+\frac{b}{\_}\right]$
/	$\sin^{2}\left[\frac{a+b}{\_}\right]$
√	$\sin^{2}\left[\frac{a+b}{\sqrt{\_}}\right]$
1	$\sin^{2}\left[\frac{a+b}{\sqrt{1\_}}\right]$
-	$\sin^{2}\left[\frac{a+b}{\sqrt{1-\_}}\right]$
x	$\sin^{2}\left[\frac{a+b}{\sqrt{1-x\_}}\right]$
^	$\sin^{2}\left[\frac{a+b}{\sqrt{1-x^{-}}}\right]$
2	$\sin^{2}\left[\frac{a+b}{\sqrt{1-x^{2}\_}}\right]$
[return]	$\sin^{2}\left[\frac{a+b}{\sqrt{1-x^{2}}\_}\right]$
[return]	$\sin^{2}\left[\frac{a+b}{\sqrt{1-x^{2}}}-\right]$
[return]	$\sin^{2}\left[\frac{a+b}{\sqrt{1-x^{2}}}\right]-$
[return]	$\sin^{2}\left[\frac{a+b}{\sqrt{1-x^{2}}}\right]$

"Sin" est reconnu comme un symbole de fonction, et le système choisit automatiquement le parenthésage.

Une exception prend en compte l'élévation à la puissance pour un terme fonctionnel

La fraction bénéficie d'une mise en écriture infixée conventionnelle

Saisie conviviale complexe à programmer malgré un sous-ensemble très restreint et figé du langage Mathématique

Pour limiter la taille déjà importante du programme (1600 lignes de C), nous n'avons pas prévu de correction autre que l'effacement total de la formule. Il va de soi qu'un éditeur complet devrait contenir des possibilités de correction évoluées.

Code Flavors associé à la figure p 267

```
;;; -*- Mode:Common-Lisp; Base:10 -*-

;;caracteristiques physiques des boites
;;-------------------------------------

(defflavor essential-dims
   ((x-dim 0)
    (y-dim 0)
    (align 0))()
    (:documentation "represente les dimensions")
    :settable-instance-variables)

(defmethod (essential-dims :ss-align)()"partie sous align"(- y-dim align))

(defmethod (essential-dims :set-ss-align)(val)
   "positionne align de telle facon que ss-align soit egal a val"
   (send self :set-align (- y-dim val)))

;;caracteristiques arborescentes des boites
;;-----------------------------------------

(defflavor essential-arbo
   ((mere)
    (l-filles ())
    (x-positions())
    (y-positions()))()
    (:documentation "fait le lien avec les autres boites")
    :settable-instance-variables)

(defmethod (essential-arbo :x-dim-filles)()
   "renvoie la liste des x-dims des filles"
   (map-send l-filles :x-dim))

(defmethod (essential-arbo :y-dim-filles)()
   "renvoie la liste des y-dims des filles"
   (map-send l-filles :y-dim))

(defmethod (essential-arbo :align-filles)()
   "renvoie la liste des aligns des filles"
   (map-send l-filles :align))

(defmethod (essential-arbo :ss-align-filles)()
   "renvoie la liste des ss-aligns des filles"
   (map-send l-filles :ss-align))

;;gestion de l'affichage d'une arborescence
;;-----------------------------------------

(defflavor essential-affiche ()()
   (:documentation "gere l'affichage de l'arborescence des boites"))

(defmethod (essential-affiche :affiche)(x-top y-top &optional (cadre t)(fen *math-fenetre*))
   "suppose la compo ok. dessine le cadre si besoin et fait passer le message aux filles"
   (when cadre(send fen :draw-rect x-top y-top  x-dim y-dim))
   (prog-send l-filles :affiche (add-list x-positions x-top)
                               (add-list y-positions y-top)  (star cadre)(star fen)))

;;gestion de la composition d'une arborescence
;;--------------------------------------------

(defflavor essential-compose ()()
   (:documentation "gere la composition de la boite"))

(defmethod (essential-compose :compose)()
   "compo par defaut: propage jusqu'aux feuilles et instancie dims avec calcule-dims au retour"
   (map-send l-filles :compose)(send self :calcule-dims))

(defmethod (essential-compose :calcule-dims)()
   "cette methode est faite pour etre masquee par ses filles instanciables")

;;regroupement des fonctionnalites essentielles des boites
;;--------------------------------------------------------

(defflavor essential-boite()(essential-dims essential-arbo essential-affiche essential-compose)
  (:documentation "boite sans ses specificites de composition et d'affichage"))
```

```
;;specificite pour une composition horizontale des constituants
;;-------------------------------------------------------------

(defflavor boites-alignees-mixin
   ((espace (nth 4(multiple-value-list
                    (gwin:calculate-string-motion gwin:hl10-font " " 0 0))))())
    :settable-instance-variables
   (:documentation "complement indispensable pour instanciation")))

(defmethod (boites-alignees-mixin :calcule-dims)()
  "calcule les dimensions une fois celles de toutes les filles connues"
  (send self :aligne (send self :x-dim-filles)(send self :align-filles)(send self :ss-align-filles)))

(defmethod (boites-alignees-mixin :aligne)
           (l-long l-haut l-bas)
   "aligne des boites par rapport a la base et affecte les positions"
   (let* ((x-filles (add-dims l-long espace))
          (x-max (apply #'+ (send (car(last l-filles)) :x-dim)(last x-filles)))
          (haut-max (apply #'max l-haut))
          (bas-max (apply #'max l-bas))
          (y-filles (sub-list l-haut haut-max t)))
         (send self :set-x-dim x-max)
         (send self :set-y-dim (+ haut-max bas-max))
         (send self :set-align haut-max)
         (send self :set-x-positions x-filles)
         (send self :set-y-positions y-filles)))
;;boite a composition horizontale de ses constituants
;;--------------------------------------------------

(defflavor boites-alignees()(boites-alignees-mixin essential-boite)
  (:documentation "aligne toutes ses filles horizontalement"))

;;specificites pour une boite texte standard
;;-----------------------------------------

(defflavor boite-string-mixin ((string""))()
    :settable-instance-variables
   (:documentation "complement indispensable pour instanciation"))

(defmethod (boite-string-mixin :calcule-dims)()
   "instancie les dims cad align, x-dim et y-dim"
        (let ((dims(multiple-value-list
                 (gwin:calculate-string-motion gwin:hl10-font string 0 0))))
          (send self :set-x-dim (nth 4 dims))
          (send self :set-y-dim (nth 5 dims))
          (send self :set-align 9)))

(defmethod (boite-string-mixin :after :affiche)
           (x-top y-top &optional (cadre t)(fen *math-fenetre*))
  "affiche le string proprement dit en police fixee"
  (send fen :draw-string gwin:hl10-font string x-top  y-top ))

;; boite texte standard
;;--------------------

(defflavor boite-string ()(boite-string-mixin essential-boite)
    (:documentation "ceci est la boite representant le string"))

;;specificites pour une boite constituee d'un terme indice
;;--------------------------------------------------------

(defflavor boite-indicee-mixin()()
   (:documentation "complement indispensable pour instanciation"))

(defmethod (boite-indicee-mixin :before :set-l-filles)(liste)
   "verification de coherence du nombre d'arguments"
   (when (/=(length liste) 2)(format t "~&erreur dans la liste des 2 arguments : ~a" liste)))

(defmethod (boite-indicee-mixin :calcule-dims)()
   "instancie les dims cad align, x-dim et y-dim, et les positions des filles"
   (let*((boite-fantome (make-instance 'boites-alignees))
         (indice (cadr l-filles))
         (old-align-indice (send indice :align))
         (new-align-indice (floor (send indice :align) 2)))
      (send indice :set-align new-align-indice)
      (send boite-fantome :set-l-filles (send self :l-filles))
```

```
      (send boite-fantome :set-espace  2)
      (send boite-fantome :calcule-dims)
      (send self :set-x-dim (send boite-fantome :x-dim))
      (send self :set-y-dim (send boite-fantome :y-dim))
      (send self :set-align  (send boite-fantome :align))
      (send self :set-x-positions (send boite-fantome :x-positions))
      (send self :set-y-positions (send boite-fantome :y-positions))
      (send indice :set-align old-align-indice)))

;;boite constituee d'un terme indice
;;---------------------------------

(defflavor boite-indicee()(boite-indicee-mixin essential-boite)
    (:documentation "ceci est la boite dont la deuxieme fille est sommation"))

;;specificites pour une sommation
;;------------------------------

(defflavor boite-sommation-mixin((sigma (make-instance boite-sigma))(y-sigma 0))()
   (:documentation "complement indispensable pour instanciation")
   :settable-instance-variables)

(defmethod (boite-sommation-mixin :before :set-l-filles)(liste)
    "verification de coherence du nombre d'arguments"
   (when (/=(length liste) 3)(format t "~&erreur dans la liste des 3 arguments : ~a" liste)))


(defmethod (boite-sommation-mixin :calcule-dims)()
   "instancie les dims cad align, x-dim et y-dim, et les positions des filles"
   (let ((inf (nth 0 l-filles))
         (sup (nth 1 l-filles))
         (term (nth 2 l-filles))
         (bloc (make-instance 'boites-vert-centrees :espace 2))
         (somm (make-instance 'boites-alignees :espace 2)))
      (send sigma :set-mini)
      (send sigma :set-align (send term :align))
      (send bloc :set-l-filles (list sup sigma inf))
      (send bloc :calcule-dims)
      (send sigma :set-x-dim (send bloc :x-dim))
      (send bloc :set-align (+ (nth 1 (send bloc :y-positions))(send sigma :align)))
      (send somm :set-l-filles (list bloc term))
      (send somm :calcule-dims)
      (send self :set-x-dim (send somm :x-dim))
      (send self :set-y-dim (send somm :y-dim))
      (send self :set-align  (send somm :align))
      (let* ((x-bloc (send bloc :x-positions))
             (y-bloc (send bloc :y-positions))
             (x-somm (send somm :x-positions))
             (y-somm (send somm :y-positions))
             (x-sup (first x-bloc))
             (x-inf (third x-bloc))
             (x-term (second x-somm))
             (y-sup (+ (first y-bloc)(first y-somm)))
             (y-inf (+ (third y-bloc)(first y-somm)))
             (y-term (second y-somm)))
        (send self :set-y-sigma (+ (second y-bloc)(first y-somm)))
        (send self :set-x-positions (list x-inf x-sup x-term))
        (send self :set-y-positions (list y-inf y-sup y-term)))))

(defmethod (boite-sommation-mixin :before :affiche)
           (x-top y-top &optional (cadre t)(fen *math-fenetre*))
  "affiche le sigma en meme temps que le reste"
  (send sigma :affiche x-top (+ y-top y-sigma) cadre fen))

;;boite sommation
;;---------------

(defflavor boite-sommation()(boite-sommation-mixin essential-boite)
    (:documentation "ceci est la boite representant une somme sur une serie"))

;;specificites pour un caractere sigma variable en dimensions
;;-----------------------------------------------------------
(defflavor boite-sigma-mixin ()()
   (:documentation "complement indispensable pour instanciation"))

(defmethod (boite-sigma-mixin :set-mini)()
```

```
   "met le caractere sigma en position minimale"
   (send self :set-x-dim 20)(send self :set-y-dim 20)(send self :set-align 18))

(defmethod (boite-sigma-mixin :after :set-x-dim)(val)
   "verifie que 20 est valeur mini"
   (when (< val 20)(send self :set-x-dim 20)))

(defmethod (boite-sigma-mixin :after :set-align)(val)
   "verifie que 18 est valeur mini, et propage (val + 2) a y-dim"
   (send self :set-y-dim (+ 2 val))
   (when (< val 18)(send self :set-align 18)))

(defmethod (boite-sigma-mixin :after :affiche)
           (x-top y-top &optional (cadre t)(fen *math-fenetre*))
  "affiche le sigma proprement dit "
    (let  ((xg x-top)
           (yh (- y-top 2))
           (xd (+ x-top x-dim))
           (yb (-(+ y-top y-dim) 2))
           (xm (+ x-top (max 10 (floor x-dim 3))))
           (ym (+ y-top (floor y-dim 2))))
     (send fen :draw-polyline (vector xd xg xm xg xd)(vector yh yh ym yb yb))))

;;caractere sigma variable en dimensions
;;---------------------------------------

(defflavor boite-sigma ()(boite-sigma-mixin essential-boite)
    (:documentation "ceci est la boite representant le signe sigma"))

;;specificites pour un alignement vertical centre
;;-----------------------------------------------

(defflavor boites-vert-centrees-mixin
   ((espace (nth 5(multiple-value-list
                    (gwin:calculate-string-motion gwin:hl10-font " " 0 0)))))()
    :settable-instance-variables
   (:documentation "complement indispensable pour instanciation"))

(defmethod (boites-vert-centrees-mixin :calcule-dims)()
  "calcule les dimensions une fois celles de toutes les filles connues"
  (send self :vert-centre (send self :x-dim-filles)(send self :y-dim-filles)))

(defmethod (boites-vert-centrees-mixin :vert-centre)
           (l-long l-haut)
   "aligne des boites verticalement, les centre et s'aligne au milieu"
   (let* ((y-filles (add-dims l-haut espace))
          (y-max (apply #'+ (send (car(last l-filles)) :y-dim)(last y-filles)))
          (x-max (apply #'max l-long))
          (x-filles (mapcar #'(lambda(x)(floor (- x-max x)2)) l-long)));;;(sub-list l-long x-max t)))
         (send self :set-x-dim x-max)
         (send self :set-y-dim y-max)
         (send self :set-align (floor y-max 2))
         (send self :set-x-positions x-filles)
         (send self :set-y-positions y-filles)))

;;boite qui compose ses constituants verticalement (vecteur...)
;;-------------------------------------------------------------

(defflavor boites-vert-centrees()(boites-vert-centrees-mixin essential-boite)
  (:documentation "aligne toutes ses filles verticalement"))
```

Schéma général de traitement du langage (projet japonais de 5ème génération)

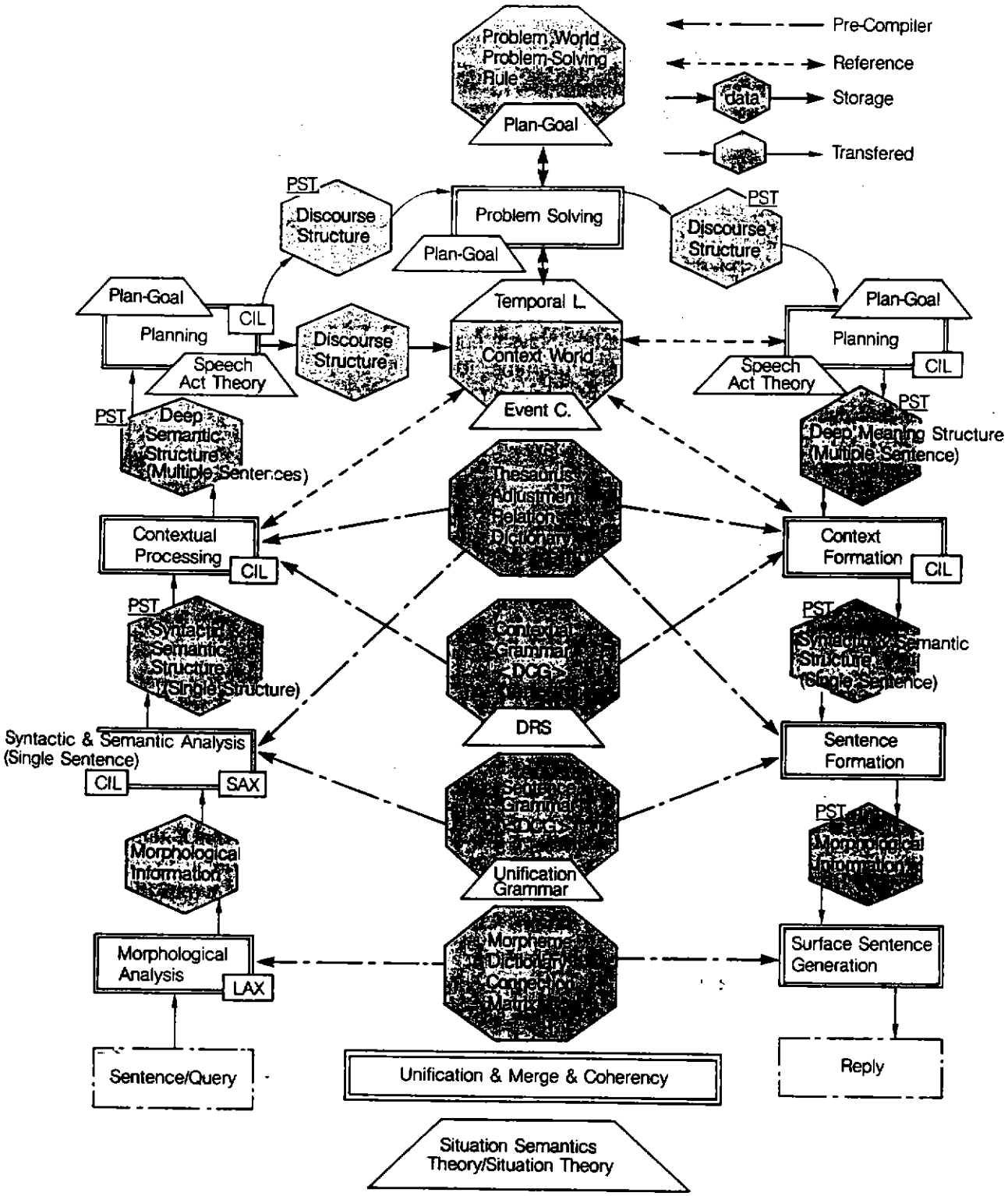


Fig.7 DUALS III Structure

A-33

Adaptation du langage par introduction de notations

Cette première page est directement consacrée aux notations du problème:

Computing Systems in Engineering Vol. 1, No. 1, pp. 63–73, 1990
Printed in Great Britain.

