Représentation de connaissances complexes : Un formalisme à base de rôles sémantiques

Paul Lotin

N° d'ordre : D.U. 747 Année 1995
EDSPIC : 101

Thèse

Présentée à

L'ECOLE DOCTORALE DE
L'UNIVERSITE BLAISE PASCAL
CLERMONT II

Pour obtenir le titre de

Docteur de l'Université Blaise Pascal
Spécialité Informatique

par

Paul LOTIN

Diplômé d'Etudes Approfondies en Linguistique et Informatique

Sujet de la thèse :

Représentation de connaissances complexes : Un formalisme à base de rôles sémantiques

Soutenue le 10 Novembre 1995 à l'Université Blaise Pascal

Michel Chambreuil	Directeur
Gérard Sabah	Rapporteur
Danièle Hérin Aimé	Rapporteur
Michel Schneider	Membre du Jury
Lotfi Lakal	Membre du Jury

à Nathalie
à ma famille

Remerciements

Je tiens ici à exprimer mes remerciements les plus sincères :

à Michel CHAMBREUIL, Professeur à l'Université Blaise Pascal et co-directeur du Laboratoire de Recherche sur le Langage, pour m'avoir guidé et soutenu dans mon travail. Sa grande disponibilité, son aide, ses nombreux conseils et sa patience sont pour beaucoup dans l'aboutissement de cette thèse.

à Gérard Sabah, Directeur de Recherche CNRS, pour avoir accepté d'être le rapporteur de ce travail. Je le remercie pour ses précieuses remarques .

à Danièle Hérin Aimé, Professeur à l'Université de Montpellier, qui a bien voulu apporter son jugement sur ce travail et accepté d'en être rapporteur. Ses commentaires très détaillés ont contribué à l'amélioration sensible de ce mémoire.

à Michel Schneider, Directeur du Laboratoire d'Informatique de Clermont 2 pour m'avoir accueilli dans son laboratoire et avoir accepté de participer à ce jury

à Lotfi Lakal, Professeur à l'Université Blaise Pascal pour avoir examiné mon travail et accepté d'être membre de ce jury

à l'ensemble des enseignants et chercheurs de l'équipe Langue Naturelle et Environnements Informatiques d'Apprentissage, pour nos discussions fructueuses et nos échanges productifs

à Nathalie, pour m'avoir supporté durant toute cette période et avoir sacrifié de si nombreux week-ends et soirées. Je la remercie aussi pour avoir patiemment attendu ce moment

à ma famille, pour m'avoir fait confiance

à tous mes amis, ainsi qu'à ma "belle" famille pour leurs soutiens et leurs encouragements.

Résumé

Le travail présenté dans cette thèse se situe dans le domaine de la représentation des connaissances. Il s'intègre dans le projet AMICAL (**A**rchitecture **M**ulti-agents **I**nteractive **C**ompagnon pour l' **A**pprentissage de la **L**ecture) dont l'objectif est de développer un environnement informatique d'aide à l'apprentissage de la lecture.

Cette étude est centrée sur la recherche d'une représentation sémantique de textes utilisés dans des sessions d'apprentissage. La problématique est de trouver un formalisme de représentation de connaissances capable de répondre à une situation fonctionnelle particulière tout en prenant compte de contraintes de natures différentes.

Trois possibilités nous sont offertes : retenir un formalisme de représentation de connaissances existant, faire coopérer des formalismes existants ou proposer un nouveau formalisme. C'est cette troisième solution qui a été retenue après avoir montré les difficultés rencontrées par quelques formalismes existants à remplir certaines contraintes formulées.

Nous proposons dans ce mémoire un système de représentations de connaissances, basé sur des structures de connaissances appelées DCAS (**D**escription **C**omposite **A** **S**ignification). Elles sont les unités de base du raisonnement. Ces structures reposent sur un ensemble d'entités cognitives de types différents. Nous décrivons les rôles différents joués par ces entités au sein d'une DCAS, ainsi que les capacités représentationnelles et inférentielles d'un système basé sur ces notions. Nous terminons cette étude par l'application de ce formalisme dans la problématique définie dans AMICAL.

MOTS-CLES : Représentation des connaissances, formalisme de représentations, environnement d'apprentissage, raisonnement, entités cognitives, rôles cognitifs, descriptions composites.

Abstract

The work presented in this thesis deals with knowledge representation. It forms part of the AMICAL project (Interactive Multi-agents Architecture Learning-to-Read Companion) whose purpose is the development of an computer-aided learning-to-read environment.

This study focuses on the search for a semantic representation of texts used in learning sessions. The problem is to find a knowledge representation formalism which can correspond to a particular functional situation while taking into account various kinds of constraints.

There are three possibilities: retain an existing knowledge representation formalism, make existing formalisms cooperate or propose a new formalism. The third solution has been chosen after showing the difficulties encountered by existing formalisms in meeting certain formulated constraints.

We propose, in this dissertation, a knowledge representation system based on knowledge structures called DCAS (Composite Description With Meaning). They are the basic units of reasoning. These structures are based on a set of various sorts of cognitive entities. We describe the different roles played by these cognitive entities inside a DCAS, together with the representational and inferential capacities of a system based on these notions.

We complete this study with the application of this formalism to the aims defined in AMICAL.

KEY-WORDS: Knowledge Representation, representation formalism, tutoring system, reasoning, cognitive entities, cognitive roles, composite descriptions.

SOMMAIRE

Introduction

Dans l'optique des recherches actuelles, les environnements informatiques d'aide à l'apprentissage sont des systèmes à base de connaissances [Wenger, 1987]. Les connaissances représentées correspondent à des domaines multiples, tels que, connaissances sur le domaine d'apprentissage ou connaissances pédagogiques. De même, les rôles joués par ces représentations sont eux aussi multiples : organisation des connaissances à transmettre, explicitation des expertises, gestion des dialogues pédagogiques**,** analyse des réponses, formulation de questions, réponses à des requêtes d'informations.

Le projet AMICAL[1] étudie un environnement destiné à l'aide à l'apprentissage de la lecture. La situation d'apprentissage de la lecture retenue dans une première phase du projet concerne le début de l'apprentissage.

Pour des raisons pédagogiques, dans une session de travail, les activités proposées à l'élève dans le cadre de cet environnement s'appuient sur le texte d'une histoire présenté à l'élève.

La représentation sémantique de ce texte s'inscrit dans le cadre de représentations du domaine d'apprentissage.

À partir d'éléments de la problématique rencontrée dans l'étude de l'environnement AMICAL, et de l'analyse critique de différents systèmes de représentation existants, nous proposerons les premiers éléments d'un formalisme de représentation susceptible de répondre à la problématique posée.

Pour illustrer des aspects de capacités de représentation et d'inférences qui auront été retenus et pour interroger les systèmes existants sur ces capacités, nous nous appuierons sur un domaine de connaissances particulier, celui des relations spatiales.

Dans le chapitre 1, nous présenterons rapidement le projet AMICAL et nous situerons la problématique étudiée dans cette thèse.

Le chapitre 2 sera centré sur l'analyse de trois systèmes de représentations de connaissances à travers les aspects capacité de représentation et d'inférences retenus dans la problématique. Cette analyse sera précédée par une description des recherches actuelles sur le domaine de la représentation spatiale.

---

[1] Architecture Multi-agents Interactive Compagnon pour l'Apprentissage de la Lecture

Dans le chapitre 3 nous présenterons les éléments du formalisme de représentation proposé.

Dans le chapitre 4, nous présenterons l'intégration de ce formalisme de représentation dans une problématique associée au projet AMICAL.

Chapitre 1 : Représentation sémantique dans AMICAL : la problématique retenue

Nous débuterons ce chapitre par la présentation de l'environnement AMICAL en décrivant les situations d'apprentissage visées et les différents modules qui le composent. Nous détaillerons plus particulièrement un de ses modules : le module "Résolution-de-problème". Nous situerons AMICAL par rapport aux systèmes à base de connaissances et aux systèmes multi-agents.

Après cette présentation générale, nous montrerons l'importance des connaissances sur une histoire, leurs rôles dans l'environnement AMICAL.

Enfin, nous terminerons cette partie en caractérisant la problématique à laquelle cette thèse cherche à apporter des éléments de solution.

1. L'environnement AMICAL

L'environnement AMICAL est un environnement d'aide à l'apprentissage et à l'enseignement de la lecture, dans une perspective d'individualisation de l'enseignement par le système ou par une collaboration entre le système et l'enseignant. [Chambreuil & al., 1994]

Situations d'apprentissage de la lecture

La situation d'apprentissage de la lecture retenue dans le projet correspond à un apprentissage initial, plus particulièrement en début de Classe Préparatoire, et concerne des enfants en scolarité normale, en milieu médiateur (scolaire ou associatif), c–a–d en présence d'un formateur.

L'une des étapes clés de l'apprentissage de la lecture passe par la maîtrise de la notion de mot par l'apprenant [Catach, 1988]. Cette maîtrise constitue l'un des objectifs des premières sessions de travail avec AMICAL. Le mot doit être vu comme un objet graphique particulier, composé de lettres, et comme un constituant d'objets plus complexes (phonèmes). L'enfant doit pouvoir faire la relation entre l'écrit et l'oral, établir la liaison entre écrit et signification.

Les différents types de modules de l'environnement AMICAL

L'environnement AMICAL est composé de différents **modules**[2] fonctionnels correspondant à des types spécifiques d'interaction avec l'élève :

- les modules ressources et ressources-exploration,
- les modules résolution-de-problème-d'enseignement.

Les modules ressources mettent à la disposition de l'élève des ressources à travers des activités libres. Par exemple, dans le module "**Bibliothèque**" de type ressource, le système propose un ensemble d'activités libres où l'utilisateur peut à sa guise lire un ensemble de textes et bien sûr se faire assister par l'environnement AMICAL s'il en ressent le besoin.

Les modules ressources-exploration fournissent à l'élève non seulement des ressources mais aussi un micro-monde dont les concepts primitifs (et leur combinatoire) peuvent être explorés. Par exemple, le module "**Dictionnaire-Didactique**" ([Tarby, 1989], [Cherkaoui, 1991]) regroupe des activités permettant, selon différentes modalités, à un enfant de constituer son propre dictionnaire à partir d'informations mises à sa disposition, ou encore de consulter ou de modifier de l'information introduite antérieurement. Ces activités donnent à ce module des aspects de type "module ressources". D'autres activités à l'intérieur de ce module permettent à l'enfant de construire des expressions à partir d'éléments du dictionnaire. Par ces dernières activités, nous obtenons des caractéristiques de type "module exploration".

Les modules résolution-de-problème-d'enseignement doivent conduire de façon contrôlée à l'acquisition de savoirs par l'élève. Les activités sont regroupées dans une bibliothèque des activités susceptibles de permettre à un enfant d'acquérir un savoir particulier : sur une forme d'identification d'un ensemble de mots, sur la notion de mot, sur des relations entre l'écrit et l'oral.

Par la suite, lorsque nous parlerons de AMICAL, nous nous référerons uniquement aux modules résolution-de-problème.

1.1. Module Résolution de problème d'enseignement

Pour présenter les concepts génériques des modules résolution-de-problème, nous nous appuierons essentiellement sur le module "**Résolution-de-Problème**" qui concerne le début de l'apprentissage de la lecture et est centré sur le rôle fondamental du mot dans cette phase de l'apprentissage.

Dans ce module, le système se propose d'accompagner l'enfant dans ses débuts d'apprentissage de la lecture.

---

[2] Informatiquement, un module est défini par un ensemble de logiciels.

Les objectifs pédagogiques d'une première session de travail sont les suivants :

- mise en correspondance de chaînes écrites avec des chaînes phoniques,
- identification et reconnaissance d'une cinquantaine de mots,
- mobilisation de stratégies multiples[3] dans l'identification de mots,
- mise en évidence des régularités de l'écrit.

Le système doit déterminer dynamiquement l'enchaînement des **sessions didactiques**[4], et pour chacune de ces sessions, calculer l'enchaînement des **situations didactiques**[5] à présenter à l'utilisateur.

Un enchaînement de situations didactiques correspond à un **plan d'enseignement** ([Chambreuil & al., 1994]) défini par le système pour favoriser le développement parallèle et complémentaire des composantes de la lecture pour un élève donné.

Les plans d'enseignement sont générés par un planificateur à partir de la définition des objectifs, leur niveau de finesse d'analyse du processus de lecture, le choix, la caractérisation et les interrelations des activités.

Un plan d'enseignement est une structure à quatre couches. Un schéma simplifié est donné dans la figure 1-1 ci-dessous .

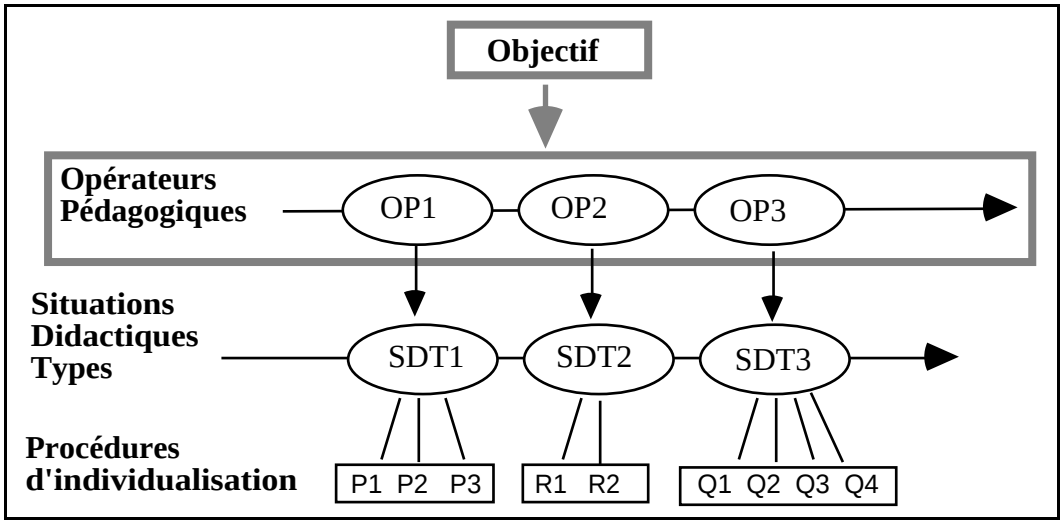


figure 1-1 : Structure d'un plan d'enseignement

Dans la première couche, se trouve un élément unique, l'**objectif d'enseignement** d'une session. Cet objectif est une structure complexe, correspondant au problème à résoudre par le système.

---

[3] Un mot peut être reconnu soit globalement, soit grâce à ses premières lettres, soit grâce à un début effectif d'analyse.

[4] Une session didactique correspond à ce qui pourrait être l'objectif d'une "leçon de lecture" dans une classe.

[5] Une situation didactique est l'homologue d'une activité pouvant être proposée à l'élève pour contribuer à la résolution du problème d'enseignement.

La deuxième couche se présente comme une suite d'**opérateurs pédagogiques**. Cette suite correspond à une suite de types d'actions à entreprendre pour atteindre l'objectif d'enseignement.

La troisième couche est une suite de **situations didactiques types** (**SDT** pour la suite), chaque situation didactique type étant associée à une occurrence d'opérateur pédagogique. A ce niveau du plan, nous obtenons un enchaînement de situations didactiques types à proposer à l'élève pour atteindre l'objectif initial de la session.

Chaque SDT peut être, selon différents aspects, adaptée à sa fonctionnalité pédagogique et à un élève particulier. Cette adaptation se fait en déterminant des procédures d'instanciation (d'individualisation) de la SDT.

La quatrième couche est constituée des procédures d'individualisation déterminées pour chaque SDT. Le couple formé par une SDT de la troisième couche et ses procédures d'individualisation de la quatrième couche conduit à une **situation didactique individualisée** (SDI) susceptible d'être proposée à l'élève.

Chacune des situations didactiques individualisées proposée à l'élève est réalisée dans des **ateliers**[6], interfaces d'interactions entre le système et l'utilisateur.

L'ensemble des activités d'une session au début d'apprentissage est basée sur un texte.

Le déroulement de sessions didactiques met en jeu des connaissances multiples telles que connaissances pédagogiques, connaissances sur l'apprentissage de la lecture, connaissances linguistiques, connaissances sur la mise en œuvre des ateliers.

1.2. AMICAL, système à base de connaissances

Les systèmes intelligents d'aide à l'apprentissage (I.C.A.I. : *Intelligent Computer Assisted Learning*) se sont distingués des systèmes d'E.A.O. traditionnels par la séparation des connaissances du système en deux composantes [Woolf, 1987] :

- les connaissances du domaine (*Domain Knowledge*), c'est-à-dire celles mises en œuvre pour résoudre un problème propre au domaine.
  Toutes les règles explicitées dans MYCIN[7] [Shortliffe, 1974] sont des connaissances appartenant au domaine.
- les connaissances pour enseigner (*Teaching Knowledge*)
  Elles intègrent non seulement les connaissances pédagogiques que l'enseignant met

---

[6] Un atelier est l'ensemble de ressources informatiques (fenêtres, icônes, textes, sons, images et codes) mises en œuvre pour réaliser une situation didactique (ou une partie de situation didactique). Un même atelier peut être utilisé par des siuations didactiques distinctes. Une situation didactique individualisée peut exiger la mise en œuvre de plusieurs ateliers.

[7] MYCIN fait partie des précurseurs les plus connus des systèmes experts. Ce système permet d'établir des diagnostics médicaux.

en œuvre durant les sessions, mais aussi le **modèle de l'élève** (page 49, [Woolf, 1987]). Les "*t-rules*" (*tutorial rules*, règles tutorielles), que l'on trouve dans GUIDON [Clancey, 1979] (une extension de MYCIN en vue d'enseigner les règles utilisées pour le diagnostic) font partie de ces connaissances. Dans cet ensemble de règles que Clancey a rajouté au système EMYCIN[8], certaines permettaient une discussion autour d'une règle donnée appartenant au domaine, d'autres permettaient de choisir des questions-types pertinentes par rapport à une règle étudiée.

Cette séparation des connaissances en deux parties présente plusieurs avantages :

- Certaines connaissances pédagogiques (par exemple, celles d'un degré général) peuvent être utilisées pour l'apprentissage d'autres domaines. L'idéal (économiquement parlant) serait l'obtention d'un système tutoriel intelligent en rajoutant uniquement des connaissances tutorielles à un système expert. Cela permettrait un transfert d'expertise du système expert vers l'apprenant moins coûteux en temps.[9]
- la mise à jour de connaissances dans une composante peut se faire indépendamment de l'autre composante.

Les connaissances du domaine dans AMICAL

Dans la plupart des tuteurs, l'élève peut observer les étapes successives de résolution d'un problème faite par l'expert. Les explications fournies à la demande de l'élève permettent à ce dernier de se façonner sa propre expertise. Par exemple, dans APLUSIX[10] [Nicaud, 1987], l'élève peut visualiser les différentes étapes de la factorisation d'un polynôme.

Dans le cas de AMICAL, il est difficile d'expliquer à l'apprenant la démarche à suivre

---

[8] EMYCIN (Essential MYCIN) [Van Melle, 1979] fut obtenu en généralisant MYCIN. Il peut être considéré comme l'ensemble de toutes les ressources de connaissances de MYCIN qui sont indépendantes du domaine. Cet ensemble est formé de l'interpréteur de règles, le module d'explication et celui d'acquisition de connaissances. On peut dire que MYCIN peut être vu comme un système de type EMYCIN.

[9] Le fait que GUIDON fut conçu pour travailler avec n'importe quelle base de connaissance de type EMYCIN constitue l'un de ses points forts. Ainsi, la connection de GUIDON à d'autres systèmes de type Emycin comme PUFF [Kunz & al., 1979] (un système donnant des interprétations de tests de fonctions pulmonaires)

[10] Il s'agit d'un environnement d'apprentissage de raisonnement algébrique (factorisation de polynômes, résolution d'équations et d'inéquations polynômiales), où les connaissances du domaine sont composées de règles de transformation ainsi que de méta-règles de choix de ces transformations.

pour lire un mot. Même une hypothétique et parfaite maîtrise des différentes ressources mises en œuvre par l'apprenant lors de l'acte de lecture n'est en mesure d'éliminer cette difficulté. En effet, nous sommes confrontés au problème du transfert d'un savoir-faire face à un savoir formalisé. Malgré la difficulté de formalisation du domaine, nous sommes néanmoins en mesure d'établir une liste de connaissances que nous voudrions transférer à l'apprenant par l'intermédiaire d'activités [Chambreuil & al., 1991]. Ces différentes connaissances du domaine sont :

- L'écrit possède une organisation.
  Le mot est un objet graphique particulier composé de lettres variant en nature, nombre et position. Il est lui-même utilisé dans des structures complexes telles que les expressions, phrases et textes.
- L'écrit est en relation avec l'oral.
- L'écrit est en relation avec des significations.

Dans le module "Résolution-de-problèmes", les connaissances du domaine se composent des informations relatives à la structure d'un mot (nombre de lettres, de consonnes, de voyelles, syllabes, ...), à la capacité du système de prononcer une phrase ou un mot, des connaissances relatives aux textes lus (sémantique, syntaxe des phrases).

L'expertise de construction sémantique de l'histoire doit alors être considérée comme faisant partie des nombreuses expertises du domaine.

1.3. AMICAL, système multi-agents

La résolution du problème d'enseignement nécessite des expertises différentes et une coopération de ces dernières dans la prise de décisions. Ces expertises ne sont pas formulées de façon complète et ne posent pas nécessairement les mêmes problèmes de représentation (possibilité de représentations distinctes).

Pour ces différentes raisons, l'environnement AMICAL est conçu dans l'optique actuelle des systèmes multi-agents. En matière de génie logiciel, grâce à la modularité inhérente à la structure d'une architecture multi-agents, nous pouvons entreprendre une intégration plus efficace du savoir-faire des chercheurs participant à ce projet. De plus, les agents développés à des fins spécifiques peuvent être intégrés aux modules de AMICAL. Ainsi, le gestionnaire d'ateliers (développé dans le cadre d'une expérimentation sur site d'une planification manuelle établie par des enseignants pédagogues) possède (à quelques détails près) toutes les fontionnalités de l'agent chargé de l'interaction avec l'élève dans le module résolution-de-problèmes de l'environnement Amical.

Les agents dans le module "résolution-de-problème-d'enseignement".

Le calcul d'un plan pédagogique ainsi que sa réalisation dans le module " résolution-de-problème" est du à la coopération de plusieurs agents :

- **l'agent de planification didactique**
  ([Cherkaoui & Nehémie, 1993], Chambreuil & al., 1994])
  Il est chargé de construire le plan d'enseignement pour une session didactique. La construction, l'exécution et la révision de ce plan est l'œuvre de trois processus, voire **sous-agents** [11] :
  - le sous-agent générateur de plan
  - le sous-agent exécuteur de plan
  - le sous-agent replanificateur

- **l'agent de représentation de l'élève** [Néhémie, 1992].
  Sa tâche consiste à bâtir un modèle de l'élève à partir des différentes interactions entre AMICAL et l'apprenant. Un modèle synthétique de l'élève créé par cet agent pourrait être mis à la disposition des autres agents.

- **l'agent de gestion d'interface**. [Mahmoud & Chambreuil, 1995]
  A partir des connaissances sur la mise en œuvre des ateliers, cet agent assure le contrôle du déroulement de l'atelier. Il génère les comptes-rendus (comme par exemple, les résultats de chaque essai) qui seront traités pour être exploités par l'exécuteur de plan.

- **l'agent linguiste.**
  Considéré comme l'expert du domaine, cet agent possède des connaissances sur les mots (analyse phonologique, morphologique, syllabique, ...), les structures syntaxiques, sémantiques.

- **l'agent spécialiste de l'apprentissage de la lecture.**
  Il assume la définition de l'objectif d'enseignement à chaque session. Cet objectif est déterminé à partir des représentations de l'élève. Il intervient aussi dans le choix des histoires qui seront lues à l'enfant.

Nous limitons l'étude présentée dans cette thèse à celle des problématiques de représentations de connaissances posées par des connaissances expertes manipulées par

[11] Un agent peut être lui-même considéré comme une communauté d'agents collaborant à la réalisation d'un but déterminé.

l'agent linguiste. La compétence sur le domaine d'apprentissage, à laquelle nous nous intéressons, concerne la capacité pour le système d'avoir une représentation sémantique des textes lus ou proposés à l'élève en lecture.

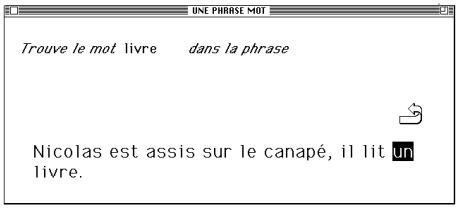
2. Connaissances sur une histoire, leurs rôles dans l'environnement AMICAL

2.1. Les textes : fil conducteur des activités

Dans le début de l'apprentissage de la lecture, toutes les activités sont conduites à partir de textes. Dans le module "résolution-de-problèmes", le système dispose de plusieurs situations didactiques types permettant de faire évoluer l'état des connaissances de l'élève. La bibliothèque de SDT comporte parmi ses éléments les SDTs suivants :

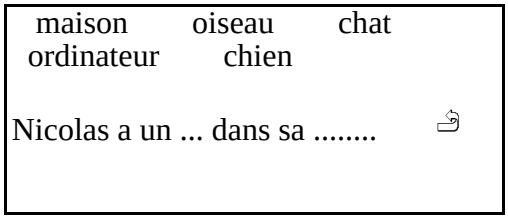
- SDT présentation de texte.

- SDT reconnaissance de mots en contexte.



- SDT reconnaissance de mots en liste avec ou sans modèle

- SDT closure



Voici une des histoires pour enfant utilisée par AMICAL :

◊ Scène 1
- C'est Nicolas.
- C'est la maison de Nicolas.
- Dans la maison de Nicolas, il y a Nicolas.
- Dans la maison de Nicolas, il y a un chat.
- Dans la maison de Nicolas, il y a un oiseau.

◊ Scène 2
7. Nicolas est assis sur le canapé, il lit un livre.
8. Le chat dort sur le tapis à coté du canapé.
9. L'oiseau vole dans sa cage.

◊ Scène 3
10. La porte de la cage est ouverte.
11. L'oiseau s'envole par la porte ouverte.
12. L'oiseau se pose sur le tapis à coté du chat qui dort.
13. L'oiseau n'a pas vu le chat qui dort à côté de lui sur le tapis.

◊ Scène 4
14. Mais le chat ne dort plus. Il a vu l'oiseau posé à côté de lui.
15. Le chat saute sur l'oiseau.
16. L'oiseau s'envole et se pose sur le canapé à côté de Nicolas qui lit.
17. Nicolas n'a pas vu l'oiseau posé à côté de lui.

◊ Scène 5
18. Quand le chat saute à son tour sur le canapé, Nicolas lève la tête.
19. Nicolas comprend que le chat veut manger l'oiseau.
20. « Viens dans mon livre ! » dit-il à l'oiseau.
21. L'oiseau s'envole et se pose sur les pages du livre.

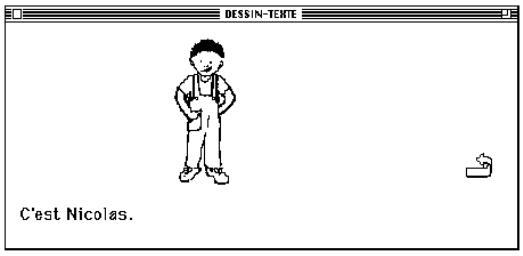
◊ Scène 6
22. Nicolas ferme son livre.
23. L'oiseau est dans le livre bien à l'abri.
24. Le chat, lui, lit le titre du livre :«Histoire du chat qui ne mangeait jamais d'oiseaux».

figure 1-2 : Une histoire utilisée par AMICAL

Le texte entier de l'histoire a été découpé en 6 unités appelées **scènes** et en 23 sous-unités phrases. Une scène correspond à une étape de la progression d'un récit (présentation des acteurs, lieux et actions de départ, phases de l'action, aboutissement).

À partir du texte précédent qui aurait été choisi pour un élève particulier, nous pourrions obtenir la séquence d'activités ci-dessous.

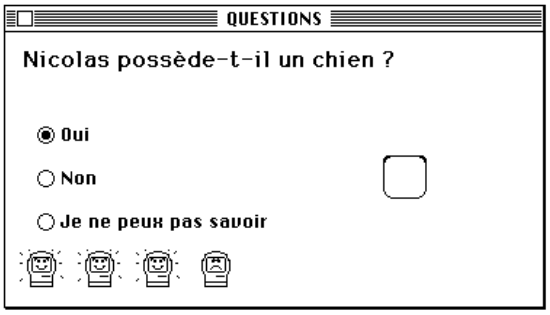
❶ Le système AMICAL lance une activité de présentation de textes. Un plan contenant la phrase « C'est Nicolas. » et l'image associée apparaît dans une fenêtre à l'écran. La phrase est mise en inversion vidéo, elle est ensuite prononcée, et remise en vidéo normale. L'apprenant a la possibilité de faire répéter la phrase en cliquant sur la phrase. Pour indiquer à AMICAL qu'il peut passer à la suite, l'enfant clique sur l'icône .



❷ La phrase « C'est la maison de Nicolas. » est présentée dans la même fenêtre. Le même scénario que précédemment, est répété : mise en inversion vidéo, lecture, remise en vidéo normale. Et ainsi de suite, jusqu'à épuiser toutes les phrases de la scène 1.

❸ Lorsque la dernière phrase de la scène a été présentée, AMICAL choisit d'afficher dans une même fenêtre tous les textes de la scène 1 accompagnés éventuellement d'une image. À son apparition, chaque texte est relu par l'ordinateur. Ici encore, l'enfant a la possibilité de faire relire les différentes phrases de la scène. La possibilité est donnée à l'enfant de pouvoir revenir aux différents plans de la scène en cours.

❹ Lorsque l'ensemble de la scène est présenté, une série de trois à quatre questions de compréhension est posée à l'élève. Elle permet de savoir si l'enfant a compris ou non les textes qui lui ont été lus. Si le système juge que l'enfant n'a pas bien saisi la scène lue, il lui représente à nouveau la scène. Dans le cas où il est estimé qu'il n'a pas rencontré de problème de compréhension, le système passe alors à des activités autour du mot.

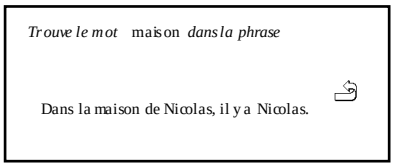


❺ Pourrait ensuite être présentée la séquence a-b-c-d d'activités de travail sur le mot.

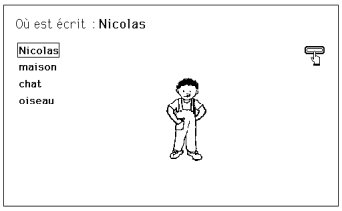
a) Reconnaissance en contexte (1 ou plusieurs textes)

b-c) Reconnaissance dans une liste de mots avec ou sans modèle

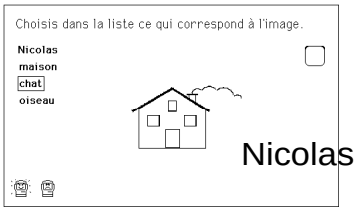
d) Exercice de closure



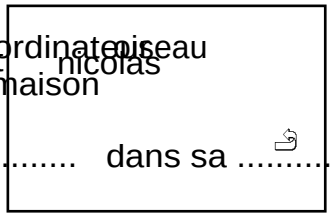
(a)



(b)



(c)



(d)

❻ Après analyse des résultats de l'élève dans les activités autour du mot, le système procède alors à la présentation d'une nouvelle scène, et ainsi de suite jusqu'à épuiser toutes les scènes de l'histoire.

Décomposition de l'histoire en scènes

L'ensemble des textes, que l'on choisit de lire à l'enfant, relate généralement une histoire destinée aux enfants. Pour éviter des problèmes de mémorisation et pour entretenir la motivation de l'élève, les histoires "longues" sont donc découpées. Ces découpages nécessaires ne sont pas réalisés de manière arbitraire. Ils ont été réalisés en tenant compte des différentes étapes de la progression du récit. Cette décomposition est proche de ce que pourrait nous donner l'utilisation d'une grammaire du récit ([Rumelhart, 1977]). Ce découpage devrait faciliter la compréhension du récit. De plus, cette structuration ne différera pas énormément de celles que l'apprenant rencontrera dans les histoires autres que celles utilisées par Amical, qu'il pourra être amené à lire.

L'histoire présentée précédemment est décomposée de la manière suivante :

- sc1 - présentation des personnages
- sc2 - situation de chaque personnage
- sc3 - changement de situation de l'un des personnages (oiseau)
- sc4 - action d'un personnage face à cette situation - réaction des autres - situations nouvelles
- sc5 - nouvelles actions - réaction des autres - situations nouvelles
- sc6 - situations finales des personnages - morale de l'histoire

La session avec AMICAL que l'on vient de décrire peut être vue sous l'aspect du cycle suivant (figure 1-3) :

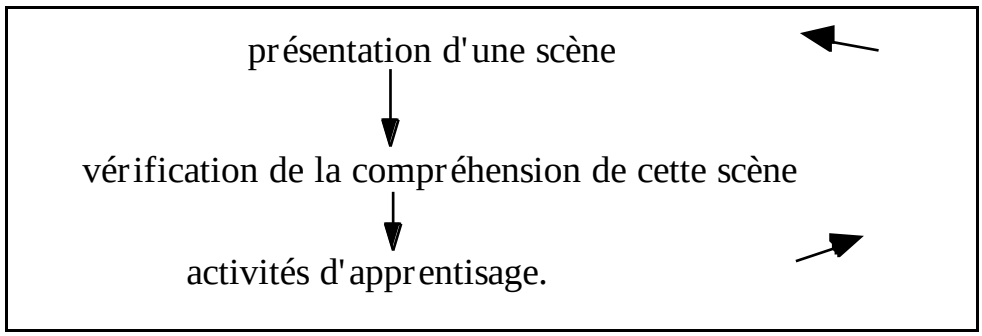


figure 1-3 : Cycle d'activités basé sur une histoire dans Amical.

Chaque scène d'une histoire est utilisée comme support des activités d'apprentissage. A chaque cycle, le système présente donc une scène nouvelle de l'histoire. Après chaque présentation d'une scène, une série de questions de compréhension est posée à l'élève afin de juger le bon suivi de l'histoire par l'enfant. Il faut s'assurer en effet que le texte lu a bien été compris. Cela permettra de détecter d'éventuels problèmes de concentration, d'inattention, de distraction (due aux médias ou autres) ou même l'existence d'un problème de compréhension de la langue chez l'apprenant. Il n'est pas question ici de remédier ou même d'essayer de remédier au problème de compréhension de la langue naturelle qui n'est nullement propre à l'activité de lecture. Néanmoins, il nous faut être capable de le détecter dans son éventualité.

Après la vérification de compréhension, le système lance un certain nombre de situations didactiques individualisées permettant à l'apprenant de saisir le concept de mot. Ces SDI utilisent des éléments (mots, connaissances) qui ont été véhiculés par les textes présentés à l'apprenant.

Le cycle est en pratique plus complexe qu'un simple enchaînement. Certaines phases ne sont pas systématiquement présentes dans le cycle et peuvent fort bien être omises pour des raisons particulières ; il se peut que l'on choisisse d'éviter de vérifier la compréhension de la scène quand on démarre une nouvelle session avec une même histoire et un même élève, sachant que ce dernier a bien compris les premières scènes.

Les textes de l'histoire lue constituent, comme on le voit, le fil conducteur d'un ensemble d'activités présentées à l'élève.

2.2. Rôles des représentations sémantiques des textes

Dans la mesure où toutes les activités de lecture sont liées directement ou indirectement à l'histoire, cette dernière jouant le rôle de support, le système aura besoin des connaissances sur l'histoire, qu'il sollicitera auprès de l'expert lors de la détermination des problèmes posés à l'élève ou de l'analyse de réponses dans diverses activités.

Pour chacune de ces sollicitations, nous donnerons quelques exemples de mise en œuvre à l'intérieur de situations didactiques diverses.

2.2.1. Détermination des problèmes posés à l'apprenant

Lorsque AMICAL veut vérifier la compréhension des textes présentés, il doit au préalable avoir sélectionné un ensemble de questions pertinentes. Parmi les critères de choix des questions, le fait que les informations portées par les textes présentés permettent à l'apprenant de répondre à la question est un des plus importants. Ce critère n'est déterminable que si AMICAL dispose d'une représentation sémantique de l'histoire lue.