0956-0521/90 $3.00+0.00

THE USE OF MATRIX VISUALIZATION IN ALGORITHMIC DESIGN

M. W. BERRY

Center for Supercomputing Research and Development, University of Illinois at Urbana-Champaign, 305 Talbot Laboratory, 104 South Wright Street, Urbana, IL 61801-2932, U.S.A.

(*Received* 16 *January* 1990)

Abstract—The use of matrix visualization in the design and development of numerical algorithms for supercomputers is discussed. Using color computer graphics, numerical analysts can gain new insights into algorithm behavior, which can then be used to design more efficient (parallel) numerical algorithms. The application of a matrix visualization tool, MatVu, in the design of algorithms from numerical linear algebra is the primary focus. Specific examples include the derivation of optimal preconditioning matrices for a conjugate gradient method, the design of parallel hybrid algorithms for solving the symmetric eigenvalue problem, the effects of operator splitting in the solution of incompressible Navier–Stokes equations, and the monitoring of Jacobian matrices associated with the application of Newton's method to a corresponding nonlinear system of equations.

NOMENCLATURE

$\mathbf{A}$	dense symmetric positive definite matrix in Eq. (1)
A_k	kth iteration matrix in one-sided Jacobi algorithm
$\tilde{\mathbf{A}}$	symmetric positive definite submatrix in coefficient matrix of Eq. (7) which has $\tilde{n}$ identical blocks
$\mathbf{a}_i$	ith column of matrix A
$\mathbf{B}$	unsymmetric positive definite submatrix in coefficient matrix of Eq. (10)
$\mathbf{C}$	coefficient matrix for pressure equations in Eq. (8)
$\tilde{\mathbf{C}}$	coefficient matrix for preconditioned pressure equations in Eq. (9)
c	number of critical sweeps prior to numerical decoupling in hybrid Jacobi method ([Eq. (6)]
$\mathbf{D}_k$	diagonal matrix comprised of diagonal elements of $\mathbf{S}_k$ [Eq. (5)]
	upper-triangular matrix comprised of superdiagonal elements of $\mathbf{S}_k$ [Eq. (5)]
	external forces vector in Generalized Stokes problem
$\tilde{\mathbf{f}}$	right-hand-side vector in preconditioned pressure equations [Eq.(9)]
$\hat{\mathbf{f}}$	right-hand-side vector of nonlinear system in Eq. (11)
$\mathbf{f}_1$	first component of right-hand-side vector in Eq. (10)
$\mathbf{f}_2$	second component of right-hand-side vector in Eq. (10)
$\mathbf{J}$	Jacobian matrix of nonlinear system of equations in Eq. (11)
L	characteristic length (constant)
$\mathbf{L}$	Cholesky factor of matrix C
$\mathbf{M}$	Off-diagonal subblock in coefficient matrix of Eq. (7)
$\tilde{\mathbf{M}}$	off-diagonal subblock in coefficient matrix of Eq. (7)
m	grid dimension for finite element discretization of Generalized Stokes problem
m_i	order of $\mathbf{T}_i$ matrix
N	symmetric positive definite preconditioner for pressure equations in Eq. (8)
$\tilde{\mathbf{N}}$	nonlinear perturbation matrix, where $\mathbf{J}=\mathbf{B}+\tilde{\mathbf{N}}$
n	order of matrix A
$\tilde{n}$	order of matrix $\tilde{\mathbf{A}}$
$\tilde{n}$	number of identical diagonal blocks in matrix $\tilde{\mathbf{A}}$
$\|\cdot\|_2$	Euclidean vector norm
$\mathcal{O}$	order of magnitude
$\mathbf{P}$	pressure vector in incompressible Navier–Stokes equations
$\mathbf{P}_x$	first partial derivative of P with respect to x
$\mathbf{P}_y$	first partial derivatives of P with respect to y
$\mathbf{p}$	pressure vector in Generalized Stokes problem
$\tilde{\mathbf{p}}$	solution vector in preconditioned pressure equations [Eq. (9)]
p	number of diagonal blocks in $\mathbf{S}_c$ or number of eigenvalue clusters [Eq. (6)]
$\mathbf{Q}$	matrix–vector product AV in Eq. (2), matrix of unscaled eigenvectors of matrix A
$\mathbf{q}_i$	ith column of matrix Q in Eq. (2)
$\mathbf{q}_i$	ith column of matrix Q
$\mathbf{r}$	arbitrary $\tilde{n}\times 1$ vector
Re	Reynolds number ($\rho_\infty V_\infty L/\mu$)
$\mathbf{S}_c$	block diagonal matrix in Eq. (6)
$\mathbf{S}_k$	matrix product $\mathbf{A}_k^T\mathbf{A}_k$ in Eq. (5)
$\mathbf{s}$	solution vector of nonlinear system in Eq. (11)
$\mathbf{T}_i$	ith subblock of matrix $\mathbf{S}_c$ (Eq. (6)
t	time variable in incompressible Navier–Stokes equations
Δt	length of time interval for *theta* scheme
$\mathbf{u}$	velocity vector in Generalized Stokes problem
$\tilde{\mathbf{u}}$	velocity component in incompressible Navier–Stokes equations
$\tilde{\mathbf{u}}_t$	first partial derivative of $\tilde{\mathbf{u}}$ with respect to t
$\mathbf{V}$	orthogonal matrix constructed from plane rotations in one-sided Jacobi method [Eq. (2)]
V_∞	free stream velocity (constant)
$\tilde{\mathbf{V}}$	velocity vector in incompressible Navier–Stokes equations
$\mathbf{v}$	kinematic velocity vector in Generalized Stokes problem
$\tilde{v}$	velocity component in incompressible Navier–Stokes equations
$\tilde{v}_t$	first partial derivative of $\tilde{v}$ with respect to t
$\mathbf{x}$	eigenvector of matrix A [Eq. (1)]
x	component of $\tilde{\mathbf{u}}$ and $\tilde{v}$
y	component of $\tilde{\mathbf{u}}$ and $\tilde{v}$
$\mathbf{z}$	matrix–vector product $\mathbf{N}^{-1}\mathbf{r}$
$\mathbf{0}$	zero matrix (vector)
α	constant in Generalized Stokes problem
Δ	Laplacian operator ($\partial^2/\partial x^2+\partial^2/\partial y^2$)
δ_{ij}	Kronecker delta
ϵ	user-supplied tolerance (constant)
λ	eigenvalue of matrix A [Eq. (1)]

A-34

List of Symbols

La diversité d'emploi des notations requiert la présence de tables de notations dans les ouvrages scientifiques tels que ce livre d'informatique théorique.

$\varnothing$, 3
$\subseteq$, 3
$\subset$, 3
$P(S)$, 3
$S \backslash T$, 3
$S - T$, 3
$S_1 \times \ldots \times S_n$, 3
S^n, 3
S^{Nat}, 3
Nat, 3
Int, 3
Bool, 3
$R(A)$, 4
$R^{-1}(A)$, 4
R^*, 4
$f: S \rightharpoonup T$, 4
$f: S \to T$, 4
$f \upharpoonright A$, 4
$f[s/t]$, 5
$g \circ f$, 5
f^i, 5

Σ_f, Σ, 21
$\models_f w$, 22, 150
$Th(\mathscr{I})$, 22
$\models w$, 22, 150
w_x^t, 22, 25
$W \models w$, 25, 150
$Cn(w)$, 25
$\bar{w}$, 25
$\vdash$, 28
$\mathscr{L}_1$, L_1^B, 41, 42
$\overset{S}{\Rightarrow}$, $\Rightarrow$, 44, 48
$\mathscr{M}_s(S)$, $\mathscr{M}(S)$, 45, 49, 67, 101, 107
$\mathscr{L}_2$, L_2^B, 47
$\mathscr{L}_3$, 51
ω, D_ω, 54
Γ_s, Γ, 58
$\vec{r}$, $\overset{*}{\vec{r}}$, 61
$\mathscr{S}$, 63
$\bar{r}$, 65

(Ci-contre une moitié de la page ix consacrée aux notations dans le livre de J. Loeckx et K. Sieber : The foundations of program verification, Wiley & Son ed.)

List of Special Symbols

(extrait de : computability theory, an introduction par N.D. Jones, Academic Press)

	Symbol	Meaning	First Reference
53.	$ID(\alpha, \#)$	true iff α is an encoded I.D. with # as separator	75
54.	$Yield(z, \alpha, \beta)$	true iff z, α, and β are encoded forms of a Turing machine Z and I.D.s α' and β', for which $\alpha' \vdash_Z \beta'$ is true	75
55.	$T_n(z, \vec{x}_n, y)$	true iff y is the encoded form of a halting computation by a Turing machine with code z, which was given $x_1, \ldots, x_n$ on its initial tape	75
56.	$U(y) = w$	iff y is an encoded computation, and w is the content of the final I.D.	75
57.	$C_R^p(\vec{x}_n)$	the partial characteristic function of predicate $R(\vec{x}_n)$	92
58.	$C_R^t(\vec{x}_n)$	the total characteristic function of predicate $R(\vec{x}_n)$	92
59.	U_n	the universal Turing machine for $n$ary functions	108
60.	$+, \dashv, \approx, c$	formal symbols representing (,), =, and the comma, respectively	131
61.	$x \to y$	a semi-Thue production	140
62.	$S = (A, B, \Pi, \alpha)$	a semi-Thue system	140
63.	$G(S)$	the set generated by S	141
64.	$P = (V, A, B, \Pi, \alpha)$	a Post canonical system	146
65.	$G(P)$	the set generated by P	146

Les notations s'adaptent à tout ce qu'elles ont besoin de représenter, y compris d'autres notations ...

A-35

A Note on Notation

SOME OF THE SYMBOLISM in this book has not (yet?) become standard. Here is a list of notations that might be unfamiliar to readers who have learned similar material from other books, together with the page numbers where these notations are explained:

Notation	*Name*	*Page*
$\ln x$	natural logarithm: $\log_e x$	262
$\lg x$	binary logarithm: $\log_2 x$	70
$\log x$	common logarithm: $\log_{10} x$	435
$\lfloor x \rfloor$	floor: $\max\{n \mid n \leqslant x,\ \text{integer } n\}$	67
$\lceil x \rceil$	ceiling: $\min\{n \mid n \geqslant x,\ \text{integer } n\}$	67
$x \bmod y$	remainder: $x - y\lfloor x/y \rfloor$	82
$\{x\}$	fractional part: $x \bmod 1$	70
$\sum f(x)\,\delta x$	indefinite summation	48
$\sum_a^b f(x)\,\delta x$	definite summation	49
$x^{\underline{n}}$	falling factorial power: $x!/(x-n)!$	47
$x^{\overline{n}}$	rising factorial power: $\Gamma(x+n)/\Gamma(x)$	48
$n¡$	subfactorial: $n!/0! - n!/1! + \cdots + (-1)^n n!/n!$	194
$\Re z$	real part: x, if $z = x + iy$	64
$\Im z$	imaginary part: y, if $z = x + iy$	64
H_n	harmonic number: $1/1 + \cdots + 1/n$	29
$H_n^{(x)}$	generalized harmonic number: $1/1^x + \cdots + 1/n^x$	263
$f^{(m)}(z)$	mth derivative of f at z	456

Prestressed concrete mathematics is concrete mathematics that's preceded by a bewildering list of notations.

Also 'nonstring' is a string.

If you don't understand what the x denotes at the bottom of this page, try asking your Latin professor instead of your math professor.

Les notations mathématiques évoluent et laissent encore une part de créativité dans leur adaptation (cf par ex. $x^{\overline{n}}$ ou la note *)

$\left[{n \atop m}\right]$	Stirling cycle number (the "first kind")	245
$\left\{{n \atop m}\right\}$	Stirling subset number (the "second kind")	244
$\left\langle{n \atop m}\right\rangle$	Eulerian number	253
$\left\langle\!\left\langle{n \atop m}\right\rangle\!\right\rangle$	Second-order Eulerian number	256
$(a_m \ldots a_0)_b$	radix notation for $\sum_{k=0}^m a_k b^k$	11
$K(a_1, \ldots, a_n)$	continuant polynomial	288
$F\left({a, b \atop c} \middle\vert z\right)$	hypergeometric function	205
$\#A$	cardinality: number of elements in the set A	39
$[z^n]\,f(z)$	coefficient of z^n in $f(z)$	197
$[m = n]$	1 if $m = n$, otherwise 0*	24
$[m \backslash n]$	1 if m divides n, otherwise 0*	102
$[m \backslash\backslash n]$	1 if m exactly divides n, otherwise 0*	146
$[m \perp n]$	1 if m is relatively prime to n, otherwise 0*	115

*In general, if S is any statement that can be true or false, the parenthesized notation $[S]$ stands for 1 if S is true, 0 otherwise.

Throughout this text, we use single-quote marks ('...') to delimit text as it is *written*, double-quote marks ("...") for a phrase as it is *spoken*. Thus, the string of letters 'string' is sometimes called a "string."

An expression of the form 'a/bc' means the same as '$a/(bc)$'. Moreover, $\log x/\log y = (\log x)/(\log y)$ and $2n! = 2(n!)$.

A-36

La nécessaire interprétation d'une notation dans un autre système de représentation est particulièrement mise sur cet exemple de physique (axis, link, direction, etc). Ici, la résolution du problème implique la maîtrise de cette autre représentation.

Table 1. Summary of Notation

- O_i: coordinate system (x_i, y_i, z_i) attached to the ith link; where z_i is the axis of rotation if the $(i+1)$st joint is rotational, or z_i is in the direction of motion if the $(i+1)$st joint is translational; further, x_i and y_i are assigned such that $z_i = x_i \times y_i$.
- A_j^i: pure rotational matrix mapping vectors in O_i into vectors in O_j.
- q_i: joint generalized variable for the ith joint.
- $r_i = A_1^0 P_i$, where P_i is the position vector of the origin of O_i.
- $r_i^* = A_1^0 P_i^*$, where P_i^* is the position vector of the mass center of the ith link.
- $s_i = r_i^* - r_i$.
- $p_i = r_i - A_i^{i-1} r_{i-1}$.
- m_i: mass of the ith link.
- $F_i = A_1^0 \bar{F}_i$, where $\bar{F}_i$ is the total external vector force exerted on the ith link.
- $N_i = A_1^0 \bar{N}_i$, where $\bar{N}_i$ is the total external vector moment exerted on the ith link.
- $f_i = A_1^0 \bar{f}_i$, where $\bar{f}_i$ is the vector force exerted on the ith link by the $(i-1)$st link.
- $n_i = A_1^0 \bar{n}_i$, where $\bar{n}_i$ is the vector moment exerted on the ith link by the $(i-1)$st link.
- $J_i = A_1^0 \bar{J}_i A_0^i$, where $\bar{J}_i$ is the inertia matrix of the ith link about its mass center.
- $\omega^i = A_1^0 \bar{\omega}^i$, where $\bar{\omega}^i$ is the angular velocity of the ith link.
- $\alpha^i = A_1^0 \bar{\alpha}^i$, where $\bar{\alpha}^i$ is the angular acceleration vector of the ith link.
- $v^i = A_1^0 \bar{v}^i$, where $\bar{v}^i$ is the linear velocity of the origin O_i.
- $a^i = A_1^0 \bar{a}^i$, where $\bar{a}^i$ is the linear acceleration of the origin O_i.
- $v^{i*} = A_1^0 \bar{v}^{i*}$, where $\bar{v}^{i*}$ is the linear velocity of the mass center of the ith link.
- $a^{i*} = A_1^0 \bar{a}^{i*}$, where $\bar{a}^{i*}$ is the linear acceleration of the mass center of the ith link.
- $G_i = gA_1^0 z_0$, where g is the gravitational acceleration.
- $g_i = A_0^i \bar{g}_i$, where $\bar{g}_i$ is the gravitational force exerted on the ith link.
- τ_i: joint generalized actuator torque at the ith joint.
- $\delta_i = \begin{cases} 1 & \text{if the } i\text{th joint is rotational,} \\ 0 & \text{if the } i\text{th joint is translational.} \end{cases}$
- $\bar{\delta}_i = 1 - \delta_i$.
- O_{n+1}: a coordinate system attached to an external object, which has the same origin as O_n.
- $f_{n+1} = A_{n+1}^0 \bar{f}_{n+1}$, where $\bar{f}_{n+1}$ is the vector force exerted on an external object.
- $n_{n+1} = A_{n+1}^0 \bar{n}_{n+1}$, where $\bar{n}_{n+1}$ is the vector moment exerted on an external object.