Il est intéressant de définir d'autres critères de choix de ces questions dans le but de créer une certaine dynamique. En effet, si la liste de questions associées à un texte (ou scène) était figée, le système poserait le même groupe de questions après chaque présentation de ce texte (ou scène). Un choix "dynamique" évite au système de poser les mêmes questions, et permet aussi d'approfondir certaines parties de l'histoire. Cette dynamique présente aussi l'intérêt de maintenir chez l'apprenant une motivation qui aurait vite disparu si le système lui posait toujours les mêmes questions.

La sémantique de l'histoire intervient aussi dans le choix des phrases dans les activités-mot de type "phrases à trous". Les phrases à compléter ont généralement un lien avec l'histoire lue.

Les représentations sémantiques des textes utilisés sont donc sollicitées lors des choix d'une des composantes du problème :

- choix de la question dans les activités questions de compréhension,
- choix de la phrase à trou dans un exercice à trou.

2.2.2. Analyse des réponses de l'élève

Certaines situations didactiques utilisent des phrases à trous qui doivent être complétées. L'illustration ci-dessous présente une situation didactique mettant en œuvre la phrase à trou « Dans la maison de Nicolas, il y a [trou]. » Les différents mots mis à la disposition de l'enfant sont : *Nicolas, chat, oiseau, maison, chien.*

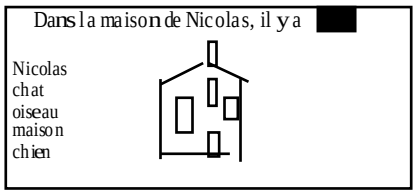


figure 1- 4 : Une situation didactique utilisant une phrase à trou

Les cinq solutions possibles sont donc :

- Dans la maison de Nicolas, il y a Nicolas. (1)
- Dans la maison de Nicolas, il y a chat. (2)
- Dans la maison de Nicolas, il y a oiseau. (3)
- Dans la maison de Nicolas, il y a maison. (4)
- Dans la maison de Nicolas, il y a chien. (5)

La solution (1) est la seule qui soit correcte syntaxiquement et sémantiquement. (2) et (3) sont correctes sémantiquement parce qu'elles restent en accord avec les éléments d'informations véhiculés par l'histoire. Ces deux phrases sont incorrectes syntaxiquement car elles n'ont pas une construction syntaxique correcte. (4) est incorrect, de même que (5) car il n'y a rien dans le texte qui puisse amener à une telle affirmation. Ici, une connaissance de l'histoire doit permettre d'apprécier à différents degrés les réponses données par l'apprenant.

Un comportement bien défini en fonction de toutes les réponses possibles n'est pas souhaitable surtout si l'on entrevoit la perspective d'ateliers, où l'enfant est amené à compléter les phrases à trous par introduction au clavier des mots manquants. Le nombre de possibilités augmente sans compter les différentes combinaisons possibles si l'on demandait de compléter plusieurs phrases à trous. On peut imaginer, par exemple, la phrase à trou précédente avec une activité la mettant en œuvre et une modalité exigeant l'introduction par l'enfant d'un ou plusieurs mots à l'intérieur de ce trou. Les réponses "Nicolas", "un garçon", "un être humain" seraient aussi acceptables. Par contre, une réponse comme "un singe" serait fausse.

Ici, la recherche d'éventuelles proximités sémantiques entre les réponses données par l'enfant et les éléments qu'il connaît de l'histoire doit être entreprise.

2.2.3. Conclusion

Pour caractériser un ensemble précis de problèmes fonctionnels que l'agent linguiste aura à gérer, nous limitons notre cadre de réflexion à un aspect de la détermination des problèmes posés à l'apprenant : le choix des questions de compréhension, dans les activités de type vérification de la compréhension.

3. Recherche d'un formalisme de représentation sémantique : éléments de la problématique retenue.

Même limités à cette situation fonctionnelle particulière dans l'environnement Amical, les problèmes de représentation de connaissances restent encore très complexes.

Par rapport à cette complexité, nous formulerons, dans les paragraphes qui suivent, un ensemble de contraintes que nous prendrons en compte. Ces contraintes sont de natures différentes.

Nous distinguerons :

- les contraintes de type "génie logiciel"

  Elles sont amenées par la description du système informatique que nous nous donnons comme objectif de réalisation pour cette thèse.
- les contraintes pédagogiques,
- les contraintes sur les capacités représentationnelles et inférentielles du système de représentation de connaissances.

3.1. Problématique fonctionnelle : choix des questions pour des activités de compréhension.

3.1.1. Les agents mis en jeu

Le choix de ce contexte de travail nous permet de restreindre la problématique de représentation de connaissances à cette partie de fonctionnalité dans AMICAL. Même restreinte à cette fonctionnalité, il est difficile de décrire, dans ce cadre précis d'interaction, les actions et les connaissances gérées par les agents (linguistes, planifieurs, modélisateur de l'élève) de AMICAL précédemment décrits.

Nous simplifierons notre vision des interactions internes entre agents dans AMICAL dans le contexte choisi aux interactions entre deux macro-agents (chacun pouvant être constitué d'un ou plusieurs agents) que nous appelons SIMULREQ (**Simul**ation de **Req**uêtes) et SYSQUEST (**Syst**ème de Choix de **quest**ions). Ces deux macro-agents peuvent contenir des éléments communs (agents, connaissances).

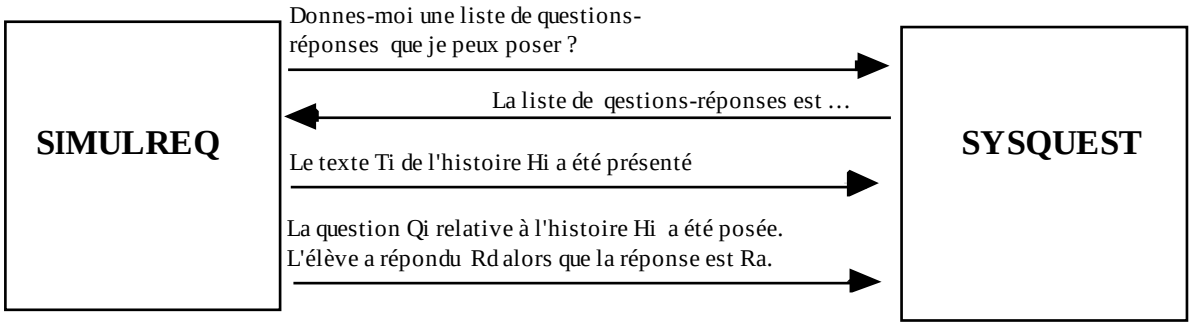


figure 1-5 : Vision des interactions internes

A chaque fois qu'un texte d'une scène quelconque est présenté à un élève, SIMULREQ informe SYSQUEST de ce fait. SYSQUEST va créer une connaissance sur l'histoire à partir des informations sémantiques associées au texte présenté. Après présentation d'une scène, SIMULREQ demande à SYSQUEST une liste de questions-réponses qu'il peut poser. SYSQUEST renvoie une liste de questions réponses. A partir de cette liste, SIMULREQ décide des questions à poser. Lorsqu'une question posée reçoit une réponse, SIMULREQ renvoie à SYSQUEST la question, la réponse attendue et la réponse effectivement donnée par l'apprenant.

En procédant ainsi, nous évitons tout d'abord de nous attarder sur des problèmes liés à la constitution effective des agents de AMICAL. Ensuite, cette conception nous permet de simuler plus facilement la phase de vérification de compréhension tout en ayant comme préoccupation la problématique de représentation de connaissances retenue. Bien entendu, vu la structure multi-agents adoptée pour AMICAL, il est clair que l'on pourra reprendre une partie des éléments qui ont été développés dans le cadre de cette thèse.

3.1.2. Bases de connaissances pour l'agent linguiste

Nous ne nous préoccupons pas de la façon dont les informations sémantiques des textes seront construites à partir du texte.

Nous retenons l'hypothèse de l'existence de deux bases de connaissances :

- BH (Base d'Histoires) où l'on trouvera pour chaque texte d'une histoire sa représentation sémantique
- BCP (Base de Connaissances Principales) mise à jour au fur et à mesure de la présentation des textes à l'apprenant. Lorsqu'un texte t_i d'une histoire h_i est présenté, l'expert linguiste à l'intérieur du macro-agent SYSQUEST) met à jour cette base en incluant les informations sémantiques r_i rattachées au couple (t_i,h_i).

Nous chercherons à montrer l'adéquation du système de représentation de connaissances présenté dans cette thèse pour structurer ces informations.

3.1.3. Génération de questions ou choix dans une base de questions

AMICAL utilise des ressources (texte, image, son) associées à chaque question de compréhension.

Ces ressources doivent exister pour les raisons suivantes :

- recherche d'une mise en évidence de la fonctionnalité de l'écrit grâce à l'affichage du texte de la question
- possibilité de "fixer" la question, en accompagnant le texte de la question d'un graphisme.
- possibilité de production orale de la question puisque le texte de la question ne peut encore être "décodé" par l'enfant.

Pour disposer de ces trois ressources, deux choix étaient possibles.

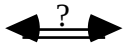
- soit la génération automatique de questions à partir de la représentation sémantique de l'histoire,
- soit la question existe sous forme texte, son (et image, optionnellement) à l'intérieur d'une base de questions

C'est cette deuxième solution que nous avons retenue.

1) Génération automatique de questions à partir de la représentation sémantique de l'histoire

Cette première solution présente quelques inconvénients :

(a) Tout d'abord, l'étape de production de formes interrogatives à partir des informations existantes implique une étude préalable du mode de représentation de ces formes interrogatives.[12] Cette difficulté pourrait être contournée en travaillant plutôt sur les composantes syntaxiques et sémantiques des phrases de la scène. En effet, en supposant que l'on dispose uniquement des composantes syntaxiques et sémantiques de toutes les phrases de la scène, il est possible de créer une question en opérant des transformations suivant des règles bien établies (ajout d'une forme en "Est-ce que", inversion du sujet). L'inconvénient de

[12] Il faut noter que certains types de représentations proposent des solutions pour marquer l'interrogation. Ainsi, Schank propose dans sa Dépendance Conceptuelle un moyen de créer des formes interrogatives à partir du marqueur "?". La présence de ce dernier dans  indique une interrogation sur l'acte lui-même. En lieu et place des rôles conceptuels comme Actor, Objet, Bénéficiaire, Direction (points de départ et d'arrivée), ce marqueur exprime l'incomplétude de la représentation sémantique.

cette solution est de ne travailler que sur des parties de l'histoire et aucunement sur les informations qui auraient pu être inférées dans le système.

(b) Une fois les structures interrogatives définies, il faut alors générer automatiquement le texte correspondant à cette forme interrogative.

(c) De plus, comme l'enfant ne sait pas encore lire, le texte obtenu n'est pas suffisant. Il faut lui associer une ressource sonore que l'ordinateur devrait être en mesure de produire. Le problème de production sonore reste entier. Les technologies proposées actuellement sur le marché n'ont pas les qualités qui permettent de les intégrer sans problèmes dans cette situation d'apprentissage particulier.

2) Base de questions gérée par AMICAL

Cette solution a été retenue pour AMICAL. Elle consiste à disposer d'une base de questions auxquelles sont associées des ressources sonores, textes, préenregistrés.

Ici, pour disposer de sons de bonne qualité, l'enregistrement sonore des questions est fait à l'avance. Le texte correspondant à la question est lui aussi introduit dans le système. Les réponses à ces questions ne sont bien sûr pas données à l'avance puisqu'elles dépendent en grande partie du moment où elles sont posées. Pour éviter de gérer des couples (réponses, contexte) pour chaque question, le système doit être capable de déterminer la réponse à une question à partir de connaissances sur l'histoire et aussi d'informations associées à la question. Ce choix implique donc que AMICAL dispose d'une forme sémantique de la question. Les formes associées à la question peuvent être calculées ou introduites de la même manière que les sons et textes. A l'intérieur de cette base de questions, peuvent aussi coexister d'autres questions relatives à d'autres histoires.

3.2. Contraintes génie logiciel sur les bases de connaissances de l'agent linguiste.

SYSQUEST dispose donc de représentations sémantiques des textes lus et des questions posées par le système. Nous formulerons ici cinq contraintes sur ces représentations sémantiques :

❶ La représentation sémantique d'une phrase d'un texte ne doit contenir que des informations relatives à cette phrase. On ne devrait pas trouver d'éléments donnant des informations non contenues dans la phrase représentée. (Par exemple, on ne choisira pas la solution d'intégrer dans la représentation des informations de mise à jour et de suppression

d'informations apportées par les phrases précédentes du texte.)

❷ La représentation sémantique d'une question est utilisée pour calculer la réponse associée à partir de l'état actuel de l'histoire. Elle ne doit pas être un ensemble de connaissances procédurales. Le respect de cette exigence permet à des non-informaticiens de décrire les questions, sans qu'il soit nécessaire qu'ils aient des connaissances d'informaticiens.

❸ La représentation sémantique d'une question doit rester "simple" à formuler.

Les structures sémantiques associées aux questions seront utilisées pour générer des requêtes à l'intention de l'expert linguiste qui structure la connaissance sur l'histoire.

La base de questions sera d'autant plus aisée à créer que cette formulation sera simple.

❹ Le système de représentation utilisé doit permettre l'ajout aisé de nouveaux éléments qui augmenteront la capacité inférentielle du système. Cette exigence est respectée dans les systèmes à base de règles de production en fournissant la possibilité d'ajouter de nouvelles règles d'inférences.

❺ Le système de représentation devra permettre de retrouver ou d'inférer de manière simple des informations à partir de fragments de connaissances. Le mécanisme de recherche d'informations à l'intérieur de la base de connaissances doit pouvoir s'accorder d'une description partielle des informations recherchées.

En effet, la facilité de description souhaitée ne devrait pas non plus conduire à des solutions entraînant des difficultés de recherche des informations. Ces difficultés pourraient par exemple se traduire en algorithmes plus ou moins simples à implémenter.

Par exemple, la recherche d'un lien (connu ou non) entre deux entités présentes dans une base de type objet requiert la plupart du temps le parcours de la base entière.

3.3. Contraintes pédagogiques sur le choix des questions : implications sur les représentations de connaissance

Les contraintes pédagogiques retenues sont importantes dans la mesure où elles ont une incidence sur les représentations.

Chaque trait caractéristique de chacune des questions de compréhension que l'on peut poser, ou de la réponse attendue à cette question, ou de l'ensemble de ces questions, pourra avoir des incidences sur les structures de représentations choisies, pour mémoriser l'histoire lue.

Pour chaque trait caractéristique, nous essaierons de justifier les décisions prises.

Nous décrirons ces fonctionnalités, de celles imposant le moins de contraintes sur la représentation à celles définissant un plus grand nombre de contraintes.

3.3.1. Limite du nombre de questions

Le nombre des questions posées par le système est limité à quatre questions afin de ne pas lasser l'apprenant. Cette caractéristique n'a aucune implication sur le choix de la représentation.

3.3.2. Ne pas poser de questions déjà posées

On ne reposera pas à l'élève une question qui lui a déjà été posée et à laquelle il a donné une réponse correcte. Le système peut néanmoins choisir une question déjà posée si de nouvelles réponses sont attendues. Pour cela, le système doit mémoriser pour chaque question posée, la réponse attendue et celle donnée par l'apprenant.

Implications sur la représentation.

Cette caractéristique met en évidence la présence d'un calcul des réponses. A moins de gérer des triplets de la forme (<question> , <réponse>, <contexte>), le système AMICAL doit posséder un processus d'inférences permettant d'établir des réponses aux questions données.

3.3.3. La réponse à la question devrait être du type *oui/non/je ne peux pas savoir*

Plusieurs raisons sont à l'origine de ce choix.

Tout d'abord, le système AMICAL n'a pas de module de compréhension de la parole.
Les difficultés sont principalement dues ([Bonnet, 1984]) :

- à l'existence de bruits parasites,
- aux variations et imperfections au niveau de la prononciation d'un individu à l'autre
- aux changements de prononciation au niveau d'un même individu pouvant provenir de l'état psychologique ou physiologique de ce même individu,
- aux différentes prononciations d'un phonème voire d'un mot suivant l'existence ou non d'un contexte,
- à l'existence/non existence de silence entre les mots et aussi à l'intérieur d'un mot,
- à l'existence d'homonymes.

Les projets de compréhension de la parole les plus connus sont HEARSEY [Erman & al., 1980] et MYRTILLE [Haton & al., 1978]. Ce domaine d'expertise fait l'objet de nombreux développements en Intelligence Artificielle. Les réalisations actuellement disponibles sur le marché ne proposent que la reconnaissance de mots isolés. Les applications se limitent actuellement à la reconnaissance de commandes vocales mettant en jeu des fonctions dans les logiciels.

Comme la parole ne peut pas être utilisée comme véhicule d'information de l'apprenant vers le système, il ne nous reste plus que les dispositifs de saisie d'information tels que le clavier ou la souris. Comme nous travaillons avec l'hypothèse d'un apprenant en phase initiale d'apprentissage de la lecture, il apparaît peu approprié de lui demander d'introduire au clavier sa réponse puisqu'il ne saura pas la formuler. La souris semble être le meilleur média. Différentes modalités d'utilisation sont possibles. On peut proposer à l'enfant de choisir (en cliquant sur) certaines images/icônes en guise de réponse. Ce choix présente l'avantage d'être agréable d'utilisation, mais n'a pas été retenue pour des raisons pédagogiques. En effet, ici, la fonctionnalité de l'écrit n'est pas mise en valeur. Pour répondre à une question de compréhension, on demandera plutôt à l'élève de choisir une phrase écrite comme réponse.

Le nombre de mots qu'il est capable de reconnaître étant encore faible, il n'est pas concevable de lui proposer une liste d'expressions-réponses qu'il n'est pas à même de "lire". Pour éviter qu'une telle situation ne se produise, il ne faudra proposer que des phrases réponses qu'il reconnaîtra facilement ou qu'on lui apprendra (dans une session 0) à reconnaître. Pour la première session, nous supposerons alors que l'enfant est capable de reconnaître les expressions "oui", "non" et "je ne peux pas savoir".[13]. Les trois réponses possibles présentent aussi l'avantage de ne pas être ambiguës.

Implications sur la représentation.

La réponse "je ne peux pas savoir" présuppose l'existence d'une base de connaissances basée sur l'hypothèse d'un monde ouvert.

En effet, si l'hypothèse d'un monde clos avait été utilisée, seules les assertions vraies devraient être explicitement présentes dans la base, et l'absence d'un fait serait équivalente à sa négation. Il n'y aurait donc pas de moyen de répondre dans ce cas "je ne peux pas savoir".