If the ith link is translational, substitution of Eq. (1) into Eq. (27) gives

$$\tau_i = \sum_{k=i}^{n} (F_k - g_k) \cdot A_k^{i-1} z_{i-1} + f_{n+1} \cdot A_{n+1}^{i-1} z_{i-1}. \quad (5)$$

If the ith link is rotational, substitution of Eq. (4) into Eq. (26) gives

$$\begin{aligned} \tau_i = & \sum_{k=i}^{n} \left\{ N_k + \left(\sum_{j=i}^{k} A_k^j p_j + s_k \right) \times (F_k - g_k) \right\} \cdot A_k^{i-1} z_{i-1} \\ & + \left\{ \left(\sum_{j=i}^{n} A_{n+1}^j p_j \right) \times f_{n+1} \right\} \cdot A_{n+1}^{i-1} z_{i-1} + n_{n+1} \cdot A_{n+1}^{i-1} z_{i-1} \\ = & \sum_{k=i}^{n} N_k \cdot A_k^{i-1} z_{i-1} \\ & + \sum_{k=i}^{n} (F_k - g_k) \cdot \left\{ A_k^{i-1} z_{i-1} \times \left(\sum_{j=i}^{k} A_k^j p_j + s_k \right) \right\} \\ & + f_{n+1} \cdot \left\{ A_{n+1}^{i-1} z_{i-1} \times \sum_{j=i}^{n} A_{n+1}^j p_j \right\} \\ & + n_{n+1} \cdot A_{n+1}^{i-1} z_{i-1}. \end{aligned} \quad (6)$$

On the other hand, as shown in Appendix A, ω^k, v^k, and v^{k*} are expanded as

$$\omega^k = \sum_{j=1}^{k} \omega_j^k \dot{q}_j, \quad (7)$$

$$v^k = \sum_{j=1}^{k} v_j^k \dot{q}_j, \quad (8)$$

and

$$v^{k*} = \sum_{j=1}^{k} v_j^{k*} \dot{q}_j, \quad (9)$$

where

$$\omega_i^k = \delta_i A_k^{i-1} z_{i-1} \qquad (i = 1, \ldots, k), \quad (10)$$

$$v_i^k = \delta_i A_k^{i-1} z_{i-1} \times \sum_{j=i}^{k} A_k^j p_j + \bar{\delta}_i A_k^{i-1} z_{i-1} \qquad (i = 1, \ldots, k), \quad (11)$$

and

$$v_i^{k*} = \delta_i A_k^{i-1} z_{i-1} \times \left(\sum_{j=i}^{k} A_k^j p_j + s_k \right) + \bar{\delta}_i A_k^{i-1} z_{i-1} \qquad (i = 1, \ldots, k). \quad (12)$$

Therefore, Eqs. (5) and (6) simplify to

$$\begin{aligned} \tau_i = & \sum_{k=i}^{n} N_k \cdot \omega_i^k + \sum_{k=i}^{n} (F_k - g_k) \cdot v_i^{k*} \\ & + n_{n+1} \cdot \omega_i^{n+1} + f_{n+1} \cdot v_i^{n+1}. \end{aligned} \quad (13)$$

Rearranging Eqs. (10)–(13) yields the resolved Newton-Euler algorithm which is summarized in Table 3. The equivalence of this algorithm to the one based on Kane's dynamical equation is shown in Appendix C.

L'indexation renforce l'expressivité du langage et aboutit à des formules synthétiques ((7),(8),(9),(10),(13)). L'art du choix des notations se révèle notamment dans la formule (13) qui synthétise les formules (5) et (6) si dissemblables.

(extrait de : The Int. J. of Robotics Research Vol 8, No 1, Fév 1989, p 63–76 : A New Parallel Algorithm for Inverse Dynamics par K Hashimoto et H. Kimura)

Notation — Greek Letters

		Page
α	modular angle (elliptic function)	590
$\alpha_n(x)$	$=\int_1^\infty t^n e^{-xt}dt$	228
$\beta_n(x)$	$=\int_{-1}^1 t^n e^{-xt}dt$	228
$\beta(n)$	$=\sum_{k=0}^{\infty}(-1)^k(2k+1)^{-n}$	807
$B_x(a,b)$	incomplete beta function	263
$B(z,w)$	beta function	258
γ	Euler's constant	255
$\gamma(a,x)$	incomplete gamma function (normalized)	260
$\gamma_1=\frac{\mu_3}{\sigma_2}$	coefficient of skewness	928
$\gamma_2=\frac{\mu_4}{\sigma^4}-3$	coefficient of excess	928
$\Gamma(z)$	gamma function	255
$\Gamma(a,x)$	incomplete gamma function	260
δ_{ij}	Kronecker delta ($=0$ if $i\neq k$; $=1$ if $i=k$)	822
$\delta_x^n(f_n)$	central difference	877
Δ	difference operator	822
Δ	discriminant of Weierstrass' canonical form	629
$\Delta(f_n)$	forward difference	877
Δx	absolute error	14
$\zeta(x)$	Riemann zeta function	807
$\zeta(z)$	Weierstrass zeta function	629
$Z(u\|m)$	Jacobi's zeta function	578
$\eta(n)$	$=\sum_{k=1}^{\infty}(-1)^{k-1}k^{-n}$	807
η_ω	$=\zeta(\omega_\omega)$ Weierstrass elliptic function	631
$H(u)$, $H_1(u)$	Jacobi's eta functions	577
$\vartheta_x(z)$	theta function	576
$\vartheta_x(z\backslash\alpha)$, $\vartheta_x(z\backslash\alpha)$, $\vartheta_x(z\backslash\alpha)$, $\vartheta_x(z\backslash\alpha)$	Neville's notation for theta functions	578
$\Theta(u\|m)$	Jacobi's theta function	577
κ_n	nth cumulant	928
$\kappa_{mn}^{(pq)}$	joining factor for spheroidal wave functions	757
$\lambda(n)$	$=\sum_{k=0}^{\infty}(2k+1)^{-n}$	807
λ_{mn}	characteristic value of the spheroidal wave equation	753
$\Lambda_0(\varphi\backslash\alpha)$	Heuman's lambda function	595
$\mu(f_n)$	mean difference	877
$\mu(n)$	Möbius function	826
μ_n	nth central moment	928
μ'_n	nth moment about the origin	928
$\pi(x)$	number of primes $\leq x$	231
$\pi_n(x)$	$=(x-x_0)(x-x_1)\ldots(x-x_n)$	878
$\Pi(n;\varphi\backslash\alpha)$	elliptic integral of the third kind	590
$\Pi(z)$	factorial function	255
ρ	correlation coefficient	936
$\rho_n(x_0,x_1,\ldots,x_n)$	reciprocal difference	878
$\rho_m(\nu,x)$	Poisson-Charlier function	509
σ	standard deviation	298
σ^2	variance	928
$\sigma(z)$	Weierstrass sigma function	629
$\sigma_k(n)$	divisor function	827
$\tau_n(x)$	tetrachoric function	934
$\varphi=$am u,	amplitude	569
$\varphi(n)$	Euler-Totient function	826
$\varphi(t)=E(e^{itX})$	characteristic function of X	928
$\Phi(a;b;z)$	confluent hypergeometric function	504
$\psi(z)$	logarithmic derivative of the gamma function	258
$\Psi(a;c;z)$	confluent hypergeometric function	504
ω_ω	period of Weierstrass elliptic functions	629
$\omega_{n,s}(x)$	Cunningham function	510

Miscellaneous Notations

		Page
$[a_{ij}]$	determinant	19
$[a_i]$	column matrix	19
∇^2	Laplacian operator	752
Δ_x^+	forward difference operator	877
$\frac{\partial}{\partial x}$	partial derivative	883
i	$(=\sqrt{-1})$	70
$\binom{n}{k}$	binomial coefficient	10
$n!$	factorial function	255
$(2n)!!$	$=2\cdot4\cdot6\ldots(2n)=2^n n!$	258
(m,n)	greatest common divisor	822
(n,k)	$=\frac{\Gamma(\frac{1}{2}+n+k)}{k!\Gamma(\frac{1}{2}+n-k)}$ (Hankel's symbol)	437
$(n;n_1,n_2,\ldots,n_m)$	multinomial coefficient	823
$[x]$	largest integer $\leq x$	66
$\langle x\rangle$	nearest integer to x	222
$\bar{z}$	complex conjugate of z $(=x-iy)$	16
$z=x+iy$	complex number (Cartesian form)	16
	$=re^{i\theta}$ (polar form)	16
$\|z\|$	absolute value or modulus of z	16
Σ	overall summation	822
Σ'	restricted summation	755
$\sum_p\prod_p$	sum or product taken over all prime numbers p	807
$\sum_{d\|n}\prod_{d\|n}$	sum or product overall positive divisors d of n	826
$\int$	Cauchy's principal value of the integral	228
$=$	approximately equal	14
$\sim$	asymptotically equal	15
$<,>,\leq,\geq$	inequality, inclusion	10
$\neq$	unequal	12

24.3. Number Theoretic Functions

24.3.1 The Möbius Function

I. Definitions

A. $\mu(n)=1$ if $n=1$

$=(-1)^k$ if n is the product of k distinct primes

$=0$ if n is divisible by a square >1.

B. Generating functions

$$\sum_{n=1}^{\infty}\mu(n)n^{-s}=1/\zeta(s) \qquad \Re s>1$$

$$\sum_{n=1}^{\infty}\frac{\mu(n)x^n}{1-x^n}=x \qquad |x|<1$$

II. Relations

A. Recurrence

$$\mu(mn)=\mu(m)\mu(n) \text{ if } (m,n)=1$$

$$=0 \text{ if } (m,n)>1$$

B. Check

$$\sum_{d|n}\mu(d)=\delta_{n1}$$

C. Numerical analysis

$g(n)=\sum_{d|n}f(d)$ for all n if and only if

$$f(n)=\sum_{d|n}\mu(d)g(n/d) \text{ for all } n$$

$g(n)=\prod_{d|n}f(d)$ for all n if and only if

$$f(n)=\prod_{d|n}g(n/d)^{\mu(d)} \text{ for all } n$$

$g(x)=\sum_{n=1}^{\lfloor x\rfloor}f(x/n)$ for all $x>0$ if and only if

$$f(x)=\sum_{n=1}^{\lfloor x\rfloor}\mu(n)g(x/n) \text{ for all } x>0$$

$g(x)=\sum_{n=1}^{\infty}f(nx)$ for all $x>0$ if and only if

$$f(x)=\sum_{n=1}^{\infty}\mu(n)g(nx) \text{ for all } x>0$$

and if $\sum_{m=1}^{\infty}\sum_{n=1}^{\infty}|f(mnx)|=\sum_{n=1}^{\infty}\sigma_0(n)|f(nx)|$ converges.

The cyclotomic polynomial of order n is $\prod_{d|n}(x^d-1)^{\mu(n/d)}$

III. Asymptotics

$$\sum_{n=1}^{\infty}\frac{\mu(n)}{n}=0$$

$$\sum_{n=1}^{\infty}\frac{\mu(n)}{n}\ln n=-1$$

$$\sum_{n\leq x}\mu(n)=0(xe^{-c\sqrt{\ln x}})$$

Les "Handbook of Mathematical Functions" sont des dictionnaires de fonctions mais aussi un recueil de conventions de notations.

L'intelligence permet d'exploiter de tels résultats en ne les connaissant que très partiellement, bien qu'en sachant souvent évaluer l'intérêt mathématique justifiant l'existence d'une telle fonction

Un défi des mathématiques consiste à analyser syntaxiquement et sémantiquement un tel contenu statique afin de l'exploiter ultérieurement dans la résolution de problèmes mathématiques (ou son aspect textuel pour les notations). Un procédé général est à l'étude...

Quels sont les usages possibles de ces propriétés ? Sont-ils modélisables ?

ANALYSEUR

```
%-*- Mode:Prolog -*-
Machine Lisp T.I. Explorer
%-----------------------------------------------------------------------
%%lancement d'analyse
```