Nous retenons l'hypothèse d'un monde ouvert. Nous devons représenter toutes les connaissances explicitement transmises par le texte, qu'elles se présentent sous forme de

[13] Les réponses à certaines des questions posées ne peuvent pas être simplement oui/non. Certaines questions peuvent requérir des connaissances supplémentaires qui n'ont pas été transmises par le texte. Le choix "je ne peux pas savoir" permet à l'apprenant de bien marquer l'insuffisance d'informations pour lui permettre d'inférer une réponse plus simple comme "oui" ou "non".

négation ou pas. Ainsi, une information comme « Nicolas n'est pas dans la maison. » se traduira toujours par l'ajout d'une connaissance supplémentaire. (Dans l'hypothèse d'un monde clos, ajouter cette information consiste à retirer l'information « Nicolas est dans la maison. » si elle existe.) A partir de l'hypothèse de monde ouvert, un fait non déduit n'est pas nécessairement faux.

3.3.4. Pertinence des questions par rapport aux scènes présentées

On suppose ici que le système dispose d'une base de questions, qui ne sont pas toutes forcément en rapport avec l'histoire qui a été sélectionnée pour le travail.

Les questions posées devront avoir un lien avec les scènes présentées et notamment avec les plus récentes.

1) Adéquation de la question à l'histoire.

La base de questions dont dispose AMICAL peut très bien receler :

- des questions d'ordre général , comme « Est-ce que les oiseaux volent ? », utilisées dans des activités autres que la vérification de la compréhension.
- des questions, n'ayant aucun rapport avec l'histoire qui a été choisie, par exemple, une question est posée à propos d'un personnage X qui n'aurait pas été présenté.

2) Adéquation de la question aux scènes dernièrement présentées.

Supposons que l'histoire de Nicolas (figure 1-2) ait été choisie pour être présentée à l'enfant. Demander à la fin de la présentation de la première scène à l'enfant s'il y a un chien dans la maison est tout à fait pertinent. Le système chercherait ici à savoir si l'enfant a bien capté tous les personnages de l'histoire qui se trouvent tous à l'intérieur de la maison. En revanche, poser la même question, au milieu de l'histoire, après la description d'une phase de l'action, paraît incongru.

Pour éviter cela, une exigence sur la proximité de la forme des réponses données aux questions et d'une des dernières informations introduites est nécessaire.

Imaginons, en effet, que le système introduise toutes les informations concernant la scène 3 et travaille sur l'adéquation de la question « Y-a-t-il un chien dans la maison ? » . La réponse est soit *non* (si on suppose que tous les éléments de la maison ont été vus), soit *je-ne-peux-pas-savoir* (si l'enfant croit que les éléments de la maison ne lui ont été que partiellement présentés). Formulées autrement, les deux réponses possibles deviennent « Il n'y a pas de chien dans la maison. » et « Je ne peux pas savoir s'il y a un chien dans la maison ». Or ces deux affirmations n'ont aucun lien évident avec les éléments de la scène 3.

Posée après présentation de la scène 3, la question « Y-a-t-il un oiseau dans la cage ? »

sera plus pertinente car la réponse « L'oiseau n'est pas dans la cage. » peut être inférée à partir des informations données par la phrase 10 «L'oiseau s'envole par la porte ouverte. ».

Implications sur la représentation.

Le système doit posséder des moyens de traiter des connaissances d'ordre général puisqu'il doit être apte à les gérer dans sa base de questions.

Ensuite, le fait que la question soit pertinente par rapport aux scènes les plus récentes amène à la réflexion suivante. Si le système sait quelles sont les informations relatives aux scènes les plus récentes, il sait donc par déduction quelles sont les plus anciennes.

Il reste à savoir, s'il est capable de distinguer parmi les anciennes, celles qui le sont plus que d'autres. S'il n'y a que deux valeurs (Récent ou Ancien) d'attribut associables à l'information, le système AMICAL peut paraître limité dans le sens où il ne serait pas capable d'établir différents degrés de "passé" dans l'histoire.

Le choix d'un système capable de faire la distinction entre différentes portions du passé implique alors la notion de gestion du temps.

Les structures de connaissances relatives à l'histoire manipulées par AMICAL sont donc temporalisées. Parmi les formalismes existants permettant la gestion temporelle des informations, il nous faudra faire un choix.

3.4. Capacités de représentation et d'inférences.

Ci-dessus nous avons identifié des contraintes de type génie logiciel et pédagogique qui seront associées aux connaissances que doit gérer l'agent SYSQUEST. Nous noterons ci-dessous quelques aspects de capacité de représentation et d'inférence que devra permettre le formalisme de représentation.

Pour illustrer ces aspects, nous nous appuierons sur des représentations de connaissances spatiales. C'est également le domaine des relations spatiales qui servira de domaine de réflexion pour toute la suite de ce travail. Ce choix est simplement lié au fait que dans les premières histoires analysées, beaucoup de questions portaient directement ou indirectement sur des connaissances spatiales. Ce choix ne signifie pas que le formalisme qui sera proposé se donne pour objectif (et pour seul objectif) la représentation des connaissances spatiales, au sens par exemple d'un travail tel que celui de Laure Vieu ([Borillo & Vieu, 1989], [Vieu, 1991]). Ce formalisme n'est pas lié à un domaine particulier de représentation, et à l'inverse nous n'avons pas cherché à inscrire dans ce formalisme tous les problèmes qui seraient spécifiquement liés à la représentation du domaine spatial.

3.4.1. Capacité de représentation

1) Relations et objets mis en relation

Nous devons représenter des informations spatiales où interviennent deux catégories d'objets différents parmi les éléments localisés :

- des objets "concrets", comme Nicolas ou l'oiseau.
- des objets "abstraits", comme des événements ou des actions.

Par exemple, dans la phrase « Nicolas est dans la chambre », l'élément localisé Nicolas est un objet concret. La phrase « Nicolas casse une chaise avec un marteau dans la maison » localise l'événement « Nicolas casse une chaise avec un marteau » dans la maison.

Cette distinction entre éléments localisés est importante. Dans le premier cas, un seul élément est localisé. Dans le second cas, un ensemble d'objets caractérisant l'événement est localisable. Dans l'exemple ci-dessus, Nicolas, une chaise et un marteau sont dans la maison.

2) Relations et quantifications

Il s'agit plus précisément de représenter des connaissances de localisation quantifiées universellement, comme celle présente dans la phrase « (toutes) Les chambres d'une maison se trouvent dans cette maison. ».

Cette représentation est importante dans la mesure où cette connaissance est exploitée par le système pour inférer de nouvelles informations.

3) Relations et négation

La négation apparaît aussi dans la description d'une localisation. En effet, des phrases comme « Nicolas n'est pas dans la maison. » peuvent être données et nécessitent une représentation. Nous devrons décrire de manière explicite ces représentations. Dans la mesure où ce type de représentation existe, la base de connaissances est gérée avec l'hypothèse du monde ouvert. Toute localisation spatiale non présente ou non déductible n'est pas obligatoirement une localisation négative.

4) Représentation temporelle

Toute localisation spatiale est valide à l'intérieur d'un intervalle de temps donné. La localisation d'une entité par rapport à une autre peut évidemment évoluer en fonction du temps. Dans cette thèse, nous nous limiterons à la recherche des différents moyens nous permettant de temporaliser les structures de représentations spatiales.

3.4.2. Capacité d'inférences

Nous nous intéressons ici au problème particulier que pose l'exploitation de la capacité inférentielle d'un système de représentation. Il nous paraît important de montrer les difficultés rencontrées pour "implémenter" des connaissances inférentielles spécifiques (ex : règles, procédures) permettant de déduire de nouvelles connaissances à partir d'une base de connaissances.

4. Problématique SYSQUEST

Nous chercherons donc à définir un formalisme de représentation pour SYSQUEST qui est l'agent chargé de fournir une liste de questions-réponses, à partir d'une base de questions, d'une base de texte et d'informations d'interactions avec l'apprenant (textes lus, questions-réponses données).

Un ensemble de contraintes de plusieurs types par rapport à l'utilisation des structures manipulées a été défini.

Ce sont des contraintes du type génie logiciel. Elles concernent la structuration des textes et des questions des deux bases de connaissances utilisées. Cette structuration devra permettre une facilité de recherche d'informations, un ajout aisé de nouvelles données dans les bases.

Ce sont aussi des contraintes sur les structures de représentation qui devront contenir des informations liées à leur utilisation pédagogique (question déjà posées, ...).

Ce sont enfin et essentiellement des contraintes de capacité de représentation et de capacité inférentielle (relations entre objets de nature différente, quantification, négation, temporalité).

Chapitre 2 : Systèmes de représentation des connaissances et problématique SYSQUEST

Nous proposerons dans ce chapitre d'étudier différents systèmes de représentations qui seraient susceptibles d'apporter des éléments de solution à la problématique SYSQUEST. Nous examinerons trois types de systèmes :

- les systèmes basés sur la théorie de la dépendance conceptuelle de Schank,
- les systèmes basés sur les travaux de Brachmann
- les systèmes basés sur les graphes conceptuels de Sowa

Pour chaque système, nous indiquerons les éléments de choix du système et nous présenterons chaque système en décrivant les entités cognitives manipulées, ainsi que les mécanismes d'inférence proposés. Ensuite, nous analyserons la façon dont ces systèmes répondraient à la problématique SYSQUEST. Nous décrirons la manière de représenter la localisation spatiale d'objets concrets, la localisation d'événements, des localisations quantifiées, la négation dans la localisation et le temps. Pour chaque point abordé, nous décrirons les possibilités d'inférences de nouvelles connaissances et les modes d'accès à ces informations.

Avant de présenter ces systèmes, nous décrirons les études actuelles sur les représentations spatiales issues de la grammaire cognitive. Cette présentation des travaux sur la localisation spatiale a plusieurs objectifs. Tout d'abord, elle nous permet d'introduire à des notions que nous utilisons dans les systèmes analysés. Ensuite, elle permettra de situer notre propre travail par rapport à ce que serait un système de représentation spatiale. Enfin, elle permet d'introduire à différentes notions générales sur lesquelles s'appuiera le formalisme des DCAS que nous proposerons par la suite.

1. Relations spatiales : approches théoriques dans le cadre des sciences cognitives

Les travaux linguistiques décrits portent sur des expressions formulant des relations spatiales, plus particulièrement quelques éléments constituant la sémantique des relations spatiales du français. Ces relations sont présentes dans des expressions qui décrivent des situations stables et qui sont introduites par les prépositions de lieu tels que : "dans", "à", "sur", "au-dessus", "au-dessous".

Nous décrirons notamment les travaux de Vandeloise ([Vandeloise, 1986]) qui

présentent une étude des énoncés de la forme "R(x,y)" où R est la relation spatiale existant entre l'objet x et y en mettant en évidence les difficultés rencontrées dans la définition de ces relations spatiales

Les travaux de Vandeloise s'inscrivent dans un cadre plus général des constructions de la forme "N(cible) V(état) Prép Loc. N(site)" décrites par Andrée Borillo ([Borillo A., 1990]).

Nous décrirons ensuite une contribution à l'étude des inférences spatio-temporelles menée sous l'aspect traitement informatique par Laure Vieu ([Borillo & Vieu, 1989], [Vieu, 1991]) qui propose un modèle formel d'un raisonnement sur la relation spatiale "dans".

Les travaux sur la localisation spatiale peuvent être classés en deux catégories :

- une catégorie regroupant des travaux sur des phrases utilisant des compléments de lieu, et plus particulièrement ceux qui participent à la localisation, au sens de Andrée Borillo ([Borillo A., 1990]), en fournissant des repères à une situation dans l'espace lorsque celle-ci reste stable,
- une catégorie où les éléments de préoccupation sont des constructions à base de verbes dit de mouvement ou de déplacement, comme *venir d'un lieu, passer dans un certain endroit, atterrir à un certain point*. ([Boons, 1987], [Lamiroy, 1987])

Du point de vue sémantique, les localisations spatiales étudiées ici peuvent être représentées par des expressions dans lesquelles une relation statique est établie entre deux termes, l'un désignant l'objet (dans un sens très général) à localiser, l'autre servant de repère à la relation.

1.1. L'approche de Vandeloise

Les études linguistiques de Vandeloise [Vandeloise, 1986] et Borillo [Borillo, 1990] sont des travaux que l'on peut situer dans le cadre de la grammaire cognitive. ([Langacker,1986a], [Langacker,1986b]).

Nous commencerons la description de cette approche en présentant les différents arguments donnés par Vandeloise pour montrer les limites d'une approche de la description des prépositions par la logique classique.

Les concepts lui permettant de décrire les mots spatiaux seront ensuite présentés.

Nous verrons que des informations extra-linguistiques sont utilisées dans les relations spatiales, et que le locuteur peut jouer un rôle important dans la caractérisation de la préposition.

Ensuite, quelques éléments de caractérisation (comme l'orientation intrinsèque et l'orientation contextuelle des objets) des termes de la relation spatiale seront donnés .

Enfin, nous terminerons la présentation de cette approche en décrivant les conditions

que doivent respecter les règles d'usages bâties autour des prépositions.

1.1.1. Limite de l'approche par la logique classique

Dans « L'espace en Français : Sémantique des prépositions spatiales » [Vandeloise, 1986], l'auteur souligne l'insuffisance des définitions logiques (exprimables en logique des prédicats) des prépositions spatiales, notamment celles proposées par Cooper (1969), Leech (1969), Bennet (1968), Miller et Johnson-Laird (1976).

Différentes propositions de définition de la relation " x dans y" furent données :

- x est situé à l'intérieur de y à condition que x soit plus petit que y. (Cooper)
- x est inclus ou contenu dans un lieu y à 2 ou 3 dimensions (Leech)
- "dans y" équivalent de "locatif(intérieur y)" (Bennett)
- dans(x,y) : le sujet prépositionnel x est dans l'objet prépositionnel y si partie(x,z) & Inclus(z,y).(Miller & Johnson-Laird)

Ces tentatives lui semblent désespérées car il considère que cette approche se ramène à envisager les prépositions comme des catégories classiques gouvernées par des conditions nécessaires et suffisantes d'appartenance alors qu'aucun mot ne se prête à des descriptions aussi rigides

Il préfère à cette catégorisation "classique", la catégorisation naturelle au sens de Lakoff (1982), pour qui le langage serait organisé en catégories naturelles où certains usages d'un mot sont plus représentatifs que d'autres.

Même si l'on admet que ces définitions permettent de capter les usages les plus représentatifs des mots (et donc en conformité avec une approche du langage structurée en catégories naturelles où certains usages d'un mot sont plus représentatifs que d'autres), il reste quatre différences essentielles :

❶ La logique a tendance à idéaliser les termes de la relation.

❷ La logique ne tient généralement pas compte des contraintes existantes sur la cible et le site (par exemple, le fait que la cible est généralement petite, mobile, de position inconnue par rapport au site)

❸ La logique ne peut exprimer certaines contraintes additionnelles ajoutées aux expressions sans sacrifier les contraintes de compositionnalités[14] (de la logique). Dans l'exemple « Annette est à son bureau. » de Herskovitz, on suppose Annette en train de travailler, et non pas seulement sur son lieu de travail.

---

[14] Le sens de deux termes est calculable à partir de leur sens isolés.

❹ La réduction de l'extensivité des définitions logiques (il reprend l'exemple de Cooper où a est plus petit que b) par des restrictions de sélection parfois erronées si on considère l'expression bien formée « L'arbre est dans un pot. ».

1.1.2. Les concepts utilisés pour l'analyse spatiale

Vandeloise décrit les mots spatiaux par rapport à des concepts liés à la connaissance extra-linguistique que partagent les locuteurs d'une même langue.

Dans son analyse des mots spatiaux, cinq concepts jouent un rôle essentiel :

❶ les directions déterminées par la symétrie du corps humain

❷ des concepts de physique naïve (axe vertical, relations porteur/porté, contenant/contenu)

❸ l'accès à la perception

❹ la rencontre potentielle

❺ les orientations générales et latérales.

La forme du corps humain détermine deux directions :

la direction frontale et la direction latérale.

Par exemple, les expressions "en face de" , "dans le dos de" sont caractérisables dans une certaine mesure par la direction frontale.

Des concepts de physique naïve sont partagés par tous les usagers d'une langue, et sont utilisés dans des localisations spatiales. Ainsi, l'axe vertical est utilisé comme direction de la chute d'un objet ou direction dans laquelle un homme se lève. Ce concept est présent dans des expressions comme "au dessus/en dessous". La relation "porteur/porté" apparaît dans la description des prépositions "sur/sous", la relation contenant/contenu dans les expressions utilisant "dans/hors de".

La préposition "derrière" (respectivement "sous") exprime l'inaccessibilité à la perception suivant la transversale à l'axe horizontal. (respectivement vertical). Dans la phrase « Le livre est sous le tapis. », le livre cité peut être inaccessible à la perception.

La rencontre potentielle, qui implique donc un mouvement, de la cible, du site, et d'un élément non exprimé, appelé pôle, permet de mieux comprendre l'usage de la préposition "avant" dans des phrases comme « Le coureur américain est avant le coureur français. »

L'orientation générale comprend comme principaux traits la direction frontale, la ligne du regard et la direction du mouvement. L'orientation latérale a pour traits principaux la direction latérale et la perpendiculaire à l'orientation générale.

Ces concepts permettent une description plus complète des expressions “devant/derrière” et “à gauche/à droite” que ne permettaient pas les directions frontale et latérale.

1.1.3. Les relations cible/site et site/locuteur

Dans l’approche de la grammaire cognitive et pour l’analyse des relations spatiales, la cible correspond à l’objet à localiser, le site à l’objet de référence.

1) L’utilisation d’informations extra-linguistiques dans le langage

A travers les remarques concernant la **non-conversité**[15] des relations “devant/derrière”, “gauche/droite”, à partir des exemples de phrases suivant, Vandeloise nous fait découvrir l’existence d’un principe général du langage qui s’appuie sur une base extra linguistique.

Prenons les deux phrase suivantes :

- Le bâton est devant la maison. (1)
- La maison est derrière le bâton. (2) **?**[16]

« L’objet de la relation dont la position est incertaine ne peut être localisé sans référence à une entité dont la position est mieux connue. » ([Vandeloise, 1986])

Le viol de ce principe dans la phrase (2) explique l’anormalité de celle-ci.