Le code de cet analyseur illustre clairement l'absence de langage d'expression de haut niveau que les DCG (Definite Clause Grammar) et la programmation par contraintes commencent à combler.

```
ana(_string,_list,_ast):-
   chrono_start(int),
   int(_string,[],_list),
   chrono(int,T),
   presente(_synth,_list),
   nl,nl,write('ceci est le resume de l analyse lexicale obtenu en '),
   write(T), write(' s'),
   nl,nl, tab(20), affiche(_synth),nl,nl,
   chrono_start(analyse),
   analyse(_list,_ast,[_,_,_,_,_]),
   chrono(analyse,A),
   nl,nl,write('ceci est le resume de l ast obtenu en '),
   write(A), write(' s'),
   nl,nl, tab(20), visu(_ast),nl,nl.

chrono_start(X):-
   retractall(nbr_courant(X,_)),
   time(D),
   assert(nbr_courant(X,D)).
chrono_start(X):-
   retractall(nbr_courant(X,_)), fail.

chrono(X,R):-
   time(F),
   nbr_courant(X,D), !,%pour etre sur qu'il n'y en ait qu un!
   T is float((F-D)/1000)/1000,
   (T < 0, !, R is T+8 ; R=T).
chrono(X,_):-
   retractall(nbr_courant(X,_)),
   time(T),
   assert(nbr_courant(X,T)),
   fail.

%problemes de boucle infinie dans append si B finit par une var
visu(T):- !, code(T,[A,B]),(B=[], ! ; true), append(L,B,A), name(X,L), write(X).
visu(nuple(_)) :- write('parenth').
visu(terminal(N,_2,_3,_4,_5)):- write(N).
visu(wif([A,B|C],_2,_3,_4,_5)):- visu(A),write(' { '), visu(B), visu_list(C).

visu_list([]):-  !, write(' } ').
visu_list([A|B]):- write(' ; '), visu(A), visu_list(B).

presente([],[]):- !.
presente([A|B],[[nuple(_)|X]|Y]):- !, presente_list(A,X), presente(B,Y).
presente([A|B],[X|Y]):- nom(X,A), !, presente(B,Y).
presente(_,_):- write('echec de presentation').

presente_list([],[]).
presente_list([E|L],[X|R]):- presente(E,X), presente_list(L,R).


affiche([E]):- !, affiche_elt(E).
affiche([E|L]):-affiche_elt(E), write(' '), affiche(L).
affiche([]):- write('affiche-vide').

affiche_elt([]):- !, write('elt-vide').
affiche_elt([E|L]):- !, write('['),affiche(E),affiche_suite(L).
affiche_elt(A):- write(A).
```

```
affiche_suite([]):- write(']').
affiche_suite([E|L]):- write(' ; '), affiche(E), affiche_suite(L).

%-------------------------------------------------------------------------------
%%initialisation bdf

%avoir une bdf avec un load file et un acces decouple du foncteur "terminal"

nettoyer_bdf :-
   nl,write('j initialise la Bdf'),nl,
   abolish(terminal,5),
   abolish(nbr_courant,2),
   assert((terminal(_91,_92,_93,_94,_95):- terminal_nbr(_91,_92,_93,_94,_95))),

   append("sin",_1,_2),
   assert(terminal(sinus,[reel,reel],[_2,_1],150,[non,100,pre,non,no3])),
   append("sin",_3,_4),
   assert(terminal(fantoche,[reel,reel],[_4,_3],110,[non,100,pre,non,no1])),
   append("si",_5,_6),
   assert(terminal(pipo,[reel],[_6,_5],160,[])),

   append("cos",_7,_8),
   assert(terminal(cosinus,[reel,reel],[_8,_7],150,[non,100,pre,non,no3])),
   append("tg",_9,_10),
   assert(terminal(tangente,[reel,reel],[_10,_9],150,[non,100,pre,non,no3])),
   append("cotg",_11,_12),
   assert(terminal(cotangente,[reel,reel],[_12,_11],150,[non,100,pre,non,no3]))
   append("Log",_13,_14),
   assert(terminal(logarithme,[reel,reel],[_14,_13],150,[non,100,pre,non,no3]))
   append("*",_15,_16),
   assert(terminal(fois,[reel,[reel,reel]],[_16,_15],150,[oui,80,inf,gch,no2]))
   append("+",_17,_18),
   assert(terminal(plus,[reel,[reel,reel]],[_18,_17],150,[oui,70,inf,gch,no2]))
   append("!",_19,_20),
   assert(terminal(fact,[reel,reel],[_20,_19],150,[oui,120,suf,non,no2])),
   append("pi_en_grec",_21,_22),
   assert(terminal(pi,[reel],[_22,_21],160,[])),
   append("pi",_211,_221),
   assert(terminal(pi,[reel],[_221,_211],160,[])),

   append("o",_212,_222),
   assert(terminal(compo,[reel,[reel,reel],[reel,reel]],[_222,_212],160,[oui,11


   append("a",_23,_24),
   assert(terminal(cst(0),[reel],[_24,_23],140,[])),
   append("x",_25,_26),
   assert(terminal(var(0),[reel],[_26,_25],140,[])),
   append("h",_27,_28),        %attention h est une fonction de 2 variables
   assert(terminal(fct(0),[reel,[reel,reel]]
      ,[_28,_27],140,[oui,100,pre,non,no1])),

   assert(nbr_courant(cst,1)),
   assert(nbr_courant(var,1)),
   assert(nbr_courant(fct,1)).

%%terminal(nom,[codom|dom],["symb"||X,X],prio,[beta,prio_ass,fix,assoc,ass]).
%specif : 5ieme champ tout instancie ou [] sinon pb de capture de param poss.
%functor(T,terminal,5); arg/3...
%ass: no1 = par oblig, no2 = par en fct prio, no3 = par sf si arg est terminal
%     no4 = no2 avec operateur implicite ""
```

```
terminal_nbr(X,[reel],[A,B],160,[]):- number(X), !,name(X,S),append(S,B,A).
terminal_nbr(_1,_2,[A,_],_4,_5):- var(A), !, write('cas non prevu dans term_nbr'
terminal_nbr(X,[reel],[A,B],160,[]):- trouve_nbr(A,B),append(S,B,A),name(X,S).

%attention celui ci doit etre infixe !? ...
terminal_impl(fois,[reel,reel,reel],[X,X],170,[oui,80,inf,gch,no4]).

trouve_nbr([N|A],B):-integer(N) ,deb_nbr([N|A],C),C \== [N|A],rest_nbr(C,B).

deb_nbr([N|A],R):- N >= 48, N =< 57,! ,deb_nbr(A,R).
deb_nbr(R,R).    % entre 0=48  et 9

rest_nbr(S,R):- append(".",A,S), !,deb_nbr(A,R).
rest_nbr(S,S).

%-------------------------------------------------------------------------
%%fct d'acces
terminal(T):- functor(T,terminal,5).
nom(T,N):- arg(1,T,N).
type(T,R):-arg(2,T,R).
dom(T,D):-arg(2,T,[D|_]).
codom(T,D):-arg(2,T,[_|D]).
code(T,C):- arg(3,T,C).
prio(T,P):-arg(4,T,P).
synt(T,S):-arg(5,T,S).
beta(T,B):-arg(5,T,[B,_,_,_,_]).
prio_ass(T,P):-arg(5,T,[_,P,_,_,_]).
fix(T,F):-arg(5,T,[_,_,F,_,_]).
assoc(T,A):-arg(5,T,[_,_,_,A,_]).
ass(T,A):-arg(5,T,[_,_,_,_,A]).

%-------------------------------------------------------------------------


%%ajout a la bdf de symboles implicites

gere_impl(_,_,_,[]):- !, fail.

gere_impl(A,B,R,[S|_]):-
   code(S,[A,B]),
   norme(S,R),
   ajoute_bdf(R),
   retire_bdf(R).

gere_impl(A,B,R,[_|L]):-
   gere_impl(A,B,R,L).


ajoute_bdf(T) :-
   nom(T,F),
   F =..[A,N],
   retract(nbr_courant(A,_)),
   M is N+1,
   assert(nbr_courant(A,M)),
   nl,write('on ajoute : '),write(T), write(' a la BDF'),
   assert(T), !.  %le slash permet d'eviter le redo du retract
         %on ne peut le mettre dans gere_impl a cause de la clause no3
```

```
retire_bdf(_).  %effet de bord qui ne fait rien avant le backtrack
retire_bdf(T):-
   nom(T,F),
   F =..[A,N],
   retract(nbr_courant(A,_)),
   assert(nbr_courant(A,N)),
   nl,write('je retire :  '),write(T),
   retract(T), !, fail.

norme(terminal(_1,_2,[X,Y],_4,_5),terminal(_1,_2,[A,B],_4,_5)):-
   normalise(X,Y,A,B).

normalise(X,Y,X,Y):-var(Y), !.
normalise(X,Y,A,B):-append(S,Y,X),append(S,B,A).

%%consultation de symboles

trouve_expl(A,T):-
   terminal(_1,_2,[A,_3],_4,_5),
   terminal(_1,_2,[A,_3],_4,_5) = T.

trouve_impl([N|B],terminal(cst(X),[reel],[[N|B],B],160,[])) :-
   97 =< N, 101 >= N, nbr_courant(cst,X), !.  %entre a et e : cst
trouve_impl([N|B],terminal(fct(X)
                ,[reel,reel],[[N|B],B],160,[oui,100,pre,non,no1])) :-
   102 =< N, 104 >= N, nbr_courant(fct,X), !.  %entre f et h : fct
trouve_impl([N|B],terminal(var(X),[reel],[[N|B],B],160,[])) :-
   105 =< N, 122 >= N, nbr_courant(var,X).  %entre i et z : var
trouve_impl([N|B],terminal(var(X),[entiere],[[N|B],B],160,[])) :-
   105 =< N, 122 >= N, nbr_courant(var,X).       % essai pipo de nondet


%quand on obtient une fonction quelle est sa syntaxe?
defaut_synt([non,100,pre,non,no1]).

%-------------------------------------------------------------------------

%%choix des meilleurs
%la dichotomie impl expl ne tient pas pour l'instant
% WARNING on prend tout pour l'instant

meilleurs_expl(X,X):-!.

meilleurs_expl([E|Z],T) :- meilleurs_exaux(Z,[E],T).


meilleurs_exaux([],R,R).
meilleurs_exaux([E|L],[F|_],R):-
   compare_expl(>,E,F),
   meilleurs_exaux(L,[E],R).
meilleurs_exaux([E|L],[F|M],R):-
   compare_expl(=,E,F),
   meilleurs_exaux(L,[E,F|M],R).
meilleurs_exaux([E|L],[F|M],R):-
   compare_expl(<,E,F),
   meilleurs_exaux(L,[F|M],R).

compare_expl(R,P,Q):-
   prio(P,A),
   prio(Q,B),
   compare(R,A,B). %pred systeme
```

```
meilleurs_impl(X,X):-!.

meilleurs_impl([E|Z],T) :- meilleurs_imaux(Z,[E],T).

meilleurs_imaux([],R,R).
meilleurs_imaux([E|L],[F|_],R):-
   compare_impl(>,E,F),
   meilleurs_imaux(L,[E],R).
meilleurs_imaux([E|L],[F|M],R):-
   compare_impl(=,E,F),
   meilleurs_imaux(L,[E,F|M],R).
meilleurs_imaux([E|L],[F|M],R):-
   compare_impl(<,E,F),
   meilleurs_imaux(L,[F|M],R).

compare_impl(R,P,Q):-
   prio(P,A),
   prio(Q,B),
   compare(R,A,B). %pred systeme

%-----------------------------------------------------------------------

%fonction d'interpretation(tete-dl,queue-dl,interpr)

int([C|X],Y,[[nuple(A)|B]|I]):-
   [C] = "(" , !, [D] = ")", par(X,[D|Z],[nuple(A)|B]), int(Z,Y,I).

int([C|X],Y,[[nuple(A)|B]|I]):-
   [C] = "[" , !, [D] = "]", par(X,[D|Z],[nuple(A)|B]), int(Z,Y,I).

int(X,Y,[R|Z]):-
   bagof(T,(trouve_expl(X,T)),L), !,
   meilleurs_expl(L,M),
   member(R,M),
   code(R,[X,U]),
   int(U,Y,Z).

int(X,Y,[R|Z]):-
   bagof(T,(trouve_impl(X,T)),L), !,
   meilleurs_impl(L,M),
   gere_impl(X,U,R,M),
   int(U,Y,Z).

int(X,X,[]).

int(X,Y,_):- var(X),nonvar(Y),write('debordement de int').
%ceci correspond a un cas du type int(sin.U,in.U,Z)


%cas d'un parenthesage

par(X,Y,[nuple(A),I|K]):-
   argument(X,Z,I), suitepar(Z,Y,K), length([I|K],A).

suitepar([C|X],Y,[I|K]):-
   [C] = "," , !, argument(X,Z,I), suitepar(Z,Y,K).
suitepar(Y,Y,[]).

argument(X,Y,R):- int(X,Y,R), R \== [].
```

```
%-----------------------------------------------------------------------
%-----------------------------------------------------------------------

%%analyse syntaxico-semantique

%traite recursivement uniqt les nuple prof gche first
ananuple([[nuple(I)|X]|Y],[A|B]):- !,
   anamult(X,C),betared([wif(nuple(I),_,_,_,_)|C],A),ananuple(Y,B).

ananuple([terminal(_1,_2,_3,_4,_5)|Y],[wif(_1,_2,_3,_4,_5)|B]):- !,
   ananuple(Y,B).  %tout doit etre sous forme de wif

ananuple([A|Y],[A|B]):- ananuple(Y,B).

ananuple([],[]).


%anamult prend une liste de listes a traiter ex: args de nuple
%et renvoie la liste des arbres semantiques wff
anamult([],[]).
anamult([X|Y],[A|B]):- analyse(X,A,[_,_,_,_,_]), anamult(Y,B).


%_att= codage ou lge pour exprimer des contraintes
% ou alors l-valeurs ou l-contraintes ds meta-lge(predicats a verifier)
%pour l'instant att=[] si terminal, att=[_,_,_,_,_]sinon.
% 1)specifie la forme de l'arbo
% 2)         son type
% 3)la forme externe ie d-liste des codes char
% 4)la prio minimale
% 5)l'usage syntaxique == beta si c'est un operateur a betareduire

analyse(_list,_ast,_att) :-
   ananuple(_list,_suite),
   verif_fix(_suite),
   ana_rec(_suite,_ast,_att),
   verifie(_suite,_ast).    %pour verifier la coherence en generation

%verif_fix regarde si liste est correct vis a vis de fix
verif_fix([E|_]):-
   beta(E,oui),
   fix(E,X),
   member(X,[inf,suf]), !,
   fail.
verif_fix(L):-
   last(L,E),
   beta(E,oui),
   fix(E,X),
   member(X,[pre,inf]), !,
   fail.
verif_fix(_).

verifie(_,_).
```

```
ana_rec([E],E,[_1,_2,_3,_4,_5]) :- !, verif_att(E,[_1,_2,_3,_4,_5]).
ana_rec(_,_,[N,_2,_3,_4,_5]) :-
   N == [nuple(_)|_], !, %== si on ne veut pas que tout matche
   fail.     %a enlever si les nuples ne sont pas traites separement
ana_rec(L,I,A) :-
   append([E|L1],[F|L2],L), %coupe en 2 parties \== 0
   decompose([E|L1],[F|L2],I,A).


decompose(L1,[O|L2],I,A) :-
   fix(O,inf), !,     % le ! suppose inf sont beta obliges
   L2 \== [],
   att_op(inf,A,A0),
   verif_att(O,A0),
   att_des_args(O,A,A1,A2),
   ana_rec(L1,I1,A1),
   ana_rec(L2,I2,A2),
   betared([O,I1,I2],I),
   verif_att(I,A).

decompose(L1,L2,I,A):-
   verif_fix(L1),
   terminal_impl(_1,_2,_3,_4,_5),
   O = wif(_1,_2,_3,_4,_5), %le reste comme si explicite
   att_op(inf,A,A0),
   verif_att(O,A0),
   att_des_args(O,A,A1,A2),
   ana_rec(L1,I1,A1),
   ana_rec(L2,I2,A2),
   betared([O,I1,I2],I),
   verif_att(I,A).

decompose([O],L2,I,A):-
   fix(O,pre),
   att_op(pre,A,A0),
   verif_att(O,A0),
   att_arg(O,A,A2),
   ana_rec(L2,I2,A2),
   betared([O,I2],I),
   verif_att(I,A),    !.

decompose(L1,[O],I,A):-
   fix(O,suf),
   verif_fix(L1),
   att_op(suf,A,A0),
   verif_att(O,A0),
   att_arg(O,A,A1),
   ana_rec(L1,I1,A1),
   betared([O,I1],I),
   verif_att(I,A),    !.

decompose(L1,L2,I,A):-
   verif_fix(L1),
   att_op(pre,A,[N,T,D,P,S]),
   (number(P), P>= 100, !, Q=P ; Q=100),
   ana_rec(L1,O,[N,T,D,Q,S]),
   fix(O,pre),  %non indispensable car les ops composes sont pre
   att_arg(O,A,A2),
   ana_rec(L2,I2,A2),
   betared([O,I2],I),
   verif_att(I,A),
%possible de remplacer par (L1,[E],I,A) car soit atomique soit nuple ?
   (L2 = [_], ! ;
      nl, write('decompose marche avec : '),write(L1), write(L2)).
%on suppose qu'on ne peut produire un op suffixe avec une liste
```

```
%verif_att(+wif(_,_,_,_,_),+att) wif car on peut avoir un op beta-oblig seul!
verif_att(O,[N,T,_,P,S]):-
   type(O,T),
%A CAUSE DE LA SUPPRESSION DE ()...   nom(O,N),
   prio(O,X),
   (integer(P), !, X>=P ; true),
   (beta(O,oui), !, S==beta ; ( S == atomic, !, \+(N=[_|_]) ; true)).
%si beta oblig alors on doit etre en position S==beta
%si on est en position S==atomic, N ne doit pas etre une liste

%att_op(+fix,+att_beta,-att_opeseul)
att_op(inf,[[O,_,_],[D],_3,_4,_5],[O,[D,_,_],_3,_4,beta]):- !.
att_op(_,[[O,_],[D],_3,_4,_5],[O,[D|_],_3,_4,beta]).

%rque ces deux preds transmettent les contraintes syntaxiques(no1...)
%att_des_args(+op,+att_beta,-att_gch,-att_drt)
att_des_args(O,[[N0,N1,N2],[D],_,_,_],
               [N1,[D1],_,P1,Q1],[N2,[D2],_,P2,Q2]):-
   nom(O,N0),
   type(O,[D,D1,D2]),
   (prio_ass(O,P), !, assoc(O,A), prio_args(A,P,P1,P2) ; P1 = 100, P2 = 100),
% ex?-BUG de capture arg ligne ci dessous!
   (ass(O,S), !, par_arg(S,N1,Q1), par_arg(S,N2,Q2) ; true).
%rque : on passe les noms lorsque la priorite a passer est aussi connue!

%att_arg(+op,+att_beta,-att_delarg)
att_arg(O,[[N0,N1],[D],_,_,_],[N1,D1,_,P1,Q1]):-
   nom(O,N0),
   type(O,[D|D1]),
   (prio_ass(O,P1), ! ; P1 = 100),   %INVERSEES pour nondet pararg!!!
   (ass(O,S), !, par_arg(S,N1,Q1) ; true).

%prio_args(+assoc_pere,+prio_pere,_prio_gch,_prio_drt)
prio_args(non,P,Q,Q):- !, Q is P+10.
prio_args(gch,P,P,Q):- !, Q is P+10.
prio_args(drt,P,Q,P):- !, Q is P+10.

%par_arg donne la nature du parenthesage de l'arg
par_arg(no1,[nuple(_)|_],_):- !.
par_arg(no2,_,_):- !.
par_arg(no4,_,_):- !.
par_arg(no3,[nuple(_)|_],_). %pb du lge d'expression trop pauvre
par_arg(no3,_,atomic).



%-----------------------------------------------------------------------


%%betared([op|args],arbo)cree  l'arbre semant apres beta reduction
%en particulier il cree une nvlle wif en conservant que les noms
%de ses filles (il jette leur envt wif)

betared(wif(_1,_2,_3,_4,_5),wif(_1,_2,_3,_4,_5)):- !.

betared(terminal(_1,_2,_3,_4,_5),wif(_1,_2,_3,_4,_5)):- !,
   write('warning terminal change en wif : '), write(_1).

betared([wif(nuple(I),_,_,_,_)|X],wif([nuple(I)|A],B,[C,D],200,E)):- !,
   mapnom(X,A),
   maptype(X,B),
   mapnuple(X,C,D),
   (X=[F], synt(F,E),!  ;  E =[]).
```

```
%ici les J ne sont plus wif
betared([X,wif([nuple(_)|J],F,[C,D],_4,_5)],wif([N|J],[B],[M,D],P,E)):- !,
   nom(X,N), %lors du beta d'une fct pre ou suf on enleve les () ds l'arbo
   type(X,[B|F]),
   code(X,[K,L]),       %PROBLEME positionne les suffixes avant!
   normalise(K,L,M,C), % unif des 2 extremites M,C suivi de C,D
   (prio_ass(X,P),!  ;  P=100),
   (B=[_|Z], Z \== [], defaut_synt(E),!  ;  E=[]).

betared([X|Y],wif(A,[B],[C,D],P,E)):- !,
   mapnom([X|Y],A),
   maptype(Y,F),
   type(X,[B|F]),
   mapfct([X|Y],C,D),
   (prio_ass(X,P),!  ;  P=100),
   (B=[_|Z], Z \== [], defaut_synt(E),!  ;  E=[]).

betared(X,Y):-
   nl, write('echec de betared avec : '), write(X), tab(3), write(Y).

%mapXXXX(l-wif,fct-synthetisee)
mapnom([],[]).
mapnom([X|Y],[A|B]):- mapnom(Y,B), nom(X,A).

maptype([],[]).
maptype([X|Y],R):- maptype(Y,B), type(X,A), append(A,B,R).

%mapfct construit le code du betared sous forme de dlist
mapfct([X,Y],E,D):-
   fix(X,pre), !,
   code(X,[A,B]),
   code(Y,[C,D]),
   normalise(A,B,E,C).
mapfct([X,Y],E,B):-
   fix(X,suf), !,
   code(X,[A,B]),
   code(Y,[C,D]),
   normalise(C,D,E,A).
mapfct([X,Y,Z],H,F):- !,
   code(X,[A,B]),
   code(Y,[C,D]),
   code(Z,[E,F]),
   normalise(A,B,G,E),
   normalise(C,D,H,G).

%mapnuple construit le nuple en rajoutant () et ,
mapnuple([X|Y],A,D):-
   mapnupleaux(Y,C,D),  %l'extremite drte D est partagee
   code(X,[E,F]),
   normalise(E,F,G,C),
   append("(",G,A).

mapnupleaux([],A,[]):- append(")",[],A).
mapnupleaux([X|Y],A,D):-
   mapnupleaux(Y,C,D),  %l'extremite drte D est partagee
   code(X,[E,F]),
   normalise(E,F,G,C),
%   normalise(E,F,G,H),  old code equivalent
%   H = C,  %append d-list G,H,C,D ==> G,D
   append(",",G,A).

%-----------------------------------------------------------------------
%%utilitaires

last([A],A) :- !.
last([_|B],C) :- last(B,C).
```

La définition et l'usage d'une notation : Exemples d'utilisation de Σ

[Godement 66]

p107 tique, *la notation additive s'emploie uniquement pour des lois de composition commutatives.*

Soit $(x, y) \to x \perp y$ une loi de composition *associative et commutative* sur un ensemble X, et soit $(x_i)_{i \in I}$ une famille *finie* d'éléments de X. Soit n le nombre d'éléments de I et écrivons ceux-ci sous forme d'une suite $i_1, \ldots, i_n$; l'élément

$$x_{i_1} \perp x_{i_2} \perp \cdots \perp x_{i_n}$$

de X ne dépend évidemment pas (vu l'associativité et la commutativité de la loi de composition considérée) de la façon dont on a écrit les éléments de I sous la forme d'une suite de n termes. On pose alors, par définition,

$$\bigperp_{i \in I} x_i = x_{i_1} \perp \cdots \perp x_{i_n}.$$

Si la loi de composition considérée est écrite *multiplicativement*, on écrit

$$\prod_{i \in I} x_i = x_{i_1} \cdots x_{i_n};$$

si elle est écrite *additivement*, on utilise la notation

$$\sum_{i \in I} x_i = x_{i_1} + \cdots + x_{i_n}.$$

Remarque 1. Les notations condensées qu'on vient d'introduire subissent dans la pratique de nombreuses modifications que l'usage enseignera. Par exemple, si I est l'ensemble formé des entiers $1, \cdots, n$, on écrit souvent

$$x_1 + \cdots + x_n = \sum_{i=1}^{i=n} x_i \quad \text{ou} \quad \sum_{1 \leqslant i \leqslant n} x_i;$$

si I est l'ensemble des couples (i, j) d'entiers tels que $1 \leqslant i \leqslant p$, $1 \leqslant j \leqslant q$, et si l'on note x_{ij} le terme « général » de la famille considérée, on utilise fréquemment la notation

$$\sum_{\substack{1 \leqslant i \leqslant p \\ 1 \leqslant j \leqslant q}} x_{ij} \quad \text{au lieu de} \quad \sum_{(i,j) \in I} x_{ij};$$

on notera que, dans ce cas, l'associativité de la loi de composition se traduit par la relation

$$\sum_{\substack{1 \leqslant i \leqslant p \\ 1 \leqslant j \leqslant q}} x_{ij} = \sum_{1 \leqslant i \leqslant p} \left(\sum_{1 \leqslant j \leqslant q} x_{ij} \right),$$

où le second membre désigne la somme

$$(x_{11} + x_{12} + \cdots + x_{1q}) + \cdots + (x_{p1} + x_{p2} + \cdots + x_{pq});$$

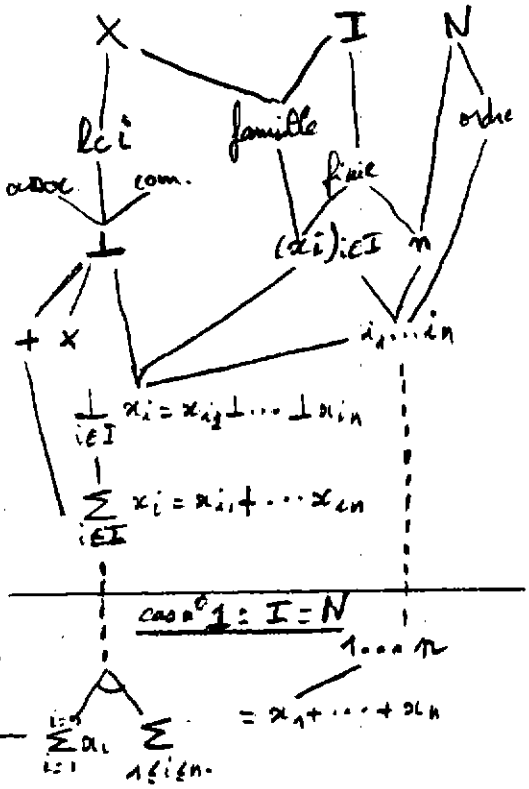
l'emploi de ces notations condensées est souvent indispensable pour éviter des formules inextricables.

Notons enfin que dans la notation

$$\sum_{i \in I} x_i,$$

la lettre i ne joue aucun rôle et n'intervient pas réellement dans le résultat — elle indique simplement une opération à effectuer (à savoir prendre la somme de tous les x_i obtenus en faisant varier i dans I), et on peut la remplacer par toute autre lettre *non encore utilisée par ailleurs* (cette dernière précaution est essentielle pour éviter des erreurs grossières).

2



cas n°2 : $I = I_1 \times I_2$

notation de remplacement

équivalence permettant la comparaison avec la notation simple

pragmatique de manipulation chargée de rendre compte de la notion de variable muette qui n'a pas été définie explicitement. Cette règle indique alors sous quelles conditions syntaxiques l'interprétation mathématique introduite précédemment reste équivalente.

---

p560 (où K est l'anneau de base) en posant

$$(*) \qquad h(x_1, \ldots, x_{p+q}) = \sum_{\substack{s \in \mathfrak{S}_{p+q} \\ s(1) < \cdots < s(p) \\ s(p+1) < \cdots < s(p+q)}} \varepsilon(s) \cdot f(x_{s(1)}, \ldots, x_{s(p)}) \cdot g(x_{s(p+1)}, \ldots, x_{s(p+q)})$$

où la sommation est étendue à toutes les permutations s des entiers $1, \ldots, p+q$ qui respectent l'ordre des p premiers, ainsi que des q derniers, de ces entiers.

la notation est étendue à une véritable description des termes concernés

---

p354 $d_n = a_n b_0 + a_{n-1} b_1 + \cdots + a_1 b_{n-1} + a_0 b_n = \sum_{r+s=n} a_r b_s$ $\left(\text{et non } \sum_{0 \leqslant i \leqslant n} a_{n-i} b_i\right)$

l'usage de la notation vise à améliorer sa clarté et non à respecter les conventions

p605 §§ 6. Montrer qu'il existe, sur l'ensemble des entiers $n \geqslant 1$, une et une seule fonction μ (**fonction de Möbius**) à valeurs entières, vérifiant la relation

$$\sum_{d|n} \mu(d) = \begin{cases} 1 & \text{si } n = 1 \\ 0 & \text{si } n > 1 \end{cases}$$

(la somme est étendue aux diviseurs d de n tels que $1 \leqslant d \leqslant n$).

§§ 7. Soit f une fonction sur l'ensemble des entiers $n \geqslant 1$, et à valeurs dans un groupe additif A. On définit une nouvelle fonction g en posant

$$g(n) = \sum_{d|n} f(d)$$

où la somme est étendue aux diviseurs d de n tels que $1 \leqslant d \leqslant n$. Montrer qu'on a inversement

L'extension de la notation s'effectue implicitement : l'auteur se contente de préciser l'interprétation à adopter

---

p271 si l'on a trois familles finies $(x_i)_{i\in I}$, $(y_j)_{j\in J}$ et $(z_k)_{k\in K}$ d'éléments d'un anneau, on a la relation

$$(7) \qquad \sum_{i\in I} x_i \cdot \sum_{j\in J} y_j \cdot \sum_{k\in K} z_k = \sum_{\substack{i\in I \\ j\in J \\ k\in K}} x_i y_j z_k;$$

plus généralement encore, supposons données p familles finies d'éléments d'un anneau, familles que nous noterons $(x_{1i_1})_{i_1\in I_1}, \ldots, (x_{pi_p})_{i_p\in I_p}$; alors on a

$$(8) \qquad \sum_{i_1\in I_1} x_{1i_1} \cdots \sum_{i_p\in I_p} x_{pi_p} = \sum_{i_1\in I_1, \ldots, i_p\in I_p} x_{1i_1} \cdots x_{pi_p}.$$

Remarque 2. Dans la formule (6) il arrive fréquemment que I = J et qu'on désigne les familles données par $(x_i)_{i\in I}$ et $(y_i)_{i\in I}$; on fera attention dans ce cas à *ne pas* écrire la relation (6) sous la forme

$$\sum_{i\in I} x_i \cdot \sum_{i\in I} y_i = \sum_{i\in I} x_i y_i$$

La notation est étendue aux familles multiples par superposition ou juxtaposition

Le Σ et les ... sont complémentaires

2 Les extensions produisent des cas particuliers que l'usage mathématique doit traiter

---

p275 Il reste à montrer que la relation (13), i.e.

$$f(x, y) = \sum u_i(x) v_j(y) c_{ij},$$

implique nécessairement $c_{ij} = f(a_i, b_j)$. Or on a

$$f(a_i, b_j) = \sum_{k,h} u_k(a_i) v_h(b_j) c_{kh}$$

Les variables d'index sont évidentes lorsqu'elles comprennent tous les indices présents dans le terme principal

---

p607 § 13. Soient $X_1, \ldots, X_n$ des indéterminées sur un anneau commutatif K. On appelle **fonctions symétriques élémentaires** de $X_1, \ldots, X_n$ les polynômes

$$s_1 = X_1 + X_2 + \cdots + X_n = \Sigma X_i$$

$$s_2 = X_1X_2 + \cdots + X_{n-1}X_n = \sum_{1\leqslant i<j\leqslant n} X_iX_j$$

$$s_3 = X_1X_2X_3 + \cdots = \sum_{1\leqslant i<j<k\leqslant n} X_iX_jX_k$$

$$\cdots\cdots\cdots\cdots\cdots\cdots$$

$$s_n = X_1X_2\ldots X_n$$

Le sens intuitif du Σ s'obtient en reconstituant les termes à ajouter :
- leur forme (ou structure)
- leurs propriétés
- la façon de les générer

a) Calculer à l'aide des fonctions symétriques élémentaires les polynômes suivants :

$$X_1^2X_2 + X_1X_2^2 + X_1^2X_3 + X_2X_3^2 + X_2^2X_3 + X_3X_2^2 \quad (n = 3)$$

$$(2X_1 - X_2 - X_3)(2X_2 - X_1 - X_3)(2X_3 - X_1 - X_2) \quad (n = 3)$$

$$(X_1X_2 + X_3X_4)(X_1X_3 + X_2X_4)(X_1X_4 + X_2X_3) \quad (n = 4)$$

$$\sum_{i\neq j} X_i^2X_j; \quad \sum_{i\neq j\neq k} X_i^2X_jX_k; \quad \sum_{i\neq j\neq k} X_iX_jX_k; \quad \sum_{\varepsilon\in\mathfrak{S}_n} (a_1X_{\varepsilon(1)} + a_2X_{\varepsilon(2)} + \cdots + a_nX_{\varepsilon(n)});$$

$$\sum_{\substack{i>k \\ j\neq i,\ j\neq k}} (X_i + X_k - X_j)^2 \quad (n \text{ quelconque}).$$

La diversité des notations Σ s'adapte au besoin d'exprimer les propriétés des termes à ajouter

---

p573 L'anneau ainsi obtenu se note habituellement $K[[X]]$; au lieu de la notation initiale $f = (a_0, a_1, \ldots, a_n, \ldots)$, on représente les séries formelles par l'écriture

$$(*) \qquad f = a_0 + a_1X + \cdots + a_nX^n + \cdots = \sum_{n=0}^{\infty} a_nX^n,$$

qui permet de retenir plus facilement les formules définissant les opérations fondamentales : pour multiplier deux séries formelles, on les multiplie « terme à terme » puis on groupe ensemble les termes de même degré dans le résultat obtenu. Bien entendu, la formule (*) n'a théoriquement aucun sens, puisqu'elle peut contenir une infinité de termes non nuls; on ne doit la considérer que comme une simple *notation* commode pour représenter la suite des $a_i \in K$.

L'écriture introduite et ses manipulations acquiert progressivement un statut autonome

[Graham89]

p 23 The $\sum$ sign occurs more than 1000 times in this book, so we should be sure that we know exactly what it means. Formally, we write

$$\sum_{P(k)} a_k \tag{2.4}$$

as an abbreviation for the sum of all terms a_k such that k is an integer satisfying a given property $P(k)$. (A "property $P(k)$" is any statement about k that can either be true or false.) For the time being, we'll assume that only finitely many integers k satisfying $P(k)$ have $a_k \neq 0$; otherwise infinitely many nonzero numbers are being added together, and things can get a bit tricky. At the other extreme, if $P(k)$ is false for all integers k, we have an "empty" sum; the value of an empty sum is defined to be zero.

définition générale mais très intuitive du sens mathématique du Σ

---

p 22 The three-dots notation has many uses, but it can be ambiguous and a bit long-winded. Other alternatives are available, notably the delimited form

$$\sum_{k=1}^{n} a_k\,, \tag{2.2}$$

which is called Sigma-notation because it uses the Greek letter $\sum$ (uppercase sigma). This notation tells us to include in the sum precisely those

"Le signe $\sum_{i=1}^{i=\infty}$ indique que l'on doit donner au nombre entier i toutes ses valeurs 1, 2, 3, ..., et prendre la somme des termes."
— J. Fourier [102]

La notation et sa définition originelle

---

p22 other letter could be substituted for k here without changing the meaning of (2.2). The letter i is often used (perhaps because it stands for "index"), but we'll generally sum on k since it's wise to keep i for $\sqrt{-1}$.