De plus, d’autres caractéristiques de la cible et du site sont implicites :

- la position de la cible est une information nouvelle,
- la position du site est une information ancienne,
- généralement, la cible est petite et difficile à repérer,
- la cible est souvent mobile ou susceptible de bouger, et le site est généralement immobile et stable.

Il note toutefois certaines exceptions dont l’acceptabilité pourrait être justifiée par un mouvement potentiel.

2) Importance du locuteur dans certaines relations spatiales

Des relations entre le site et le locuteur existent.

---

[15] Deux relations R1 et R2 sont converses <=> Si a R1 b alors b R2 a
Si a R2 b alors b R1 a

[16] Le point d’interrogation est utilisé pour marquer l’anormalité de la phrase qui la précède.

Par exemple, le site qui paraît à Vandeloise le plus vraisemblable dans une phrase comme « Paris est près. » est la position du locuteur. Il postule que « au fur et à mesure que la description s'objective, que le site se dégage du locuteur, l'expression linguistique devient plus complexe. » (figure 2-1)

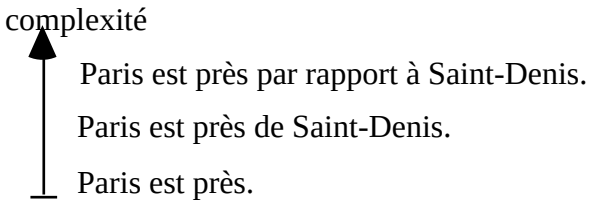


figure 2-1 : complexité d'expressions linguistiques

Quatre étapes dans les descriptions, du stade le plus égocentrique au stade le plus indépendant de son locuteur, sont proposées :

<<

❶ - Site inexprimé référent à la position réelle du locuteur

❷ - Site inexprimé référent à la position virtuelle du locuteur

❸ - Site exprimé par la préposition de, traduisant un détachement partiel du site et du locuteur.

❹ - Site exprimé introduit par l'expression par rapport à, traduisant le détachement total du locuteur et du site qui devient un point de référence délibérément adopté.

>>

Cette influence entraîne l'idée d'une dépendance de la préposition spatiale (voir scène ci-dessous) de trois termes : la cible, le site et le locuteur.

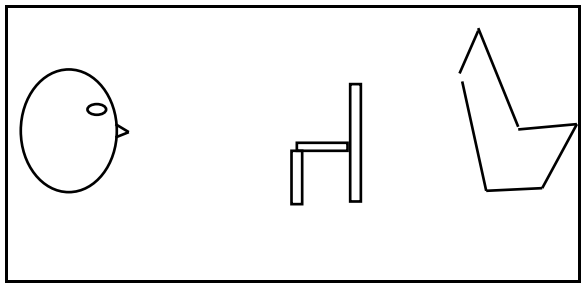


Figure 2.2 : La chaise est devant le fauteuil.

Vandeloise introduit l'idée de deux catégories de prépositions :

- les **prépositions directionnelles** ("au dessus/en dessous", "devant/derière1", "à gauche/à droite") où la caractérisation de la cible par rapport au site se fait selon des directions précises (verticale, frontale, latérale). Dans cette catégorie, les sites sont généralement introduits par "de" Elles possèdent généralement deux termes.
- les **prépositions fonctionnelles** ("avant/après", "devant/derrière2") où sont décrites la manière dont les termes de la relation s'organisent spatialement selon n'importe laquelle des directions . Ces prépositions admettent difficilement des comparatifs. (? La voiture est plus avant que l'arbre.). En général, ces expressions font intervenir trois termes.

Même si les deux versions de "devant/derrière" contribuent à estomper la distinction directionnel/fonctionnel, cette dernière peut aider à clarifier les utilisations des prépositions spatiales.

1.1.4. Caractérisation des termes de la relation spatiale

1) Idéalisation des termes d'une relation spatiale

La multiplicité des usages d'une relation proviendrait des différentes facettes des termes mis en jeu. Ces objets ne sont pas utilisés avec rigueur, mais « idéalisés en fonction de la perspective selon laquelle ils sont considérés ».

Selon les perspectives choisies, certaines caractéristiques seront prises plutôt que d'autres. Il affirme, de plus, que les limites de certains objets (problème de commencement et de fin de parties du corps humain, par exemple) sont indéterminées par nature.

Il démontre l'insuffisance de la définition topologique de dans (« a est dans b si les limites du site incluent les limites de la cible. ») dans le cas de contenants ouverts comme le verre dans la scène ci-dessous.

figure 2-3 : La mouche est dans le verre.

Même si les limites topologiques sont étendues aux enveloppes convexes des objets, la phrase « la mouche est dans le verre » ne peut s'appliquer à la scène du dessin ci-dessous.

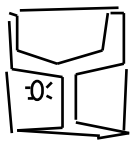


figure 2-4 : La mouche n'est pas dans le verre.

La relation fonctionnelle contenant/contenu pour définir la relation "dans" est retenue par Vandeloise.

2) Orientation intrinsèque des objets

Il existerait deux types d'orientations intrinsèques.

La première appelée **orientation positionnelle**, notée par Bierwisch en 1967, concerne l'orientation d'un objet par rapport à sa position usuelle. Cette orientation se fait grâce au principe de fixation postulé par Vandeloise formulé de la manière suivante :

« Un objet peut être qualifié par rapport à sa position usuelle, même si sa position diffère par rapport au moment de l'énonciation. »

Pour une bouteille, son dessus est la partie située au niveau du bouchon quelle que soit sa position actuelle (normale, renversée ou retournée).

Vandeloise ajoute parmi les conditions nécessaires à une orientation intrinsèque d'un objet, deux conditions :

- les différentes parties selon lesquelles il est orienté sont reconnaissables,
- la position usuelle de cet objet doit être connue de presque tous les membres d'une communauté linguistique.

La seconde orientation, appelée **orientation anthropomorphique,** fut signalée par Teller en 1969.

Par analogie avec la face humaine, la face d'un objet la plus fonctionnelle ou la plus chargée de détails est appelée son "devant" et la face opposée sera dite son "derrière".

3) Orientation contextuelle des objets

Des objets, comme les arbres ne sont pas intrinsèquement orientés selon la direction frontale ou latérale (Où est le côté gauche d'un arbre ? son devant ?)

Cependant, si les locuteurs étaient amenés à désigner l'avant de tels objets, ils choisiraient soit une orientation en miroir (pour les locuteurs français), soit une orientation en tandem (pour les locuteurs du haussa, langue africaine) que l'on peut découvrir dans les schémas de Vandeloise ci-dessous :

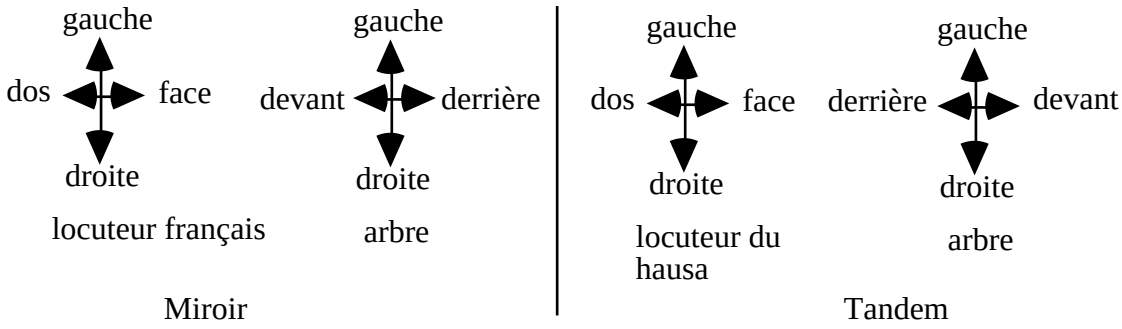


figure 2-5 : Orientation contextuelle des objets

Le locuteur ne serait pas le seul à présenter la capacité d'orienter contextuellement un objet. D'autres objets intrinsèquement orientés dans le contexte en sont capables. (ex du comédien, du metteur en scène, du public et de la table où le comédien est invité à se mettre devant).

1.1.5. Règles d'usage de la relation spatiale

L'analyse de Vandeloise s'appuie sur les descriptions fonctionnelles des prépositions spatiales du français. Pour rendre compte de tous les usages, Vandeloise cherche à caractériser chaque préposition par une (ou plusieurs) règles d'usage. Dans l'idéal, cette règle est unique et est une condition nécessaire et suffisante pour l'usage d'une préposition. Cependant, la règle unique peut n'être que nécessaire et pas suffisante. Dans ce cas, les usages inappropriés sont écartés par des conditions supplémentaires raisonnables (en nombre restreint) avec des concepts introduits pertinents par rapport à la description de la préposition. La plupart des règles énoncées par Vandeloise sont de ce type. Cette unicité s'obtient au prix de l'utilisation de la ressemblance de famille. La ressemblance de famille est un concept introduit par Wittgenstein (1953) et est représenté par différentes combinaisons d'un ensemble de caractéristiques ou de similitudes. Ces traits ne sont pas obligatoirement nécessaires ou suffisants. Ce concept global, non définissable de façon logique, est considéré comme l'impulsion de la préposition correspondante.

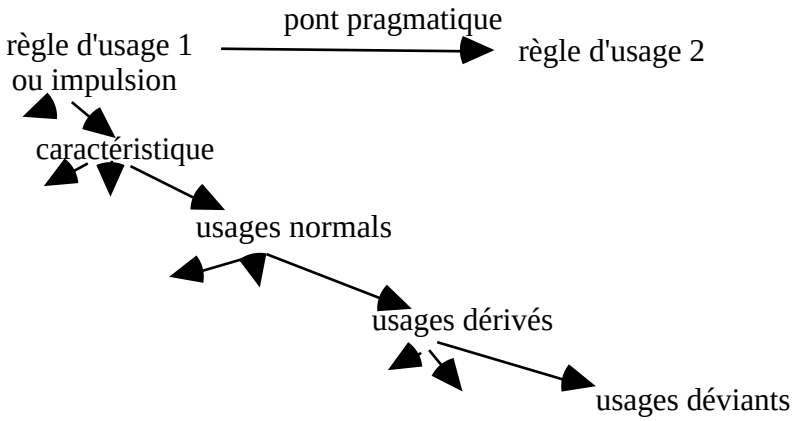


figure 2-6 : Règles d'usage d'une relation spatiale

Exemple :

Le concept global de *sur* est "porteur/porté".

Ce concept global possède des aspects ou caractéristiques (accès à la perception, ordre sur la verticale, contact, opposition à la pesanteur) qui peuvent être analysés séparément et motiver des usages de la préposition.

. a) accès à la perception : la mouche est sur le plâtras

. b) ordre sur la verticale : la tasse est sur la table

. c) contact : la table est sur le sol, le couvercle n'est pas sur la table

. d) opposition à la pesanteur : il y a des feuilles sur l'arbre

La règle d'usage unique est S, définie comme suit :

S : a est sur b si a est le deuxième élément de la relation "porteur/porté" et b le premier élément de cette relation.

1.2. Des traits de localisation dans le verbe

Dans les phrases utilisant des compléments de lieu, Andrée Borillo [Borillo A., 1990] recense plusieurs types de constructions :

- la forme la plus courante,

  N(cible) V(état) Prép Loc. N(site).

  V(état) = être, se trouver, ...

- **N(cible) V (état) N(site)**, où

V(état) sont des verbes de nature transitive comme : courir, recouvrir, occuper, remplir, encombrer, obstruer, embarrasser, inonder, joncher, parsemer

Ex : Les livres occupent toute la table.

• **N(site) V(état) N(cible)**, où

V(état) est du type contenir, comporter, emprisonner, entourer, encercler, encadrer, surplomber, supporter.

Ex : cette armoire contient tous mes livres.

• **N(site) V(état) Prép loc N(cible)**

V(état) Prép(loc) sont du type : déborder de, regorger de, crouler sous, être plein de, être rempli de, être bourré de.

Dans les constructions de ce type “N(cible) V(état) Prép Loc. N(site)”, Andrée Borillo fait état de la présence, dans le verbe d’état, de traits sémantiques[17] qui renvoient « soit à des caractéristiques propres à la cible : caractère humain/non humain, caractère concret / abstrait, ou propres au site : caractère solide /liquide, soit impliquent l’idée de nombre, soit encore comportent des propriétés de dimension spatiale ou d’orientation, soit enfin ajoutent l’idée de durée ou de subsistance dans l’état ».

Elle note aussi que « certains verbes (comme *être*, *y avoir*, *se trouver*, *être situé*, *être placé*) expriment la relation spatiale sans autre indication, en particulier sans ajouts de traits précisant la nature de la cible, sa manière d’être, son aspect. » Elle les qualifiera par l’adjectif neutre.

Elle propose aussi un regroupement d’un certain nombre de verbes suivant la dominance d’un trait de caractérisation (voir annexe 1).

Elle montre l’existence d’un trait de localisation, parfois secondaire, mais présent dans le verbe lui-même. Elle circonscrit un ensemble de verbes désignés par **Vloc** (“verbes de localisation”) qui remplissent sur le plan sémantique des conditions qui font d’eux des indicateurs de repérage dans l’espace. Cette caractérisation sur le plan syntaxique reste encore difficile, et semble provenir de cette « mobilité sémantique dont les verbes font preuve » qui se répercute sur leurs compléments, parfois contraint de « modifier leurs propriétés au risque d’un changement de statut. »

1.3. Traitement informatique des localisations spatiales : propositions d’un modèle formel spatio-temporel

En IA, l’étude de l’espace linguistique est peu abordée. Bien que de nombreux travaux dans le domaine linguistique existent, un des aspects importants pour l’informatique et l’I.A. est peu abordé. Il s’agit de l’étude des mécanismes inférentiels, qui, comme le note Laure Vieu, n’entre pas dans la problématique des linguistes. Ces derniers s’attachent plus à

[17] La possibilité de combinaison de ces traits n'est pas exclue.

déterminer une couverture complète de l'usage de chaque préposition.

1.3.1. Les premiers jalons d'une méthodologie liant l'espace et sémantique du temps

Dans « Eléments pour la formalisation du raisonnement spatio-temporel naturel », Mario Borillo et Laure Vieu ([Borillo & Vieu, 1989]) proposent un modèle formel spatio-temporel pour la relation "se trouver dans". Cette relation leur paraît suffisamment importante et représentative de l'expression de l'espace dans la langue, pour poser les premiers jalons d'une méthodologie liant l'espace et sémantique du temps. Ils s'appuient sur les travaux de Borillo et Vandeloise sur les prépositions spatiales du français.

Pour leurs besoins, ils limiteront la relation *dans* à celle où le contenu est englobé totalement par le contenant. Il leur paraîtra nécessaire d'exprimer la place occupée par les entités physiques concernées. Ils définiront le référent spatial de x, noté Refs(x), comme la partie de l'espace délimité par ce qui "se trouve dans" ou "fait partie de " x.

Pour construire la base de connaissances spatio-temporelles, il leur semble nécessaire de représenter des informations négatives, plus précisément, de travailler sur un monde ouvert.

De plus, chaque entité de cette base peut être située. Une référence globale appelée "univers" est utilisée dans le cas où l'on ne connaît pas sa localisation.

La croissance de cette base est telle que seules les informations nécessaires sont conservées et les informations déductibles détruites.

<u>Le modèle spatio-temporel proposé.</u>

Le langage temporel de Shoham fournit les formules de base du type :

True(t1, t2, p) où

p est une proposition atemporelle, [t1,t2] un intervalle de temps.

Ce formalisme a pour avantage de ne pas définir la nature des termes temporels t1 et t2. ([Bras, 1990])

Ce modèle manipule des entités de trois univers distincts :

- TW, l'univers temporel,
- SW, l'univers spatial
- W, l'univers des entités du discours

Après avoir :

- défini les symboles associés à ces trois univers

- défini les termes de W, SW, et TW
- défini les fbf (formules bien formées) de ces univers
- introduit la notation
  False(tt_1, tt_2, r(t_1,t_2,...., t_n))
- défini la sémantique (interprétation, calcul du sens)
- introduit des axiomes de base (concept de *dans* et *constitue,...*)

plusieurs propriétés du modèle ont pu être démontrés. (ex : la transitivité de dans)

Ces propriétés peuvent être utilisées pour deux fonctions d'inférence bien distinctes : une gérée par des règles de déduction, l'autre par des règles d'introduction.

Les règles de déduction permettent de déduire des informations non présentes dans la base.

Les règles d'introduction sont déclenchées lors d'une introduction d'information dans la base de connaissances. Elles sont plus complexes que les règles de déduction et se présentent sous la forme d'algorithmes. Elles assurent le contrôle de la non redondance d'informations, ainsi que de la cohérence des localisations obtenues et permettent d'éliminer des informations qui étaient au préalable non déduites et qui peuvent être déduites à partir de la nouvelle information.

1.3.2. Une méthodologie pour la recherche sur la sémantique de l'espace en langage naturel

Cette idée est présentée dans le cadre de la thèse de Laure Vieu. [Vieu, 1991]. Elle y montre l'inadéquation des tentatives de formalisation de la sémantique de l'espace, exclusivement basés sur les propriétés géométriques des entités. Des travaux de Vandeloise et de Herskovits, elle tire la nécessité de prendre en compte des propriétés fonctionnelles des objets ainsi que des constituants pragmatiques des relations spatiales. Les structures des représentations spatiales du langage naturel lui paraissent inadaptées.

Elle pose les bases d'une méthodologie pour la recherche sur la sémantique de l'espace en langage naturel en faisant ressortir grâce à une analyse en profondeur trois niveaux distincts dans la signification des marqueurs :

- Le premier niveau d'analyse concerne un niveau "basique" de représentation de l'espace. Il permet d'exprimer les concepts primitifs d'inclusion, de contact et d'alignement.

- Au second niveau, elle prend en considération la nature de chaque entité désignée par un terme du discours, ainsi que les aspects fonctionnels de la relation spatiale étudiée.

- Le troisième niveau prend en considération les aspects pragmatiques basés sur le

contexte extra-linguistique.

L'étude de la préposition spatiale a été menée en trois principales étapes.
- analyse de cette relation spatiale du point de vue linguistique
- description d'un modèle spatio-temporel
- description du module fonctionnel décrivant cette relation spatiale.

1) Analyse de la relation spatiale "dans"

L'inclusion est la notion principale de la sémantique de "dans". La relation d'inclusion revêt une grande importance car sa propriété de transitivité structure fortement l'espace en termes de hiérarchisation de lieux. Elle est aussi à la base des relations d'inclusion entre ensemble (taxinomies), des relations de méronomie (relations de partie à tout entre deux individus étudiés en particulier)

Différentes configurations topologiques descriptibles par "dans" sont étudiées. Elles amènent à définir des notions comme place, intérieur.