It turns out that a generalized Sigma-notation is even more useful than the delimited form: We simply write one or more conditions under the $\sum$, to specify the set of indices over which summation should take place. For

Well, I wouldn't want to use a or n as the index variable instead of k in (2.2); those letters are "free variables" that do have meaning outside the $\sum$

pragmatique du choix des symboles

p24 Iverson's convention

book. The idea is simply to enclose a true-or-false statement in parentheses, and to say that the result is 1 if the statement is true, 0 if the statement is false. For example,

$$\sum_{k} a_k(P(k))\,. \tag{2.5}$$

deux extensions de la notation

---

p23 The biggest advantage of general Sigma-notation is that we can manipulate it more easily than the delimited form. For example, suppose we want to change the index variable k to $k+1$. With the general form, we have

$$\sum_{1\leqslant k\leqslant n} a_k \;=\; \sum_{1\leqslant k+1\leqslant n} a_{k+1}\,;$$

it's easy to see what's going on, and we can do the substitution almost without thinking. But with the delimited form, we have

$$\sum_{k=1}^{n} a_k \;=\; \sum_{k=0}^{n-1} a_{k+1}\,;$$

it's harder to see what's happened, and we're more likely to make a mistake.

pragmatique de l'avantage d'une notation sur une autre dans une situation donnée. (critères d'ergonomie)

---

p24 People are often tempted to write

$$\sum_{k=2}^{n-1} k(k-1)(n-k) \qquad \text{instead of} \qquad \sum_{k=0}^{n} k(k-1)(n-k)$$

efficiency of understanding! We will find it advantageous to keep upper and lower bounds on an index of summation as simple as possible, because sums can be manipulated much more easily when the bounds are simple. Indeed, the form $\sum_{k=2}^{n-1}$ can even be dangerously ambiguous, because its meaning is not at all clear when $n = 0$ or $n = 1$ (see exercise 1). Zero-valued terms cause no harm, and they often save a lot of trouble.

pragmatique d'utilisation mathématique d'une notation

Démonstration:

[Pac 86]

p 11 Il est clair que $0 \le P(B \mid A) \le 1$. Il suffit donc de vérifier la σ-additivité de la probabilité conditionnelle. Elle résulte de la σ-additivité de la probabilité P, en effet, si (B_j) est une suite d'événements incompatibles : $B_i \cap B_j = \emptyset$

on a aussi $(B_i \cap A) \cap (B_j \cap A) = \emptyset$, et donc par additivité de P, on a:

$P[(\cup B_j) \cap A] = P[\cup (B_j \cap A)] = \sum P(B_j \cap A)$ d'où

$P((\cup B_j) \mid A) = \sum P(B_j \mid A)$

Suites et sommations sont utilisées intuitivement sans devoir préciser le domaine

La variable du Σ n'a pas besoin d'être précisée lorsqu'elle est évidente pour le mathématicien

---

p 40 Loi de X discrète, loi de Y discrète: $dP(x,y) = \sum_{a,b} p_{a,b}\, d\delta_a(x)\, d\delta_b(y)$

$P(Y=b \mid X=x) = p_{x,b} / \sum_y p_{x,y}$ $\quad h(x) = \sum_y y\, p_{x,y} / \sum_y p_{x,y}$

Loi de X continue, loi de Y discrète: $dP(x,y) = \sum_b f(x,y)\, dx\, d\delta_b(y)$

$P(Y=b \mid X=x) = f_b(x) / \sum_y f_y(x)$ $\quad h(x) = \sum_y y\, f_y(x) / \sum_y f_y(x)$

Loi de X discrète, loi de Y continue: $dP(x,y) = \sum_a f(a,y)\, d\delta_a(x)\, dy$

$dP_Y(b \mid X=x) = f(x,b)\, db / \int f(x,y) dy$ $\quad h(x) = \int y f(x,y) dy / \int f(x,y) dy$

Loi de X continue, loi de Y continue: $dP(x,y) = f(x,y)\, dx\, dy$

$dP_Y(b \mid X=x) = f(x,b)\, db / \int f(x,y)\, dy$ $\quad h(x) = \int y\, f(x,y)\, dy / \int f(x,y)\, dy$

Dans cette notation, la variable est précisée et le domaine estomis

La sommation est utilisée comme intégrale discrète

Remarquons l'extension de l'usage de la notation "$f(x)$"

---

p 71 Soit $(X_n)_{n \in \mathbb{N}}$ une chaîne sur E. Pour que ce soit une chaîne de Markov homogène, il est nécessaire et suffisant qu'il existe une fonction $(i, j) \in E^2 \to p(i, j)$ telle que $\forall n \in \mathbb{N}$ $\forall x_0, \ldots, x_n$ éléments de E,

$$P(X_0 = x_0, X_1 = x_1, \ldots, X_n = x_n) = P(X_0 = x_0)\, p(x_0, x_1) \ldots . p(x_{n-1}, x_n) \qquad (14)$$

Remarques :

1) la proposition $p(x_n, x_{n+1}) = P(X_{n+1} = x_{n+1} \mid X_n = x_n)$

utilisée dans la réciproque peut aussi être établie comme suit :

$P(X_{n+1} = x_{n+1} \mid X_n = x_n) = \sum P(X_0 = x_0, \ldots, X_{n+1} = x_{n+1} \mid X_n = x_n)$

(où la sommation porte sur $(x_0, \ldots, x_{n-1}) \in E^n$)

$$= \frac{\sum P(X_0 = x_0, \ldots, X_n = x_n, X_{n+1} = x_{n+1})}{\sum P(X_0 = x_0, \ldots, X_n = x_n)}$$

$$= \frac{\sum P(X_0 = x_0)\, p(x_0, x_1) \ldots . p(x_{n-1}, x_n)\, p(x_n, x_{n+1})}{\sum P(X_0 = x_0)\, p(x_0, x_1) \ldots . p(x_{n-1}, x_n)}$$

$= p(x_n, x_{n+1})$

manipulation de Σ à variables et domaine fixés : seuls le Σ et le terme principal subsistent.

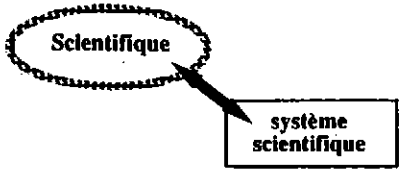
pour éviter l'ambiguïté les variables sont précisées une fois pour toute dans le texte

Les opérations mathématiques utilisées ici sont celles qui sont indépendantes des bornes

Intro

LE PROBLEME

Peut-on fournir des outils informatiques adaptés au travail scientifique ?

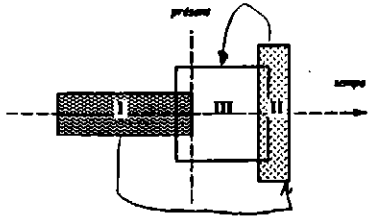


☛ **Pourquoi les mathématiques ?**

- machine universelle
- maturation millénaire
- bien formalisée ?!
- l'un des défis de l'IA
- etc.

Plan

NOTRE APPROCHE



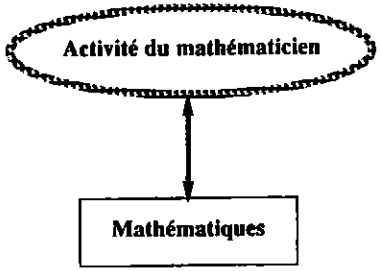
I) Quelle est la nature de l'activité mathématique ?

II) Que peut-on souhaiter ?

III) Que peut-on proposer ?

I

L'ACTIVITÉ MATHÉMATIQUE



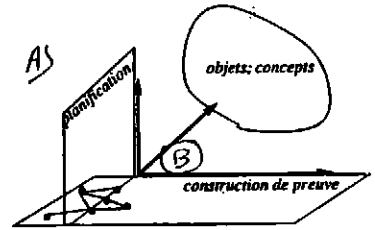
Je dirai que j'ai trouvé la démonstration de tel théorème dans telles circonstances ; ce théorème aura un nom barbare, que beaucoup d'entre vous ne connaîtront pas ; mais cela n'a pas d'importance : ce qui est intéressant pour le psychologue, ce n'est pas le théorème, ce sont les circonstances.

Henri POINCARÉ

I

L'ACTIVITÉ MATHÉMATIQUE

A) Recherche de preuve
B) Activité de communication



A-52

I

ACTIVITÉ ET DÉMONSTRATIONS

pour achever la démonstration il reste donc à vérifier la relation

$$\binom{n}{r} = \binom{n-1}{r} + \binom{n-1}{r-1}.$$

Or le second membre est égal à

$$\frac{(n-1)\,(n-2)\ldots(n-r)}{1.2\ldots r} + \frac{(n-1)\,(n-2)\ldots(n-r+1)}{1.2\ldots(r-1)}$$

$$= \frac{(n-1)\ldots(n-r+1)\,(n-r) + (n-1)\ldots(n-r+1)\,r}{r!}$$

$$= \frac{[(n-r)+r]\,(n-1)\ldots(n-r+1)}{r!} = \frac{n(n-1)\ldots(n-r+1)}{r!} = \binom{n}{r},$$

ce qui achève la démonstration.

Revue Géométrie n 145

I

ACTIVITÉ ET DÉMONSTRATIONS

$$\frac{(n-1)\,(n-2)\ldots(n-r)}{1.2\ldots r} + \frac{(n-1)\,(n-2)\ldots(n-r+1)}{1.2\ldots(r-1)}$$

$$\frac{(n-1)(n-2)\ldots(n-r) + (n-1)(n-2)\ldots(n-r+1)\,r}{1.2\ldots r}$$

I

ACTIVITÉ ET DÉMONSTRATIONS

$$\frac{(n-1)\,(n-2)\ldots(n-r)}{1.2\ldots r} + \frac{(n-1)\,(n-2)\ldots(n-r+1)}{1.2\ldots(r-1)}$$

$$= \frac{(n-1)\ldots(n-r+1)\,(n-r) + (n-1)\ldots(n-r+1)\,r}{r!}$$

$$= \frac{\qquad\qquad (n-r) + \qquad\qquad r}{r!}$$

- Structure prégnante
- Zones d'intérêt
- Micro-tâches

I

ACTIVITÉ ET DÉMONSTRATIONS

$$= \frac{\qquad\qquad (n-r) + \qquad\qquad r}{r!}$$

$$= \frac{(n-r+r)\qquad\qquad}{\qquad\qquad} = \frac{n\qquad\qquad}{\qquad\qquad}$$

I — L'ACTIVITÉ DU MATHÉMATICIEN

Un constat

- des démonstrations jetées à la louche
- une structure de surface
- une structure profonde

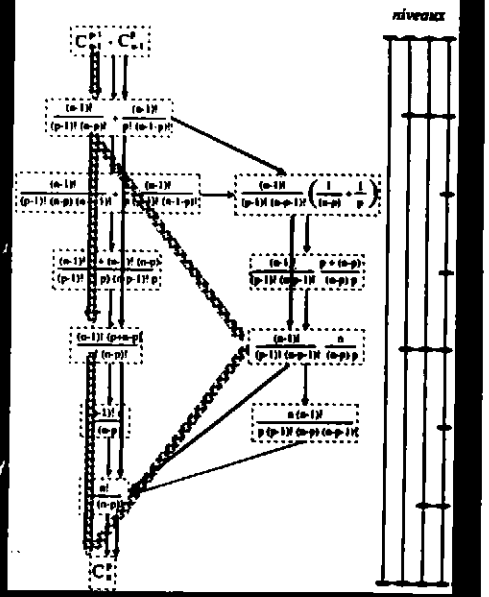
Contenu du texte

- des calculs
- des actions de présentation visuelle
- des procédures basées sur la forme et le contenu

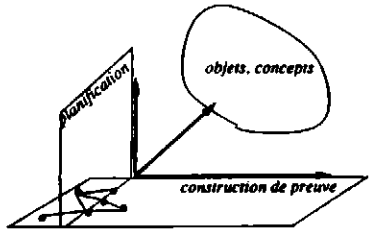
Bilan

- nécessité de faire apparaître la structure profonde
- étudier et modéliser les tâches et les procédures

I — PREUVE A GRANULARITÉ VARIABLE



I — L'ACTIVITÉ MATHÉMATIQUE

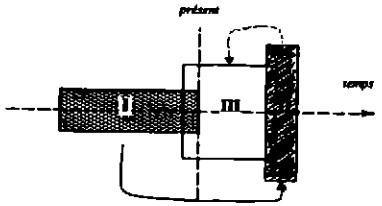


DE SURFACE	PROFOND
énoncé démonstration schéma de présentation	problème preuve plan de résolution
syntaxe d'inférence marqueurs d'inférence usage du langage	inférence effective étapes d'inférence usage des concepts
procédures visuelles	procédures mentales

Fin de la partie I

A 53

Plan — NOTRE APPROCHE D'ASSISTANCE



I) Quelle est la nature de l'activité mathématique ?

II) Que peut-on souhaiter ?

III) Que peut-on proposer ?

papier/crayon → informatique

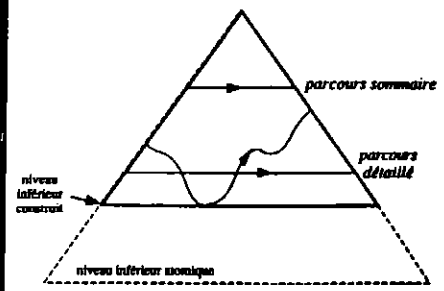
II — Ami : Assistant mathématicien intelligent

Aspirations du mathématicien

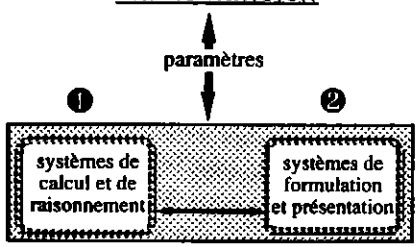
- support "intelligent"
- déléguer les tâches ingrates
- diminuer le temps perdu pour des tâches secondaires

Notre étude

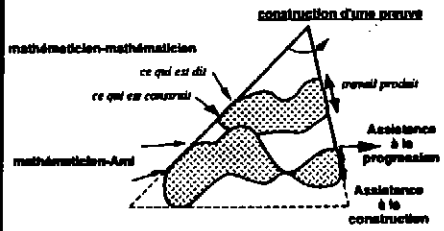
assistance à la construction de preuves et démonstrations



II — SYSTEMES EXISTANTS ET MAO



II — PROBLÉMATIQUE DE L'ASSISTANCE



Assistance à la progression

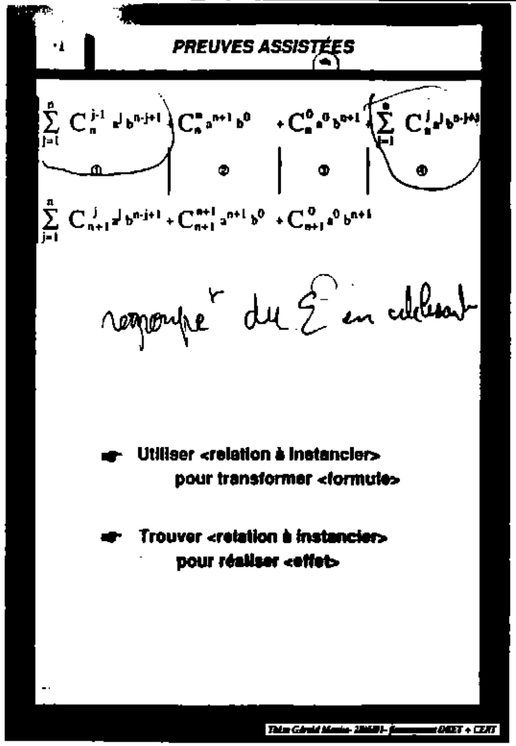
- connaître les développements standard
- comprendre un plan à demi-mots
- trouver une suite évidente

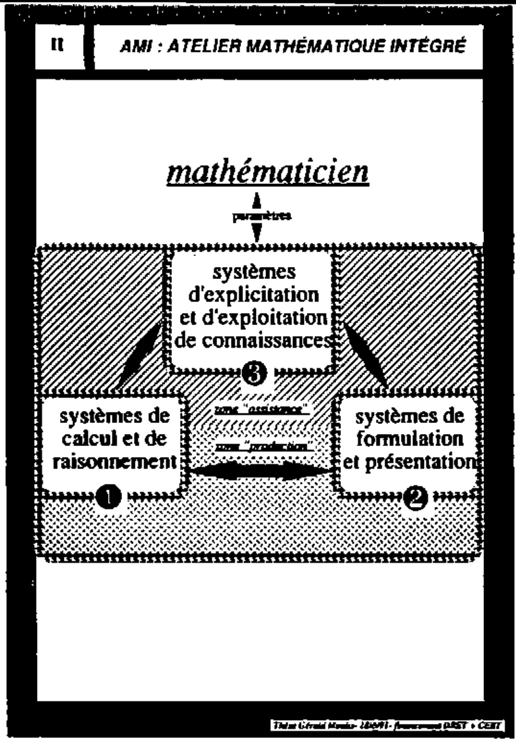
Assistance à la construction

- connaître les opérations de construction de preuve
- comprendre une opération à demi-mots
- entreprendre les inférences nécessaires

Démonstration par récurrence de la formule du binôme

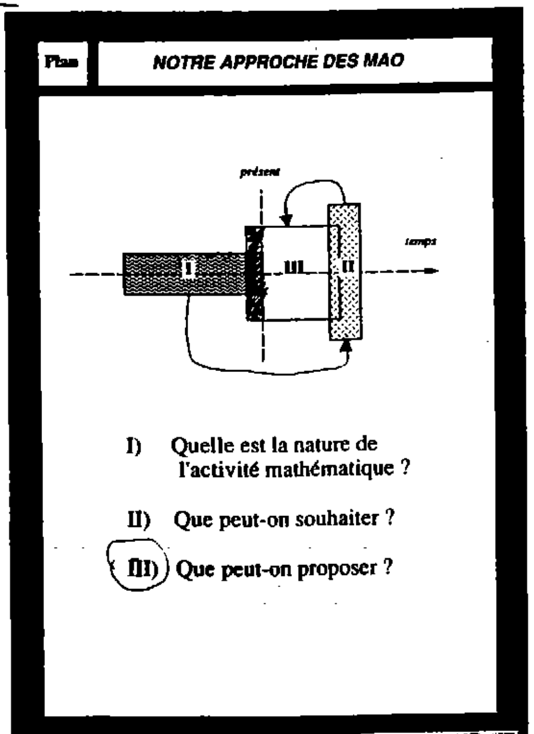
$$\sum_{j=1}^{n+1} C_{n+1}^{j} a^{j} b^{n+1-j}$$

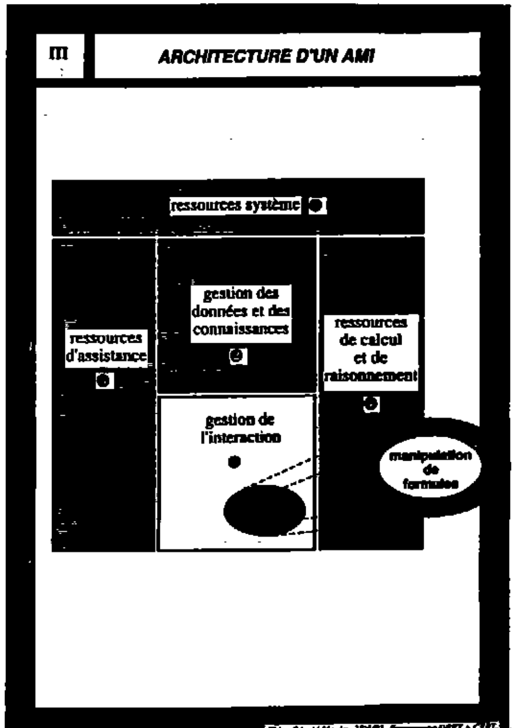


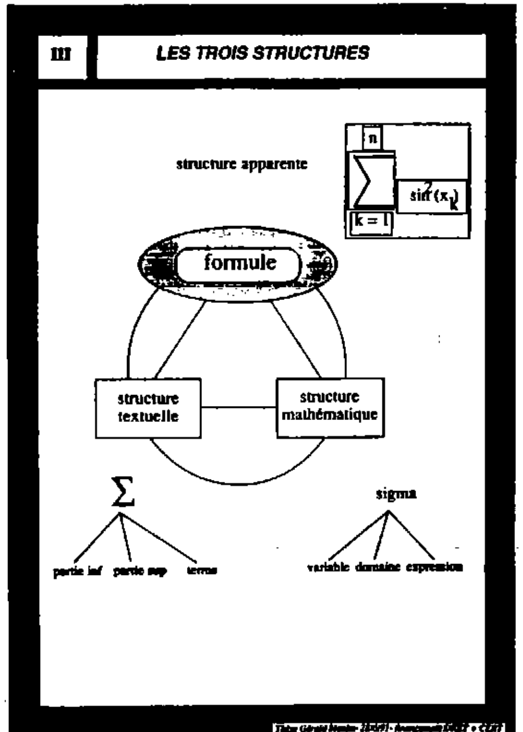


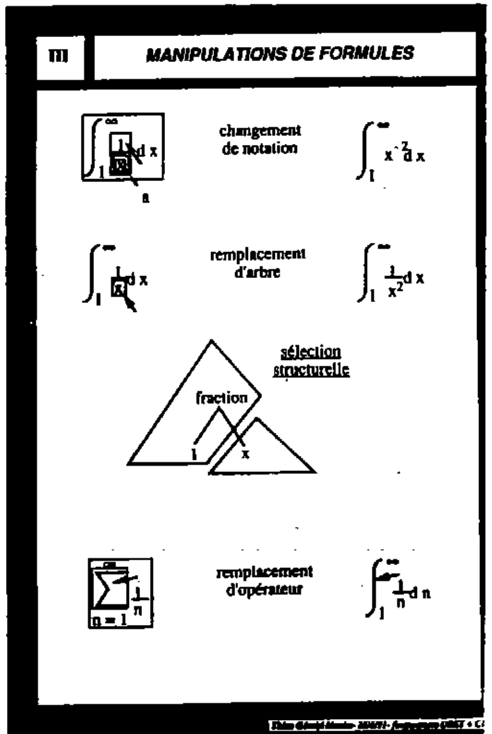
Fin de la partie II

A-54

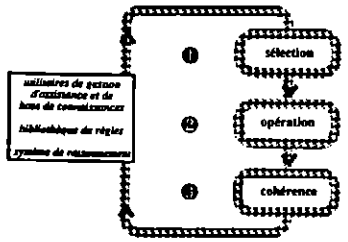




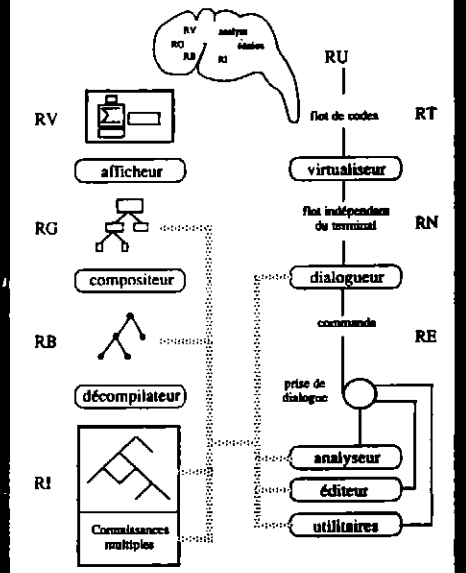




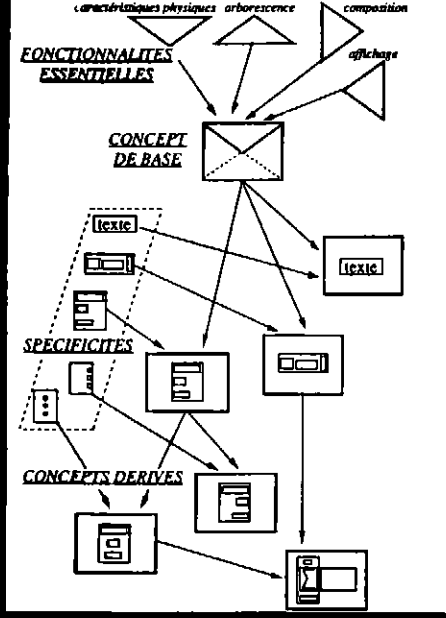
III **CYCLE DE MANIPULATION**



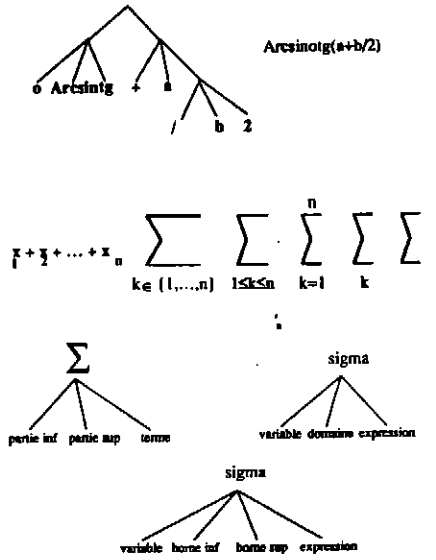
III **SCHÉMA DE LA BOUCLE D'INTERACTION**



III **LES BOITES**



III **ANALYSE DES FORMULES**



A-55

$\sum_{i \in I} a_i$ .....(1) (p.107)

$\sum_{i=1}^{n} a_i$ .....(2) (p.108)

$\sum_{1 \le i \le n} a_i$ .....(3) (p.108)

$\sum_{i} a_i$ .....(4) (p.275)

$\sum a_i$ .....(5) (p.275)

$\sum_{i \in I} a_i$ .....(6) (p.560)

$P(i)$
$Q(i)$

$\sum_{P(i)} a_i$ .....(7) (p.607)

$Q(i)$

$\sum_{i=0}^{\infty} a_i X^i$ .....(8) (p.573)

III **LE POINT SUR UN AMI**

Des résultats modestes

- manipulations de formules
- AMI : Atelier Mathématique Intégré
- MAO : Mathématiques Assistées par Ordinateur

Travers constatés

- raisonnement mathématique = logique
- nature, diversité et quantité de connaissances sous-estimées
- activité et utilisation des connaissances méconnues

Approche intercom

- analyse d'une activité conceptuelle via le langage
- transcription en une activité "naturelle" d'assistance
- développement expérimental du système d'assistance

Fin de la partie III

Fin! **BILAN & PERSPECTIVES des MAO**

Bilan

- Analyse de l'activité mathématique
- Modèles pratiques :
  - preuve à granularité variable
  - trois structures des formules
  - etc.
- Approche intercom
- Linguistique des mathématiques

Perspectives

- mathématiques pour constituer des modèles informatiques
- création de bases de connaissances mathématiques
- compréhension de documents mathématiques
- production de raisonnements et de démonstrations
- Assistance au travail scientifique

Assistance au mathématicien

assistance au travail scientifique 21, 249, 272, 347, A-52
Assistant mathématicien intelligent (Ami) 138, 180, 341, A-53 *(les numéros A-i référencent les pages des Annexes)*
assist. à la construction de preuves 141, 156+, 163+, A-53
assistance à la construction de démonstrations 163+, A-54
guidage des tâches 170+, 338
modèle d'assistance adapté à l'activité natuelle 177+
approche Intercom 182+, 342, 344, 348+, A-52, A-55
MAO (cf rubrique)
approche d'assistance via le langage et les tâches élémentaires 343, 347, A-52

Activité mathématique

distingue entre activité et mathématiques 5, 341, A-52
partition paramathématique / purement mathématique 9
processus perceptifs 10, 55, 167, A-11+, A-52
processus créatifs 10, 58, 64, A-13
processus cognitifs 10, 61
phases et tâches 11+, 310
modèle interactif de résolution de problèmes 16, 249
conduite du raisonnement 79
inférences et mise en œuvre de règle de déduction 88, A-52
plan, stratégie, schéma, ordonnancement 104, 321+, A-19
opération de déduction 106, 144+, 150+, A-52
étude expérimentale 175+, 344, 348+

Preuves et démonstrations

distinguo 9, 118, 119, 165, A-53
construction d'une preuve 14, 130, 141+, 338, A-52
axes de construction d'une preuve 18+, 291+, 298, A-52
construction de démonstrations 59+, 141+, A-54
démonstrations et processus heuristiques 61, 65
structure apparente / profonde 85+, A-53
voies possibles dans une preuve 113+, A-53
habillage d'une preuve 116, 132, 145, 336, A-17, A-21+
preuve à granularité variable 120+, A-53
niveaux dans une preuve 122+, A-53
problème et contexte dans une preuve 127, 303+, A-18+
généralisation et réutilisation 133, 201, 207, 236+, 243+, A-13
récurrence 147+, A-24+
trouver / vérifier une preuve 199

Ergonomie et ergonomie cognitive

les ... et,ou la reconnaissance de "formes symboliques" 49, 94+, 108+, 168, 285+, 330+, 335+, A-11+, A-52
ergonomie du langage 51+, 118, A-14+
ergonomie et choix de notations 56+, A-15+, A-37
ergonomie et organisation des connaissances 62+
ergonomie et pédagogie 91
représentations intuitive / formelle 94+
capacités de perception et de calcul 100, 118, 321+
micro-tâches, micro-opérations 106, A-52
modèle d'analyse des formules (structure prégnante, zone d'intérêt) 107+, 321+, 330+, A-52
ergonomie et qualité d'interaction 151, 206, 219, 233, 343+, 349
visualisation conceptuelle ou visualétique 180
ergonomie et interfaces 254+

Langage Mathématique

terminologie de la représentation du langage 25
approche logique du langage 26+
modélisation du langage 32, 67+, 271, 291, 303+, A-1+
pragmatique 33, 158+, 161+, 169+, 213
fonctions du langage 40, 44, 292, 298+
élémentaire / macroscopique 41, 299
une typologie des exemples 48
langages de Formules, Phrases et Croquis 45, 308
langages de Figures et typologie des Figures 45+
rôle du langage 50, 102, 120
linguistique des mathématiques :
. enjeux et développements 73, 341, 343
. analyse du langage 71+, 118
. niveaux de discours 74+, 293
. analyse d'exemples 79+ (parabole), 94+ (binôme), 104+ (binôme), 142+ (Gauss), 171+ (Σ), 286 (Σ), 303+ (couple), 321+ (Σ), A-48 (Σ)
traitement du Langage Mathématique (multi-langage) 271+, 277+, 284+, 289, A-33
adaptation du langage 271+, 286, 308+, A-34+

Connaissances mathématiques

modélisation et réseau de concepts 81+, 279+ 296+, A-48
diversité des connaissances 90+, 291
connaissances pragmatiques 90, 95, 164, 169, 205, 219, 234, 249, 259+, 272, 286, 290, 309, A-1, A-7+, A-48+
expertise de présentation 109, A-1+, A-48+
langage et connaissances 140
connaissances de surface / profondes 193
organisation des connaissances mathématiques 279, 296+
les structures de la sommation 285+, A-48+, A-55
dualité entre acquisition et traitement 289, 332+
partition et exploitation des connaissances 292+
classification et principe de généralisation 300+
la notion de définition 303+, A-48+
variables et informations incomplètes 314+
rattachement et exploitation des propriétés 316+
identités remarquables et expertise d'utilisation 326+, 335+

Mathématiques Assistées par Ordinateur (MAO)

formulation du besoin d'un Atelier Mathématique Intégré (AMI) 184, 214, 341+, 347, A-54
panorama des MAO : objectif et champs d'action 189+, 345+
conception des systèmes informatiques 192+
systèmes de calcul formel 195+, A-26
production de raisonnements 199+, 343
tuteurs intelligents comme source d'innovation 203
analogie entre mathématiques et programmation 203
maîtrise du savoir-faire 206
hypermédia et métamédia 207+
représentation et production de démonstrations 212
limites, défis et perspectives des MAO : 214, 343+, 345+, A-55
abord et pièges des MAO 217, 341
fonctionnalités et architecture d'un AMI 249+, A-54
conception des interfaces 254+

Manipulations de formules

expertise d'analyse et d'utilisation de formules 174, 325+
outils génériques de manipulation de structure 192
limites/ besoins des traitements de textes 204+, 239, 342+
SOFTMATH 217+, 234, 246, 252, 342
modèle des trois structures couplées 221, 257, A-54
cycle de manipulation de formules 222, 231+, A-55
exemples de manipulations de formules 223+, A-27
désignation, translation, sélection 226+, 231
style de présentation 228+
taxonomie des opérations disponibles 232
apprentissage par manipulation directe 236+
cycle de vie et gestion d'une formule 256+, 263+, A-55
boîtes et abstractions mathématiques 264+, A-29+, A-55
analyse informatique des formules 275+, A-39+, A-55

TABLE DES MATIERES

Plan

Introduction xi

PARTIE I : Le Langage Mathématique

1 ..... Le travail du mathématicien 3
1.1 ..... Notre approche des mathématiques 3
1.1.1 ..... Le langage en Mathématiques 3
1.1.2 ..... Regards sur l'activité mathématique 4
1.1.3 ..... Quelques approches de l'activité mathématique 5
1.1.4 ..... Une étude des Mathématiques à partir du langage 7
1.2 ..... Une classification des activités du mathématicien 8
1.2.1 ..... Caractérisation de l'activité mathématique 8
1.2.2 ..... Description des processus mentaux du mathématicien 10
1.2.3 ..... Les phases de l'activité mathématique 11
1.2.4 ..... Typologie des tâches de l'activité mathématique 12
1.3 ..... L'activité mathématique dans sa phase d'utilisation 14
1.3.1 ..... La gestion de la construction d'une preuve 14
1.3.2 ..... Un modèle interactif de résolution de problèmes 16
1.3.3 ..... Les dimensions de la construction d'une preuve 18

2 ..... Introduction à la modélisation du langage 21
2.1 ..... Regards sur les objets mathématiques courants 21
2.1.1 ..... Deux usages de l'informatique 21
2.1.2 ..... Rappels classiques sur la définition d'un langage 21
2.1.3 ..... Plusieurs sortes de représentation pour les formules 23
2.1.4 ..... Terminologie et représentation du langage 25
2.2 ..... La formalisation logique du langage 26
2.2.1 ..... L'approche logique 26
2.2.2 ..... L'interprétation des phrases 28
2.2.3 ..... Formaliser les déductions 29
2.2.4 ..... Apport de la logique aux mathématiques 30
2.3 ..... Limites de la logique pour modéliser l'activité mathématique 32
2.3.1 ..... Modélisation du langage et monde de référence 32
2.3.2 ..... La logique et la résolution de problèmes 34
2.3.3 ..... Des procédés déductifs différents 35
2.3.4 ..... Une argumentation inexistante 36

3 ..... Regards sur l'usage du Langage Mathématique 39
3.1 ..... L'étude du Langage Mathématique 39
3.1.1 ..... L'outil de pensée du mathématicien 39
3.1.2 ..... Partition du Langage Mathématique 40
3.1.3 ..... La distance élémentaire-macroscopique 41
3.1.4 ..... Les fonctions du Langage Mathématique 44
3.2 ..... Les langages de Figures 45
3.2.1 ..... Les voies de communication de l'activité mathématique 45
3.2.2 ..... L'importance des Figures 46
3.2.3 ..... Typologie des Figures 47
3.2.4 ..... Le rôle des Figures 48
3.2.5 ..... Moyens de traitement des Figures 49
3.3 ..... Les rôles du langage pour le mathématicien 50
3.3.1 ..... Les langages de Phrases et de Formules 50
3.3.2 ..... Les processus perceptifs 55
3.3.3 ..... Les processus créatifs dans une démonstration 58
3.3.4 ..... Les processus cognitifs 61
3.3.5 ..... Les processus créatifs dans l'activité de raisonnement 64
3.4 ..... Synthèse de ce panorama du Langage Mathématique 67