La place d'un objet correspond à la place des objets qui le constituent. Cette définition de place permettra d'esquisser un prototype de la relation « x est dans y » différent de « place(x) inclus dans place(y) » et plus proche de « place(x) inclus dans place(intérieur(y)) ».

Laure Vieu conçoit l'intérieur d'un objet comme une entité à part entière (qui a des propriétés et qui se déplace en même temps que l'objet dont elle est l'intérieur).

Un objet peut posséder plusieurs intérieurs La somme des intérieurs d'un objet est l'intérieur de cet objet, et cet intérieur est toujours inclus dans la fermeture convexe de la place d'un objet.

L'intérieur est lié au concept de contenance.

Le principe de fixation est aussi un facteur qui entre en jeu. L'intérieur qui correspond à l'usage habituel ou aux types d'objets supposés contenir est utilisé pour restreindre certains emplois. Par exemple, l'intérieur d'un bol en usage normal est celui d'un bol en position debout. Dans la figure ci-dessous, dans (a), le sucre n'est pas dans le bol, alors que la mouche y est dans (b). Dans (c) et (d), la mouche et le sucre sont dans le bol.

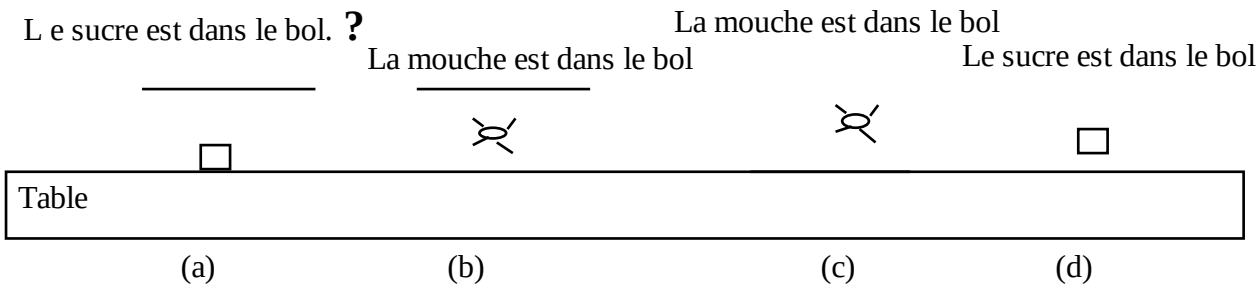


figure 2-7

Dans certains situations, le site ne possède pas de parties contenantes. Pour définir la notion d'inclusion, pour répondre par exemple au cas d'un homme dans une foule, la notion de contours est utilisée.

L'analyse de la relation dans doit être affinée pour tenir compte des situations où la cible est une partie du site.

Laure Vieu distingue plusieurs types de relation "dans/partie" :

• composant/assemblage

Exemple : *Le moteur est dans la voiture.*

Ces utilisations s'expliquent par la notion de contraste.

• morceau/tout.

Elle s'utilise plutôt en combinaison avec un verbe indiquant le détachement d'une portion d'un tout.

Exemple : *Couper une part dans un gâteau.*

• élément/collection

Elle y distingue :

- des inclusions fonctionnelles (*Pierre est dans le jury.*)
- des inclusions contingentes qui permettent des interprétations géométriques. (*L'homme est dans la foule,*)

• sous-collection/collection, substance/tout.

La place des entités est ici difficile à situer.

Exemple : *Il y a du café dans mon café au lait.*

Différents types d'individus doivent être considérés par rapport à une localisation spatiale.

L'intérieur est un **morceau d'espace** décrit comme l'espace dans l'objet. Ces morceaux

d'espaces ne sont désignables qu'à travers des objets (exemple : l'espace dans un objet, l'espace sur un objet, l'espace derrière un objet).

Parmi ces morceaux d'espaces, les **trous** sont utilisés comme cible de la relation "dans".

Exemple : *Il y a un trou dans le mur.*

La définition donnée par Vieu est la suivante :

« Un trou est un morceau d'espace défini par un objet, soit par comparaison avec une forme type ou une forme antérieure, soit par des considérations sur la nature topologique de l'objet, ces deux cas ne s'excluant pas. »

Les **lieux** sont fréquemment rencontrés dans des expressions spatiales. Ces entités ne peuvent être situées dans un objet.

Exemple : *? Paris est dans un cercle de chars.*

La transitivité de *dans* fait l'objet d'une analyse complète pour déterminer les formes autorisées par le langage.

2) Structure spatio-temporelle

Cette structure est utilisée pour la représentation des aspects géométriques des relations spatiales. L'espace est associé au temps pour cette représentation. Ce travail est basé sur les théories logiques de l'espace présentées dans sa thèse.

Cette structure est construite en s'inspirant du formalisme de Clarke pour la topologie et la définition des points. Elle est enrichie pour prendre en compte la relation de contact, la notion de frontière et la notion de limite.

Les définitions s'appuient sur une relation primitive, la relation C de connexion. On écrira C(x,y) lorsque « x est connecté à y ».

Le modèle est ensuite étendu pour compléter la géométrie en vue des aspects géométriques qui sont envisagés dans d'autres relations.

Les aspects esquissés sont :

- la distance
- l'orientation.

Dans cette structure, la construction du temps s'inspire du modèle de Kamp.

Les relations temporelles primitives entre les individus[18] (événements ou non) sont la précédence α et le recouvrement temporel σ.

L'espace temps, alors défini, est purement relationnel. Cela lui permet de ne représenter (avec précision) dans un discours donné que les individus nécessaires et seulement ceux là.

Le modèle spatio-temporel est suffisamment complet pour traiter d'un grand nombre de relations spatiales.

3) Le module fonctionnel

Il est élaboré à partir d'une réflexion sur l'ontologie (le langage dévoile une certaine structure pour ces entités). Elle tient en compte des éléments spécifiques mis en évidence dans l'étude de la relation spatiale étudiée (ici, "dans") ainsi que des éléments retenus pour la modélisation de la structure temporelle.

Il s'agit d'intégrer des aspects fonctionnels, donc non géométriques à des aspects purement spatio-temporels.

Dans ces aspects fonctionnels, interviennent le typage des entités, la fonction *Intérieur* basée sur le concept de contenance et les différentes relations méronomiques.

Les liens entre entités sont décrits précisément :

- la relation de constitution C1
- la relation C2 pour relier une entité collective à l'entité plurielle correspondante à un temps donné (et non quel que soit le temps considéré)
- la relation C3 pour exprimer le fait qu'une entité représente une tranche temporelle de l'autre.

Plusieurs entités différentes sont décrites en donnant leurs définitions.

Les objets normaux (respectivement lieux, morceaux d'espaces, substance) sont distingués par les prédicats OBJ (respectivement LIEU, M.ES et SUBST). Ces trois prédicats sont définis en utilisant le sous-type MMAT (morceau de matière).

A partir des définitions des relations EL_COLLECTION(x, y) , SCOLL_COLL(x,y), PORT_TT(x,y), SUBST_TT(x,y) et FONCTIONNEL(x,y), elle donne les définitions des éléments de la relation composant/assemblage : COMP_ASS(x,y), PART(x,y) et RESTE(x,y,z).

La fonction INT (intérieur) est par la suite définie en utilisant les définitions établies

---

[18] Au sens de Laure Vieu, toute entité possède une composante temporelle.

pour la relation CONV (convexe), la fonction FCONV (fermeture convexe), la fonction PREINT (pré-intérieur).

Pour les différents usages de la relation spatiale dans, elle utilise trois prédicats pour chaque usage :

- DST pour un dans total.
- DSP pour un dans partiel
- DPT, pour un dans/partie de.

Le module fonctionnel proposé permet ainsi le raisonnement spatial sur la relation “dans”. Ce module doit être étendu lorsque l’on voudra intégrer de nouvelles relations spatiales basées sur d’autres prépositions.

2. L'approche par la dépendance conceptuelle de Schank

Nous avons choisi la théorie de la dépendance conceptuelle ([Schank, 1975], [Schank & Riesbeck, 1981]) car elle constitue un système de représentation basé sur l'hypothèse de l'existence de primitives sémantiques. De plus, ces travaux sont à l'origine de recherches sur des structures complexes (scénarios, plans) permettant la représentation d'histoires.

2.1. La dépendance conceptuelle de Schank

2.1.1. Primitives sémantiques

En 1963, Osgood introduit l'idée suivante :

> « *De la même façon qu'un phonème est défini comme un faisceau de traits phonétiques simultanés, la signification peut être définie comme un faisceau de traits sémantiques simultanés.* »

Les sens des mots seraient décomposables en éléments de sens appelés **traits sémantiques** ou **primitives** ou **atomes de sens** ou **marqueurs** ou **sèmes**.

Les premiers travaux développant cette conception sont ceux de Katz et Fodor (1963). Un ensemble de traits a été tout d'abord défini. A chaque sens d'un mot, on associe un sous ensemble de traits présents.

Malheureusement, ces atomes de sens souffrent de leur choix arbitraire et le niveau description résultant n'est pas précisé.

Toutefois, Wilks [Wilks, 1977] met en évidence certaines règles concernant un système à base de primitives .

Finitude : le nombre de primitives est fini et (très) inférieur au nombre de sens à représenter.

Etendue : elles doivent permettre des représentations différentes pour des sens différents.

Indépendance : une primitive ne peut être exprimée à partir d'une autre.

Non réductibilité : aucun ensemble de primitives ne peut être remplacé par un sous ensemble plus petit.

Compositionnalité : le sens d'un mot doit pouvoir se déduire des primitives qui le

composent (l'ensemble des 26 lettres de l'alphabet et des 32 phonèmes ne peut former cet ensemble)

2.1.2. L'approche Schankienne

Les investigations de Schank concernent le traitement du Langage Naturel, plus précisément la compréhension d'histoires. Il s'attache à la recherche de procédures d'analyse (assignation d'un sens) de phrases proposées en entrée ainsi qu'au codage de ce sens (génération) dans un langage naturel, ainsi qu'à la création de représentations significatives et à leur codage .

Deux exigences sont formulées :

- la représentation du sens doit être indépendante du langage (pour marquer la distinction structurelle entre pensée et langage)
- ces procédures doivent se conformer au mieux à ce qui est connu dans le comportement humain.

Schank accorde une grande importance aux prédictions qui se feraient lors de l'analyse de phrases par un humain. D'ailleurs, cette notion est utilisée pour aider à analyser la phrase en créant, par exemple, des attentes d'un certain type d'information.

Des travaux de Lamb (1966), il retient l'idée que chaque niveau de langage a une syntaxe qui prescrit comment les éléments de ce niveau se combinent entre eux.

Selon lui, la fameuse phrase « Des idées vertes dorment furieusement » peut alors se conformer aux règles syntaxiques du niveau syntaxique du langage, mais serait en violation avec les règles de syntaxe du niveau sémantique.

Ces dernières règles interdisent des combinaisons qui n'ont aucun sens.

Les **éléments de la représentation sémantique** doivent être définis à un niveau plus profond (bas) que le niveau sémantique qui relie des mots entre eux. Schank lui donne le nom de "**niveau conceptuel**".

Ce niveau conceptuel se différencie du niveau sémantique puisqu'il est indépendant d'un langage particulier.

Schank définit un ensemble d'unités d'abstraction utiles pour la représentation du sens qui remplissent les conditions suivantes :

- unique
- psychologiquement valide : Cette condition reflète la seconde exigence de Schank. Ce modèle doit se conformer aussi fidèlement que possible aux

processus humains. Il est difficile de se rappeler quels mots furent utilisés après un certain temps, lorsque deux phrases dites avec différents mots signifient la même chose. C'est la structure sous-jacente qui est mémorisée.

2.1.3. Formalisme

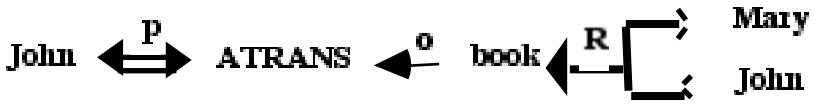
Le niveau conceptuel de Schank comporte différents éléments d'abstraction que l'on peut regrouper en cinq catégories :

- 1 les rôles conceptuels.
- 2 Les catégories conceptuelles
- 3 Les règles syntaxiques de combinaison
- 4 Les règles sémantiques du niveau conceptuel.
- 5 Les différentes inférences

1) Rôles conceptuels

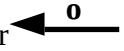
Les **rôles conceptuels** utilisés sont :

Conceptualisation : C'est l'unité de base du niveau conceptuel. Un exemple de conceptualisation est donné dans la figure ci-dessous. Elle représente la phrase « John donna un livre à Mary ».



Une conceptualisation est bâtie à l'aide des notions suivantes :

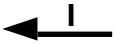
Acteur : celui qui fait (réalise) l'action (l'Act). C'est l'animé qui produit une action. La présence de ce rôle est dénotée par le signe

Objet : ce sur quoi porte l'action. Il est relié par 

à l'action.

Bénéficiaire et Donateur : Le bénéficiaire est le récepteur de l'objet (livre dans l'exemple) sur quoi porte l'action. Dans l'exemple de conceptualisation donnée ci-dessus, le bénéficiaire est Mary et le donateur de l'objet, John. Le symbole utilisé pour lier récepteur et donateur à l'action est : 

Direction : les points de départ et d'arrivée de l'objet déplacé indiqués par 

Instrument : élément (pouvant être une conceptualisation) qui contribue à l'action. Le symbole 

pointe sur l'action dont la conceptualisation est

instrument.

État : état d'un objet. Le signe  relie le couple attribut-valeur à l'objet décrit. Certains états d'objets peuvent être décrits par des échelles avec valeurs numériques. Même si l'homme n'a pas cette manière de décrire ces états, ce choix est justifié par l'implémentation informatique et de l'usage possible des états dans les inférences.

2) Catégories conceptuelles

Ces **catégories conceptuelles** ou **types de concepts** sont utilisées par des règles conceptuelles.

a) Les PPs (Producteurs d'image, *Picture Producers*)

Un PP est un objet physique (pour pouvoir produire une image).

Les PPs peuvent jouer plusieurs rôles dans une conceptualisation. Les PPs qui sont animés ou qui ont des qualités animées (comme les "machines") peuvent être acteurs. (Les forces naturelles, un type spécial de PP (comme le vent), peuvent aussi être acteurs) Tout PP peut jouer le rôle d'objet. Un PP pourrait être écrit dans le rôle de direction, auquel cas il est considéré comme indiquant l'emplacement de ce PP. Les PPs animés peuvent aussi servir de bénéficiaires.

b) Les Acts

Il y en a 11. Tous les verbes peuvent être décomposés en éléments conceptuels basiques, parfois en terme d'un ou plusieurs de ces 11 éléments.

Ces acts peuvent être regroupés dans quatre classes :

• 1) Acts physiques : Propel, Move, Ingest, Expel, Grasp.

PROPEL : Appliquer une force.

MOVE : Bouger une partie du corps.

INGEST : prendre quelque chose à l'intérieur d'un objet animé.

EXPEL : prendre quelque chose à l'intérieur d'un objet animé et le forcer à sortir.

GRASP : attraper physiquement un objet.

• 2) actions mettant l'accent sur le résultat de l'action : Ptrans et Atrans.

PTRANS : changer la position physique d'un objet.

ATRANS : changer une relation abstraite d'un objet.

• 3) actions mettant l'instrument d'action : Speak et Attend.

SPEAK : production de son.

ATTEND : diriger un organe d'un sens vers son stimulus.

• 4) actes mentaux : Mtrans et Mbuild.

MTRANS : transfert d'information d'un individu à un autre ou entre deux éléments de la mémoire d'un individu.

MBUILD : Création de nouvelles pensées.

La manipulation de ces deux types d'action requiert un certain modèle de fonctionnement de la mémoire : celle ci est considérée comme composée d'un processeur conceptuel (CP, c'est là où arrivent toutes les informations conscientes et il ne traite qu'un item à la fois), d'une mémoire intermédiaire (MI, lieu où sont stockés les connaissances servant aux opérations en cours) et d'une mémoire à long terme (MLT, qui contient toutes les connaissances).

c) Les LOCs

Ce sont les emplacements. Tout ACT physique possède un emplacement. Ce dernier, inclus dans une conceptualisation, donne le lieu de l'occurrence de l'événement représenté. Les emplacements sont considérés comme coordonnées dans l'espace. Les LOCs peuvent aussi jouer le rôle de direction.

d) Les Ts

La plupart des conceptualisations ont un temps. Le temps est considéré comme un point ou un segment sur une ligne de temps. Ce point sur ce segment peut être absolu (à 2 heures le 14 Mars 1995) ou relatif (avant hier).

Ces systèmes permettent de situer temporellement une conceptualisation par rapport à des temps (présent, passé, futur) ou par rapport à des débuts ou fins de transitions. Une conceptualisation peut aussi être considérée comme continue dans le temps.

e) D'autres modalités

La dépendance conceptuelle propose aussi d'autres modalités que celles de type temporel comme l'interrogation (?), la négation (/), et le conditionnel (c).

f) Les AAs (ou *Action Aiders*)

Ce sont des modificateurs de traits d'action. (On dit aussi aide à l'action) Par exemple, PROPEL a un facteur de vitesse qui est un AA.

g) Les PAs (ou *Picture Attribute*)

Les PAs prennent la forme : STATE(VALUE).

Un PA est une caractéristique d'attributs (comme la couleur ou la taille) plus une valeur pour cette caractéristique (comme rouge ou 10 pieds). Les PPs peuvent être considérés comme formant un ensemble de PAs qui les définissent. Tout objet physique peut être défini comme un ensemble d'états attributs avec des valeurs spécifiques.

3) Syntaxe du niveau conceptuel

Ces catégories conceptuelles se combinent suivant les règles ci-dessous:

(1) Les PPs peuvent réaliser des actions.

Il s'agit de la description de la relation entre un acteur et l'événement qu'il provoque.

Cette relation peut être dans les deux sens.

(2) Un PP peut être décrit par un attribut PA.

ex : « la main de John » , « la grenouille est dans la main » , « John a 1m80 (de taille) »

main ⟺ PART(John) grenouille ⟺ CONTAIN(main) John ⟺ taille(1m80)

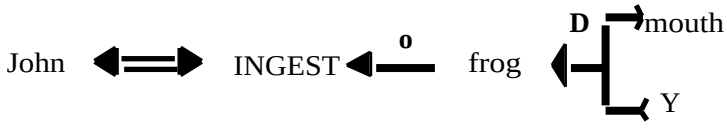
gre grenoui

(3) Les Acts ont des objets.