4 Analyse linguistique d'écrits mathématiques 71
4.1. Vers une linguistique des mathématiques 71
4.1.1 Analyse linguistique des textes de démonstrations 71
4.1.2 Notre objectif pratique d'analyse du langage 73
4.1.3 Les niveaux de discours des textes mathématiques 74
4.1.4 Une approche d'analyse 75
4.2. Analyse d'un texte de résolution de problème 79
4.2.1 La modélisation des données 79
4.2.2 La structure d'une démonstration 85
4.2.3 Synthèse de l'analyse du document 90
4.3. Analyse directe de textes d'un ouvrage mathématique 94
4.3.1 L'interprétation des écritures 94
4.3.2 La maîtrise des écritures 95
4.3.3 Diversité des représentations utilisables dans une argumentation 96
4.3.4 Compétences et connaissances du lecteur 99
4.3.5 Rôles possibles d'une démonstration 102
4.4. Analyse de quelques étapes de manipulation de formules 104
4.4.1 Production du plan d'une démonstration 104
4.4.2 Le raisonnement lors du déroulement des calculs 105
4.4.3 Présentation des étapes de démonstration 107
4.4.4 Le suivi des étapes de démonstration 109
4.5. La production de preuves et démonstrations 110
4.5.1 La conduite des calculs 110
4.5.2 La détermination des étapes de preuve 112
4.5.3 Plusieurs voies possibles dans une preuve 113
4.5.4 Des parcours selon plusieurs niveaux de détail 115
4.5.5 Habillage d'une preuve en démonstration 116
4.6. Synthèse de notre analyse du langage 118

5 L'usage des preuves mathématiques 119
5.1. L'intérêt de preuves à granularité variable 119
5.1.1 La diversité des preuves possibles 119
5.1.2 Nos objectifs de modélisation d'une preuve 120
5.1.3 Un modèle de preuve à granularité variable 121
5.2. La construction de preuves à granularité variable 123
5.2.1 Les segments de preuve à granularité variable 123
5.2.2 Les niveaux de décomposition 125
5.2.3 Le parcours effectif d'une preuve 126
5.3. La résolution des problèmes mathématiques 127
5.3.1 Problèmes et contextes dans une preuve 127
5.3.2 Les procédés de construction de preuve 130
5.3.3 L'exploitation des preuves construites 132

PARTIE II : Les Mathématiques Assistées

1 L'assistance au mathématicien 137
1.1. Le mathématicien face à l'informatique 137
1.2. Introduire un Assistant mathématicien intelligent 138
1.3. Notre choix de construction de preuves et démonstrations 139
1.4. Le langage et les connaissances 140

2 Étude et modélisation de preuves 141
2.1. La construction d'une preuve à partir d'une démonstration 141
2.2. Étude de la construction de preuves et démonstrations 142
2.3. Les causalités qui régissent les opérations de déduction 144
2.4. Une démonstration par récurrence 147
2.5. Les opérations mathématiques élémentaires 150

3 Assistance lors de la construction de preuves 153
3.1. La démonstration de la formule du binôme 153
3.2. La production d'une transformation de formule complexe 155
3.3. La construction par isolement-regroupement 157
3.4. Les facilités de commande 158
3.5. Pragmatique de la présentation 161

4 ..... Fonctionnalités d'un système d'assistance 163
4.1 ..... La construction du texte de démonstration 163
4.1.1. ..... Production du contenu mathématique 163
4.1.2. ..... Définition de la structure interne d'une démonstration 165
4.1.3. ..... Assistance à la construction d'une démonstration 166
4.2 ..... Fonctionnalités de perception 167
4.2.1. ..... Un modèle de traitement perceptif 167
4.2.2. ..... Application à la comparaison de formules 168
4.2.3. ..... Pragmatique et recueil d'informations implicites 169
4.3 ..... Le guidage des tâches 170
4.4 ..... La production d'une preuve complexe 172

5 ..... Problématique de l'assistance au travail 175
5.1 ..... Une assistance adaptée à l'activité 175
5.1.1. ..... L'étude expérimentale de l'activité mathématique 175
5.1.2. ..... Un environnement de travail naturel 176
5.1.3. ..... Les besoins de définition d'un modèle d'assistance 177
5.1.4. ..... Un modèle d'assistance adapté à l'activité 177
5.2 ..... Les principes de mise en œuvre d'un modèle d'assistance 178
5.2.1. ..... Le paradigme d'Engagement direct pour l'interaction 178
5.2.2. ..... Un module Ami comme assistant 180
5.2.3. ..... La visualisation de propriétés conceptuelles 180
5.3 ..... La conception d'un système d'assistance 182
5.3.1. ..... L'approche Intercom 182
5.3.2. ..... Spécificités de l'approche Intercom 183
5.3.3. ..... Vers un Atelier Mathématique Intégré 184
5.3.4. ..... Discussion de l'approche Intercom 185

PARTIE III : Les Techniques Informatiques

1 ..... État des lieux des systèmes informatiques 189
1.1 ..... Introduction aux Mathématiques Assistées par Ordinateur 189
1.1.1. ..... Les objectifs du panorama des systèmes existants 189
1.1.2. ..... Topologie des MAO 190
1.1.3. ..... Présentation de l'évolution des systèmes informatiques 192
1.1.4. ..... Typologie des techniques de conception informatique 194
1.2 ..... Les outils pour raisonner et calculer 195
1.2.1. ..... Les systèmes de calcul 195
1.2.2. ..... Les systèmes de production de théorèmes 198
1.2.3. ..... Les systèmes experts en raisonnement 199
1.2.4. ..... Les systèmes qui découvrent ou apprennent 201
1.2.5. ..... Les systèmes d'enseignement et de modélisation cognitive 202
1.3 ..... Les outils de formulation et de présentation 203
1.3.1. ..... Comparaison entre mathématiques et programmation 203
1.3.2. ..... Les systèmes de traitement de texte et de formules 204
1.3.3. ..... Les systèmes d'aide, de documentation, d'archivage et de dialogue 207
1.3.4. ..... Les systèmes de brouillon, de gestion d'idées et de visualisation 211
1.3.5. ..... Les systèmes de représentation et de production de démonstrations 212
1.3.6. ..... Les systèmes de modélisation de problèmes spécialisés 213
1.4 ..... Limites de l'existant 214

2 ..... Les manipulations de formules 217
2.1 ..... SOFTMATH et les manipulations de formules 217
2.1.1. ..... Nos objectifs pour les manipulation de formules 217
2.1.2. ..... Notre approche pour SOFTMATH 219
2.1.3. ..... Un modèle de formules basé sur trois structures couplées 220
2.1.4. ..... Le cycle de manipulation de formules 221
2.2 ..... Typologie des manipulations de formules 223
2.2.1. ..... Exemples de manipulations 223
2.2.2. ..... Diverses possibilités d'opérer 225
2.2.3. ..... Les manipulations de construction de formule 226
2.2.4. ..... Les manipulations d'usage mathématique 229
2.3 ..... Synthèse des fonctionnalités de manipulation observées 230
2.3.1. ..... Description des modes de sélection 230
2.3.2. ..... Taxonomie des opérations disponibles 232

2.3.3 La phase de maintien de la cohérence 233
2.3.4 Récapitulatif des fonctionnalités de manipulation 234
2.4. La description des intentions par l'exemple 235
2.4.1 Présentation de mécanismes originaux 235
2.4.2 Apprentissage par manipulation directe 237
2.4.3 La généralisation d'une transformation 238
2.4.4 Expressivité des manipulations proposées grâce aux trois structures 240
2.4.5 Une sémantique des manipulations en vue de leur généralisation 241
2.5. L'usage de généralisations en mathématiques 243
2.5.1 Présentation des généralisations possibles 243
2.5.2 Des critères de décision pour une généralisation sémantique 243
2.5.3 Influence des manipulations sur les généralisations 245
2.5.4 Utilisation pratique des manipulations généralisées 246

3 Modèles informatiques pour une mise en œuvre ergonomique 249
3.1. Les fonctionnalités d'un Atelier Mathématique Intégré 249
3.1.1 Notre proposition d'intégration du cycle de conception 249
3.1.2 Notre approche Intercom vers un AMI 251
3.1.3 L'architecture structurelle d'un AMI 253
3.1.4 Utilisateurs et conception des interfaces 254
3.2. Gestion des manipulations de formules 256
3.2.1 Les opérations du système sur les formules 256
3.2.2 La saisie des formules 258
3.2.3 Gestion des formules au sein de l'interface 261
3.2.4 Les modules de la gestion des formules 263
3.3. Un modèle informatique de visualisation structurée et interactive 264
3.3.1 L'introduction d'abstractions mathématiques 264
3.3.2 Les langages à objets 265
3.3.3 Représentation du concept de boîte 266
3.3.4 L'extension du code 268

4 Le traitement du Langage Mathématique 271
4.1. La prise en compte du Langage Mathématique 271
4.2. L'analyse des formules 273
4.2.1 Présentation de l'analyse 273
4.2.2 Notre approche de l'analyse 274
4.2.3 L'état des lieux actuel 276
4.2.4 Les besoins spécifiques aux mathématiques 277
4.3. La gestion interne des formules 278
4.3.1 Une organisation mathématique "à objets" 278
4.3.2 Organisation de la gestion interne 280
4.3.3 Constitution de la représentation interne des formules 281
4.3.4 La liberté de choix des structures 282
4.4. Les mécanismes de génération 284
4.5. Le cas de la sommation 285
4.5.1 Les formes de structure textuelle 285
4.5.2 Structures mathématiques pour la sommation 287
4.5.3 Les correspondances entre structures 288
4.6. Bilan sur le traitement du langage 289

5 Organisation des connaissances 291
5.1. Introduction à la modélisation des connaissances mathématiques 291
5.1.1 La constitution de connaissances mathématiques 291
5.1.2 L'utilisation des connaissances 291
5.1.3 L'action du mathématicien sur les connaissances 293
5.1.4 Les connaissances qui nous intéressent 294
5.1.5 Les causes de limitation du modèle des connaissances disponibles 295
5.2. Nos propositions d'organisation 296
5.2.1 L'organisation autour des concepts 296
5.2.2 Partition des connaissances de traitement 298
5.2.3 La classification des connaissances 300
5.2.4 Les structures d'organisation 301
5.3. Une analyse des connaissances : la notion de définition 303
5.3.1 Les dimensions d'analyse des définitions mathématiques 303
5.3.2 Le texte d'une définition 305
5.3.3 Introduction d'une construction mathématique 306
5.3.4 Les conséquences dans une théorie ensembliste intuitive 309

5.4 ..... *Les structures d'organisation de la représentation* 311
5.4.1. ..... Les langages à objets pour la représentation des connaissances 311
5.4.2. ..... Les fonctions des manipulations ensemblistes 312
5.4.3. ..... Les manipulations d'éléments 314
5.5 ..... *L'exploitation des connaissances* 315
5.5.1. ..... Finalités de l'organisation 315
5.5.2. ..... Le rattachement des propriétés 316
5.5.3. ..... L'exploitation de propriétés 317

6 ..... Opérationnalisation des formules 321
6.1 ..... *L'usage des formules et des identités* 321
6.1.1. ..... Un exemple d'utilisation typique en mathématiques 321
6.1.2. ..... Une démarche de raisonnement générale 323
6.1.3. ..... Besoins élémentaires des Mathématiques Assistées par Ordinateur 325
6.2 ..... *L'analyse d'une identité* 326
6.2.1. ..... La nature d'une identité 326
6.2.2. ..... L'état des lieux 327
6.2.3. ..... Les possibilités d'utilisation 328
6.2.4. ..... Notre proposition d'analyse 330
6.3 ..... *L'exploitation d'identités* 332
6.3.1. ..... Acquisition d'une nouvelle identité 332
6.3.2. ..... L'utilisation d'une identité pour les transformations 334
6.3.3. ..... Conclusion pour l'exploitation des formules 337

FINAL

Conclusion 341
1. ..... *Une étude de l'activité mathématique* 341
2. ..... *Présentation chronologique* 342
3. ..... *Notre contribution vers un AMI* 343
4. ..... *Défis et perspectives des MAO* 345
5. ..... *Bilan* 348

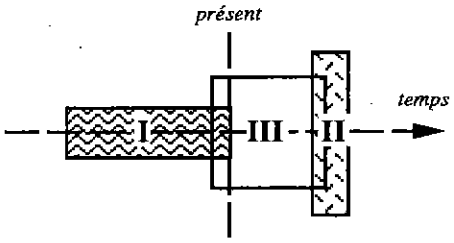
Références bibliographiques. 351

Descriptif des Annexes 359

Index thématique ..... **en marque page**

Table des matières ..... **en fin du document**

RÉSUMÉ : Cette thèse étudie les spécifications d'un système informatique d'**assistance au mathématicien** s'inspirant de l'activité mathématique "naturelle". La démarche est structurée relativement aux trois questions suivantes :



I) Quelle est la nature de l'activité mathématique ?

II) Que peut-on souhaiter pour un tel système ?

III) Que peut-on proposer ?

La partie I élabore des modèles cognitifs, se situe en complément du raisonnement heuristique, analyse le Langage Mathématique et préconise le développement d'une **linguistique des mathématiques**. Elle étudie des textes de démonstration, et les structure notamment à partir de la notion de preuve à granularité variable.

Le cas apparemment simple des **manipulations de formules** est alors étudié en partie II afin de réaliser un Assistant mathématicien intelligent (Ami), chargé de tâches élémentaires dans un Atelier Mathématique Intégré (AMI).

La partie III débute par un état des lieux des Mathématiques Assistées par Ordinateur (MAO). Elle propose un modèle de formules basé sur **trois structures couplées** (apparente, textuelle et mathématique) pour enrichir l'interaction de manipulation de formules. Elle étudie l'analyse et la génération de formules pour le cas du "sigma", et montre la complexité des connaissances mathématiques à fournir. Elle présente enfin les mécanismes envisagés pour l'utilisation d'identités remarquables.

Cette thèse illustre une approche plus générale :
I) analyse d'une activité conceptuelle via le langage ;
II) transcription en une activité "naturelle" d'assistance par un système ;
III) développement expérimental ascendant de ce système.

Le rôle de l'homme est prédominant dans cette approche dite Intercom, comme il l'est également dans la problématique générale de l'**Assistance au Travail Scientifique**. Elle est validée dans cette thèse par un jeu de modèles et de scenarii.

MOTS CLÉS : *assistance, ergonomie, activité mathématique, Langage Mathématique, linguistique des mathématiques, preuves mathématiques, manipulation de formules, atelier mathématique.*

ABSTRACT: This thesis studies some specifications of a computer system which **assists the mathematician**. This system is inspired by the "natural" mathematician's activity. Our approach is structured according to the three following questions:
I) What is the nature of the mathematical activity?
II) What could we wish for such a system?
III) What could we propose?

We elaborate in part I some cognitive models and situate our work in complement of heuristic reasoning. We analyze the Mathematical Language and recommend the development of a **linguistics of mathematics**. We study some textual proofs and structure them more especially with variable granularity proof-graphs.

In part II we study the apparently simple case of **formulae manipulations**. We try to elaborate an intelligent mathematical Assistant (Ami) which manages elementary tasks in an Integrated Mathematical Workshop.

Part III starts with a state-of-the-art of Computer-Assisted Mathematics (CAMath). We propose a formulae model based on **three tied structures** (graphical, textual and mathematical) in order to improve formulae manipulation. We study the parsing and the generation of formulae for the case of "sigma". We then exhibit the complexity of the required knowledge to assist the mathematician's activity. Finally, we present the intended mecanisms for using mathematical identities.

This thesis illustrates a more general approach:
I) the analysis of a conceptual activity from a linguistic point of view
II) its transcription in a "natural" activity assisted by a system
III) an experimental bottom-up development of this system.

In this approach that we call Intercom, the user's role is predominant as it is in the general problematic of **Scientific Work Assistance**. Intercom has been validated through a set of models and scenarii.

KEYWORDS: *assistance, ergonomics, mathematical activity, Mathematical Language, linguistics of mathematics, mathematical proofs, formulae manipulation, mathematical workbench.*

ECOLE NATIONALE SUPERIEURE DE L'AERONAUTIQUE ET DE L'ESPACE
10, Avenue Edouard-Belin – B.P. 4032 – 31055 TOULOUSE CEDEX
Tél. 61.33.48.48 – Télex SUPAERO 531642 F – TOULOUSE