(4) Les Acts ont des directions.

ex :

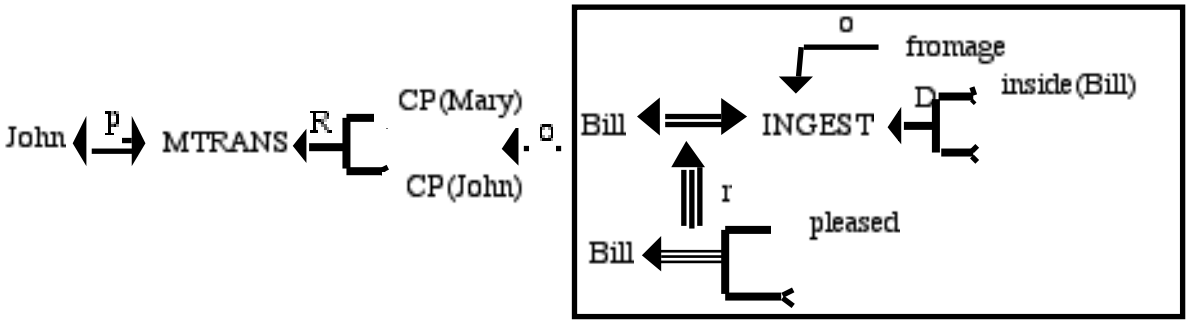
« John mange une grenouille »



(5) Les Acts ont des bénéficiaires.

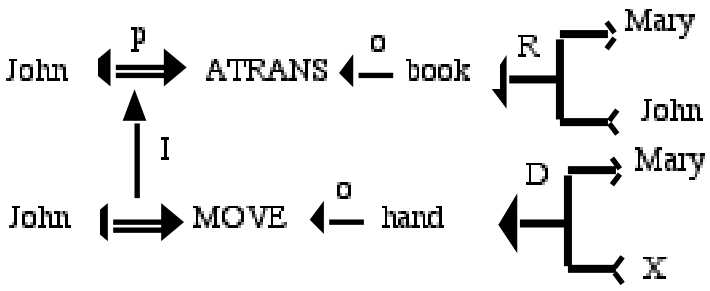
(6) Certains Acts (comme MTRANS) requièrent des conceptualisations ou des combinaisons de conceptualisations comme objets.

ex : « John dit à Mary que Bill aimait le fromage »



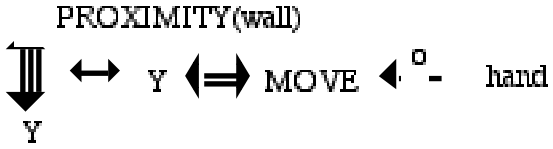
(7) Les Acts ont des conceptualisations comme instruments.

ex :
L'instrument de l'acte de transfert de possession est un mouvement fait par John à sa main et vers Mary.



(8) Les PPs peuvent être décrits par des conceptualisations dans lesquels ils arrivent.

ex :
« la personne Y qui est à proximité du mur et qui bouge sa main »

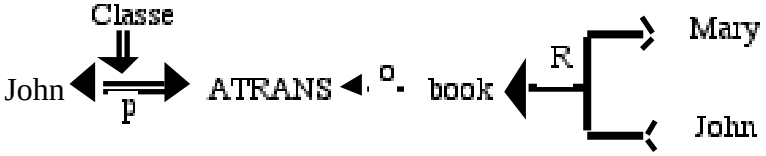


(9) Les Conceptualisations ont des temps.

ex:
« John donna un livre à Mary hier »

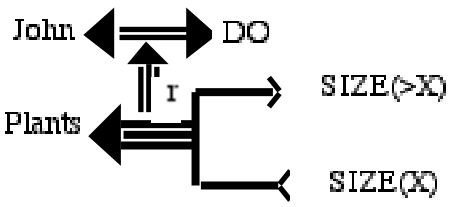
(10) Les Conceptualisations ont des lieus (emplacements).

ex:
« John donna un livre à Mary dans la classe »



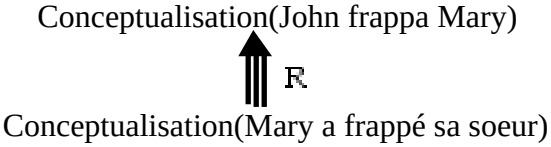
(11) Les Conceptualisations peuvent résulter en des changements d'états pour Pps.

ex :
« John fait pousser des plantes »



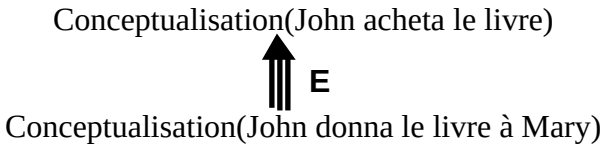
(12) Les Conceptualisations requérant des Acts mentaux peuvent servir de raisons pour des conceptualisations.

ex :
« John frappa Mary parce qu'elle a tapé sa soeur »



(13) Des états ou des changements d'états peuvent rendre possible l'occurrence des conceptualisations.

ex :
« John acheta le livre pour le donner à Mary. »



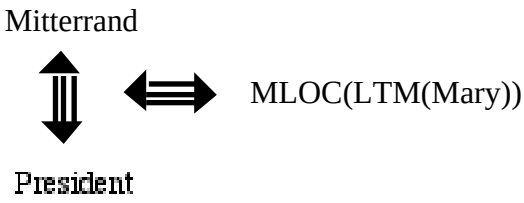
(14) Un PP peut être équivalent à une instance d'un autre PP. L'un des PPs appartiendrait à l'ensemble défini par l'autre.

ex :
« Mitterand est président »

(15) Les Acts peuvent avoir des modifications de traits (ex, vitesse pour des Acts de mouvement)

(16) Une conceptualisation peut avoir un attribut (ici, être située à un emplacement donné de la mémoire).

ex :.
« Mary sait que le président est Mitterand »



4) Sémantique du niveau conceptuel

Des restrictions sémantiques existent sur les objets associés à des ACTs donnés. Elles sont appelées "*conceptual semantics*" par Schank. A chaque ACT est associé un ensemble de restrictions sémantiques.

Ainsi, PROPEL : (Appliquer une force)possède un objet, un acteur et une direction. Cet objet doit être un objet physique. A la différence des autres ACTs, PROPEL permet des acteurs inanimés comme les machines.

Ces restrictions sont vérifiées par le système de représentation par l'intermédiaire de règles. Pour l'ACT PROPEL, une règle teste si la taille de l'objet est petite par rapport à la force exercée dans le but d'établir si l'objet sera dans un nouvel endroit. La direction est celle de la force exercée.

5) Inférences

Une facilitation du raisonnement devrait découler de l'emploi de primitives pour des représentations [Rich, 1983].

Tout d'abord, le nombre de règles d'inférences utilisés est plus restreint . Ainsi, au lieu

de décrire, à chaque fois, pour chaque mot utilisant un certain type d'action, un certain nombre de règles, il suffit de décrire ces règles une seule fois pour chaque Act primitif.

Par exemple, pour les verbes "donner", "prendre", "voler", "faire don" où il y a transfert de possession (ATRANS), il sera facile de décrire des inférences sur l'objet ou les bénéficiaires (le donateur et le récepteur) en travaillant sur la primitive ATRANS, alors que des systèmes travaillant sur des éléments non primitifs devraient établir un ensemble de règles pour chacun de ces verbes.

De plus, le système à base de primitives propose un certain nombre d'inférences qui sont contenues dans la représentation et qui génèrent de nouvelles connaissances réutilisables.

Par exemple, si l'on est amené à coder en Dépendance Conceptuelle (DC) l'information « Bill a menacé John de lui casser le nez », le système générera, sans l'entremise de la création d'une nouvelle règle, l'information « John sait que Bill mettra sa menace à exécution s'il fait quelque chose (qu'il ne faut pas faire) ».

a) Inférence au sens de Schank

Pour Schank, une inférence est toute conceptualisation qui peut être dérivée d'une autre conceptualisation avec une probabilité inférieure ou égale à 1. Une inférence est une conceptualisation qui est **probablement vraie** mais **pas nécessairement vraie** pour une entrée donnée. Cette conception de l'inférence est relativement différente de l'inférence traditionnelle que l'on connaît.

b) Inférences Raisonnables (ou non absurdes)

Si l'inférence ne peut être niée tant que l'on maintient la proposition initiale, on considérera que la conceptualisation inférée fait partie du diagramme de Dépendance Conceptuelle pour la proposition initiale.

Une simple manière d'établir si une conceptualisation appartient au domaine d'inférences, c'est le "test du MAIS ... PAS" (*BUT TEST*).

S'il est absurde de dire "**X MAIS PAS Y**", alors Y fait partie de la Représentation Conceptuelle correcte pour X.

-> John alla à New York mais il n'y est pas parvenu. (**non absurde**)

-> John arriva à New York mais il n'y est pas parvenu (**absurde** car s'y trouver fait partie d'arriver)

Si "**X MAIS PAS Y**" **est raisonnable** et qu'il altère quelque espérance **alors Y est une inférence valide**.

Si "X **MAIS PAS** Y" **est plausible** mais apparemment sans relation (« John dit à Marie qu'il l'aimait, mais il ne mangea pas ce sandwich »), on peut interpréter cela comme une tentative de faire un rattachement entre deux propositions. Quelques propositions sont plus facilement reliables à d'autres. On trouvera par exemple un peu moins bizarre la proposition

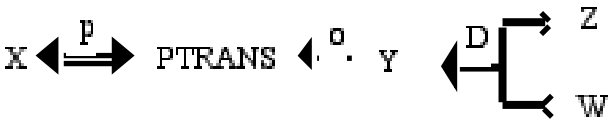
« John dit à Marie qu'il l'aimait, mais il hait sa cuisine ».

Il pourrait être important que le programme de compréhension puisse s'imaginer comment les propositions interagissent. MARGIE détecterait l'inconsistance dans « John dit à Marie qu'il l'aimait, mais il la tapa. ».

c) Inférences Systématiques

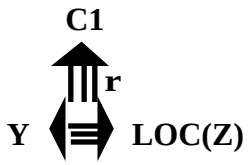
Nous n'évoquerons que les inférences systématiques produites au niveau de PTRANS. Il faut remarquer que pour chaque Act physique (INGEST, EXPEL, GRASP, PROPEL, MOVE) présent dans une conceptualisation, un PTRANS peut être inféré. Cette nouvelle conceptualisation sera à tout point identique dans ces "cas" (excepté le cas instrumental) avec PTRANS comme ACT. Cela donnera une conceptualisation de la forme C1.

C1 :

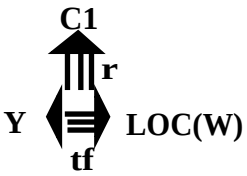


À partir de cette forme C1, il est possible de générer les inférences suivantes :

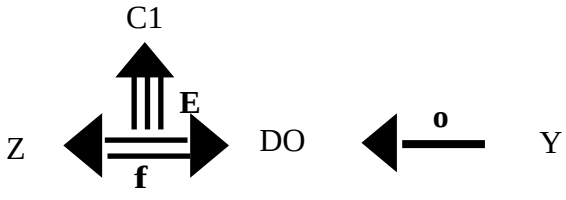
(1) Y est donc situé en Z



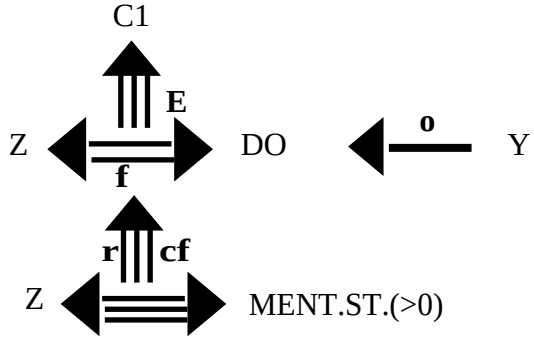
(2) Y n'est plus en W



(3) Si Z est humain et si Z requiert C1
ou si Z = X
alors Z fera probablement ce qui se fait avec Y



(4) On peut ajouter en plus du (3) que cela entraînerait que Z soit content (c-a-d qu'il veut faire tout ce qui se fait avec Y)



Un exemple pour illustrer ces inférences :

John est parti à New York du Texas (*John went to New York from Texas*).

- John devait être à New York. (1)
- John cessa d'être au Texas. (2)
- John voulait faire quelque chose à New York. (3)
- John pensait qu'il serait content d'être à New York. (4)

On notera que ces inférences peuvent être fausses. Toutefois, c'est une partie nécessaire à la compréhension de faire de telles inférences.

d) Inférences arrières

Jusqu'ici, les deux types d'inférences citées pouvaient être considérées du type "chaînage avant". C'est à dire qu'à partir de l'Act ayant pris place, on essaie de décider quelles choses vont résulter lorsque cet Act nécessaire à leurs réalisations.

Il existe deux types d'inférences arrières.

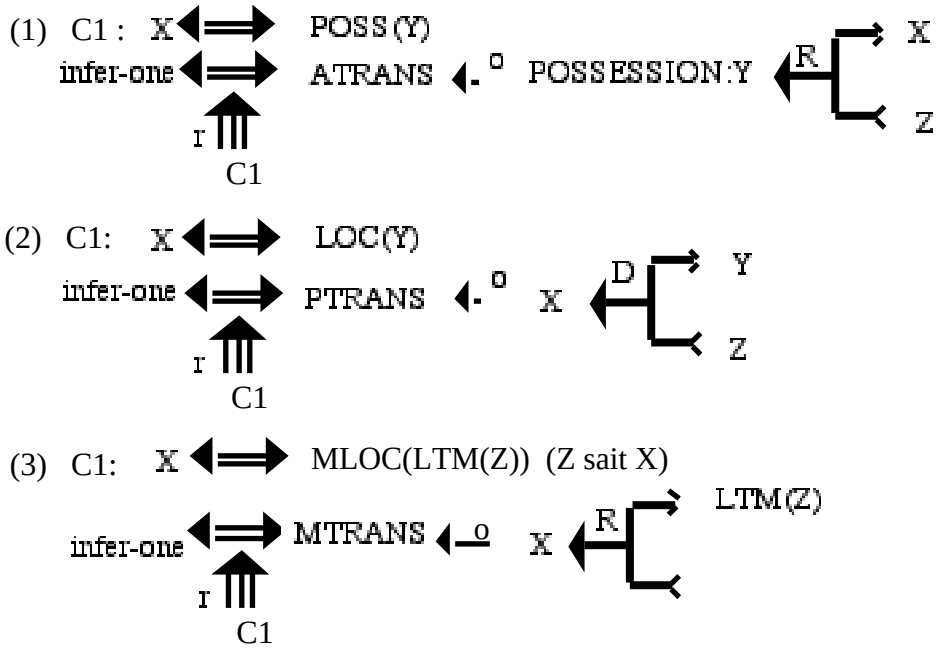
En effet, une conceptualisation peut être

- soit **l'information** (message, renseignement) **du résultat d'une autre conceptualisation** non dite,
- soit **l'instrument d'une autre conceptualisation** non dite.

α) Premier type d'inférence arrière

Si on dit « John a un livre », alors nous savons que quelque chose a du être la cause de cet état. Ainsi, on peut se demander si « Quelqu'un PTRANSa le livre à John ». De même, avec « Mary sait ce que Fred fit », nous devons être capable d'inférer que cette information fut MTRANSé à Mary. Cela permet des questions du genre « Qui le lui a dit ? ».

On aura des règles d'inférences du genre, « Chaque fois qu'une relation d'état existe, un ACT TRANS (un ATRANS ou un PTRANS ou un MTRANS) peut être inféré ».



De (1), si X possède Y, qui le lui a donné ?

De (2), si X se trouve en Y, qui l'a mis à cet endroit ?

De (3), si X est pensé par Z, qui a transféré l'information X à Z ?

β) Second type d'inférence arrière

Cela se passe quand l'instrument est mentionné sans l'Act principal. Supposons l'act X (John donna la balle à Bill). On peut supposer que X pourrait être l'instrument de l'Act Y, c'est à dire que cet Act X aurait pu être réalisé dans le but de permettre l'Act Y duquel il est l'instrument d'arriver. (Y serait le transfert de possession du ballon vers Bill).

Par exemple, chaque fois qu'un PTRANS survient, il est utile de générer la possibilité que ce PTRANS puisse être l'instrument de ATRANS.

Chaque fois qu'il y a une corrélation entre les restrictions sémantiques de l'Act X cité et l'Act Y pour lequel il est l'instrument (John donna à Bill une banane -> Un PTRANS peut être l'instrument d'un INGEST -> De plus, l'objet du PTRANS peut aussi être l'objet du INGEST), on infère que cet Act X a été fait dans l'intention de permettre l'occurrence de Y (on aura Bill mangea la banane)

e) Structure de représentations internes

Avant de décrire les structures permettant de représenter les types de localisation spatiale, il est important de décrire les structures informatiques ainsi que le mécanisme de base utilisé pour l'implémentation des dépendances conceptuelles de Schank.

A chaque entité réelle à propos de laquelle un certain nombre de faits est connu, il est associé une entité unique en mémoire pour la représenter : cette entité est appelée

"*superatom* ".

Ces "*superatoms*" sont généralement générés automatiquement par le générateur d'atomes de LISP et possèdent des noms comme "C0045".

Toute information conceptuelle à propos d'une entité est associée à cette entité via la propriété ASET de LISP.

De plus, chaque conceptualisation est représentée par un superatom.

Dans la figure ci-dessous, les conceptualisations

(PTRANS o o o o) , (LOC o o), (ISA o o)

sont des *superatoms* tout comme l'est SA1 qui est lié à une entité du monde réel.

Les pointeurs (désignés par des o) vers SA1 indiquent la participation de ce dernier aux conceptualisations. De plus, dans une liste (appelée "*occurrence set*"") associée à SA1, le système gère des pointeurs vers ces dernières conceptualisations.

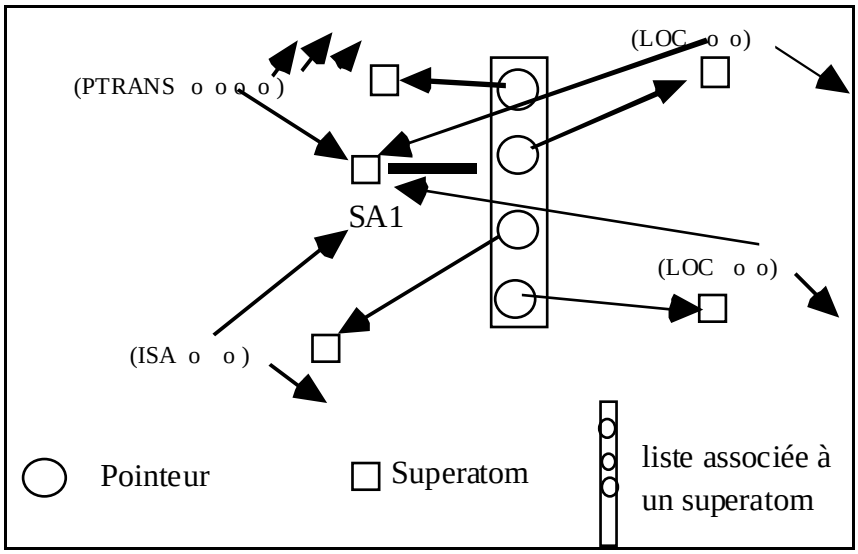


figure 2.8 : Structure interne des dépendances conceptuelles

Cette organisation permet de représenter toutes les conceptualisations attachées à une entité réelle.

Les inférences sont basées sur un mécanisme de recherche appelé "*pattern matching*". Ce dernier peut rechercher des conceptualisations à partir d'une forme donnée. Par exemple, pour savoir qui est parti à NewYork, la forme

(PTRANS (ACTOR ?X) (OBJECT ?X) (TO NewYork))

doit être fournie au système. Ici, la recherche sera rapide car le système recherchera parmi toutes les conceptualisations attachées à l'entité NewYork. Il ne sélectionnera que celles de type PTRANS qui ont pour destination NewYork, et pour acteur /objet la même entité.

La manière de formuler la requête est aisée, et les recherches sont rendues relativement

rapides par cette organisation de l'information.

Néanmoins, cette rapidité est amoindrie si un nombre relativement important d'informations peu intéressantes est généré automatiquement lorsque l'on rajoute une information à la base de connaissances. Ce flot d'informations provient en particulier des inférences systématiques proposées par le système.

f) Quelques critiques générales par rapport à ce système de représentation

Une première critique faite à de tels systèmes concerne le travail nécessaire pour convertir un fait de niveau supérieur en sa forme primitive. Dans de nombreux cas, cette représentation détaillée n'est pas nécessaire.

La seconde critique découle de la première.

La traduction de bas niveau de concepts de haut niveau réclame beaucoup de stockage du fait de l'éclatement en primitives. (figure 2.9)

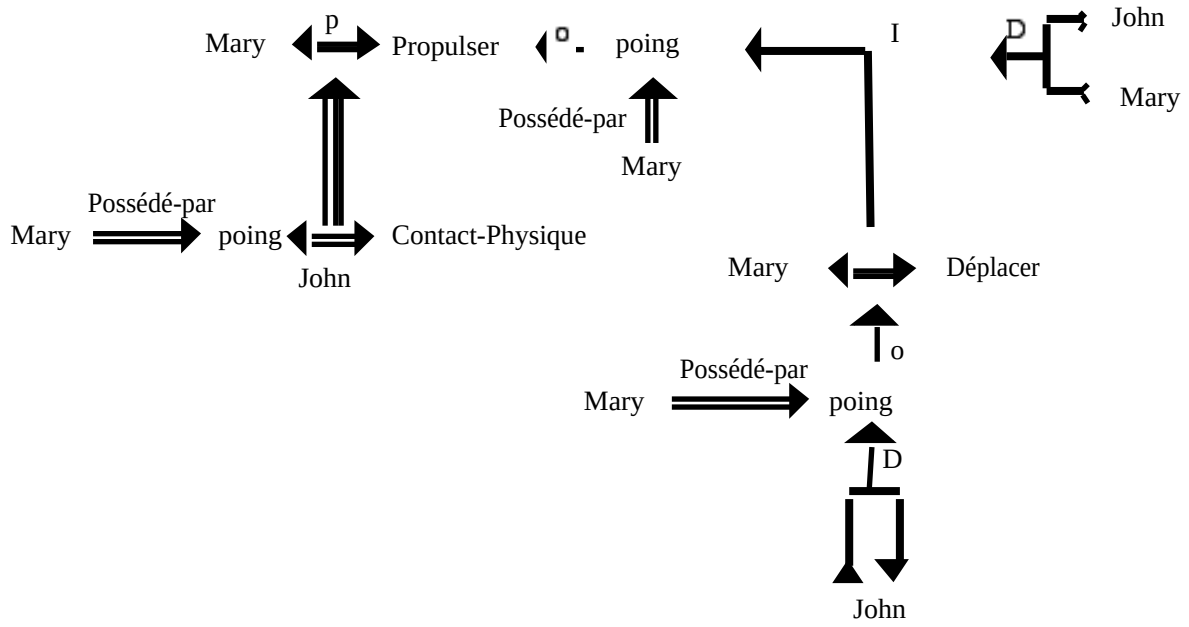


figure 2.9 : Représentation de « Mary a frappé John » [Rich, 1962]

En conséquence, la recherche d'une information d'un fait de niveau supérieur prendrait inévitablement plus de temps que celui consacré à un fait de niveau inférieur, ce qui ralentit d'autant les inférences.

La dernière critique que l'on pourrait formuler concerne le cas où un manque d'information sur un fait de niveau supérieur peut entraîner l'impossibilité de conversion en composantes primitives. Rich (1962) illustre cette difficulté en traitant le sens "cousin" à l'aide de l'ensemble de primitives { père, mère , fils, fille, frère, sœur }.

2.2. Dépendance conceptuelle et problématique SYSQUEST

2.2.1. Représentations

1) Localisation spatiale d'objet "concret"

Nous utiliserons ici la règle de syntaxe du niveau conceptuel n° 2.

L'interprétation en dépendance conceptuelle de la phrase « Nicolas est près de la maison. » se traduit par :

Nicolas ⟷ proximity(maison)

Dans Micro-PAM ([Schank & Riesbeck, 1981]), cette interprétation est représentée par la liste :

```
( prox   (actor      (person (name (nicolas)))
         (location   (maison (owner ()))))
```

La dépendance conceptuelle de Schank met à notre disposition les primitives suivantes : *contain* (dans) , *proximity* (à côté de), *location* (à), *physical-contact, below* (sous)

Un problème apparaît ici :

Il n'existe pas de primitives associées aux relations comme « X sur Y », « X au-dessus-de Y » et « X est entre Y et Z ».

Deux solutions peuvent être envisagées :

La première solution consiste à décrire les relations n'ayant pas de primitives associées à partir des autres primitives.

La seconde revient à créer de nouvelles primitives.

Solution 1 : Description d'une relation à partir des primitives existantes.

Il semble difficile d'exprimer ce type de relation à partir de celles existantes.

Prenons par exemple la relation « X sur Y ».

On ne peut substituer à « X sur Y » l'affirmation « Y sous X » .

Ces deux relations, comme le décrit Vandeloise, ne sont pas converses.

A titre d'exemple, nous pouvons reprendre l'exemple donné par Vandeloise.

La clé est sous la neige (1)

La neige est sur la clé (2).

Dans (1), la cible “clé” est entièrement recouverte par le site tandis que la phrase (2) suggère l’existence de quelques flocons sur la clé.

Une combinaison à partir de primitives existantes ne semble donc, à priori, pas possible pour représenter avec finesse la relation « X sur Y ».

Solution 2 : Création de nouvelles primitives.

La création d’une nouvelle primitive entraîne le problème de son intégration dans le système. Les règles déjà existantes, ont besoin d’être modifiées. Elles sont généralement codées sous forme procédurale et requièrent une connaissance approfondie de l’implémentation du système (connaissance des structures LISP manipulées, de la manière de générer de nouvelles règles, ..) lorsque l’on envisage cette intégration.

En supposant que cette intégration de nouvelles primitives puisse se faire (par exemple avec les relations comme « X sur Y », « X au-dessus-de Y »), il apparaît que la relation « X entre Y et Z »[19] crée un problème supplémentaire.

En effet, l’implémentation d’un système en DC utilise l’hypothèse d’un site unique. Cela signifie que l’intégration d’une primitive associée à “entre” nécessite une ré-écriture d’une partie du code.

Par exemple, dans MicroTaleSpin, l’ajout de cette primitive aurait pour conséquence la nécessité de la redéfinition des fonctions KNOWS-LOC, LOC et WHERE-IS. (figure 2.10)

```
(DE KNOWS-LOC (KNOWER OBJECT)
  (CDPATH '(VAL) (KNOWS KNOWER (WHERE-IS OBJECT))))

^ Returns location of X
(DE LOC (X) (KNOWS-LOC 'WORLD X)
(DE WHERE-IS(X) (LIST 'LOC (LIST 'ACTOR X)(LIST 'VAL '?UNSPEC))
```

figure 2.10

En termes de “*pattern matching*”, une localisation ne se chercherait plus par la forme unique (LOC (ACTOR ?X) (VAL ?V)) mais sous deux formes, la première identique à la précédente, l’autre étant du type (LOC (ACTOR ?X) (VAL1 ?V1) (VAL2 ?V2)).

La fonction WHERE-IS devra donc être modifiée pour tenir compte de ces deux formes. De plus, la modification même des fonctions appelant WHERE-IS peut se trouver

---

[19] Ici, la création d'une nouvelle primitive semble être nécessaire car il semble difficile d'exprimer cette relation à partir d' autres prilmitives.

nécessaire si l'hypothèse de forme unique avait été prise dans l'écriture des fonctions appelantes.

Le fait d'ajouter de nouvelles primitives ne remet pas en cause la dépendance conceptuelle de Schank. Nous remarquons simplement les difficultés d'intégration de nouvelles primitives.

2) Localisation spatiale d'une action

La règle (10) permet d'écrire des conceptualisations qui peuvent être localisées. Ainsi , la structure représentant « John donne un livre à Mary dans la classe », serait traduite par :

```
( ATRANS    (actor  (person (name   (john)))
            (object (book))
            (from   (person(name(john)))
            (to     (person(name(mary)))
            (location (classe )))
```

Si l'action s'était passée près de Bill, nous aurions trouvé dans la base la conceptualisation suivante :

```
( ATRANS (actor  (person (name (john)))
         (object (book))
         (from   (person(name(john)))
         (to     (person(name(mary)))
         (location (near (person (name (bill))))   )
```

Deux problèmes se posent ici.

(1)

Le premier est relatif au manque d'uniformité qui apparaît au niveau des éléments jouant le rôle "*location.*" En effet, on trouve dans les deux conceptualisations précédentes dans le rôle *location* , un PP ou une expression contenant un PP.

Une première initiative d'uniformisation consiste à n'autoriser que des formes comme

```
( ATRANS (actor (person    (name (john)))
                (object    (book))
                (from      (person(name(john)))
                (to        (person(name(mary)))
                (location (in classe )))
```

Une question se pose alors :

Si autour de l'ACT principal ATRANS, collaborent des entités "*John*", "*book*", "*Mary*", à quoi correspondrait le terme "*in classe*" ?

(2)

Le second problème apparaît lorsque nous voulons indiquer plusieurs localisations pour une même action.

Imaginons que l'action précédente se passe aussi près de Bill. La structure suivante pourrait être trouvée :

```
( ATRANS (actor    (person (name (john)))
         (object   (book))
         (from     (person(name(john)))
         (to       (person(name(mary)))
         (location (near bill )))
```

Si l'on analyse cette situation d'une manière analogue à celle menée dans [Wilensky, 1991b], aucune information ne nous permet d'affirmer qu'il s'agit effectivement de l'événement dont on parle.

3) Représentation de localisations quantifiées

Des informations du type « Toute pièce d'une maison se trouve dans la maison. » ne peuvent être représentées par ce système de représentation.

Malgré ces limitations, SAM ([Schank & Riesbeck, 1981]) pouvait utiliser des scripts comme celui du métro qui intègre une forme comme

```
((ACTOR  &PATGRP <=> (*PTRANS*)   OBJECT &PATGRP
  TO (*INSIDE* PART &STATION1)))
```

en utilisant la définition suivante :

```
&PATGRP : CLASS (#PERSON #GROUP)
          DUMMY T
          SFUNCTION (*NONE*)
```

La forme &PATGRP pouvait être unifiée avec des entrées où un PP acteur pouvait être une personne ou un groupe. Ici, la solution revient donc à donner des connaissances sémantiques au "*pattern matcher*" et utilise l'hypothèse de l'appartenance de PPs à une classe conceptuelle donnée (être humain, objet physique, organisation, etc...) Cette solution n'est pas satisfaisante. L'idéal serait d'intégrer dans la dépendance conceptuelle des éléments pour décrire des informations quantifiées.

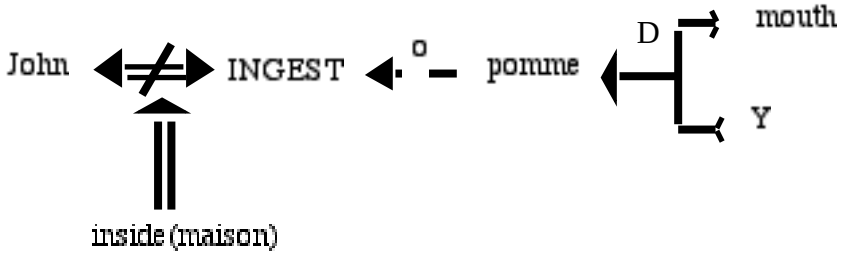
4) Représentation de la négation dans la localisation

Dans la théorie de la dépendance conceptuelle, les modifieurs se trouvent soit entre l'acteur et l'action, soit entre l'objet et son état. Par exemple, l'expression « Nicolas n'est pas dans la maison » est associée à la représentation suivante :

A priori, pour les localisations d'objets concrets, si on suppose le problème des relations de localisations (comme "entre", "sur", etc ...) non implémentées résolu, il ne semble pas y avoir de difficultés majeures.

Par contre, les problèmes apparaissent lorsqu'il faut exprimer la négation au niveau de la localisation des actions.

La conceptualisation ci-dessous



se traduit par la représentation :

```
not(( INGEST    (actor    (person (name (john)))
                (object   (pomme))
                (to       (mouth)
                (location (inside(maison )))  )
```

Ici, la négation porte sur la conceptualisation entière.

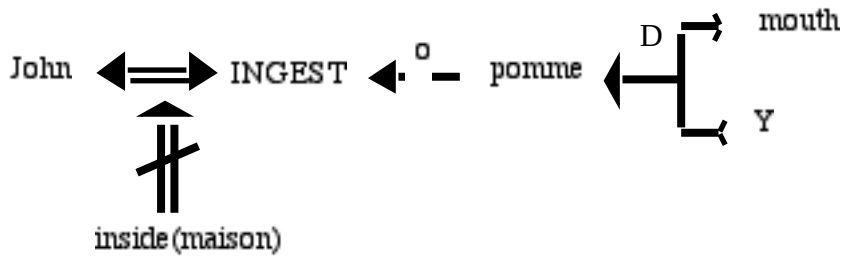
L'expression qui correspond le plus à cette conceptualisation est « John ne mange pas une pomme dans la maison. »

A cette expression, il correspond au moins deux situations différentes :

- le fait que John mange une pomme, mais que cet événement ne se situe pas dans la maison,
- le fait que John fait quelque chose dans la maison, mais qu'il ne s'agit pas du fait de manger une pomme.

La représentation associée semble plus proche de la seconde interprétation qui négativise l'action même plutôt que la relation de localisation elle-même.

La dépendance conceptuelle de Schank rend difficile la création de structures pour représenter la première interprétation que l'on pourrait schématiser comme suit :



Ces problèmes proviennent du fait que la négation porte sur toute la conceptualisation.

Bien que la dépendance conceptuelle soit basée essentiellement sur les événements, Schank ne considère pas un événement comme un “objet” sur lequel des informations seront données. Ici, tout comme les rôles “objet”, “instrument” associées à un événement, la localisation d’une action est plutôt considérée comme une information constituant l’événement, que comme une information supplémentaire à propos de l’événement.

5) Représentation de la localisation et temporalité

Du fait que le temps est pris en compte dans la théorie de la dépendance, nous ne rencontrons aucun problème de représentation de l’aspect temporel dans une relation spatiale.

A la phrase

> « *Sunday morning, Enver Hoxha, the Premier of Albania, and Mrs. Hoxha arrived in Peking at the invitation of Communist China.* »

Schank fait correspondre la structure

```
( (ACTOR TMP32
   <=>  (*PTRANS*)
   OBJECT TMP32
   TO (*INSIDE* PART (#POLITY POLTYPE (*MUNIC*)
                      POLNAME (PEKING)))
   FROM (NIL)
   INST (NIL)
   )
  MODE (MOD1)
  TIME (TIME2))
```

avec

```
MOD1 = (*TF*)
TIME2 = ((WHEN TMP7) (DAYPART MORNING) (WEEKDAY SUNDAY)),
```

TMP7 correspondant à la conceptualisation associé à l’invitation de la Chine.

On trouve aussi des objets temporels définis comme suit :

```
TIME1 : ((BEFORE *NOW* X))
TIME2 : ((VAL GN0))
```

Des objets temporels sont ici rattachés à des conceptualisations.

Le système gère des informations à propos de ces objets temporels. Il s'agit, soit de liaisons par rapport à d'autres objets temporels qui sont soit des dates, soit des objets prédéfinis comme *NOW*, soit de liaison par rapport à d'autres conceptualisations.

2.2.2. Inférences sur ces représentations

De nouvelles informations peuvent être produites grâce à des connaissances figées dans le système. Le système dispose à l'origine de règles d'inférences permettant de déduire de nouvelles informations. Ainsi, par exemple, une inférence systématique et déclenchée par l'entrée de « John alla à New York » pourra permettre la déduction d'une information spatiale liant John à New York à un moment donné.

Conceptuellement, il ne semble pas nécessaire de définir de nouvelles règles d'inférences si l'on suppose que toutes les règles d'inférences possibles sur toutes les primitives décrites et utilisées ont été introduites.

Le problème se pose lorsqu'une certaine finesse d'informations sémantiques est requise et qu'il est nécessaire de réadapter les règles en fonction de cette nouvelle exigence.

Le codage procédural de ces règles d'inférences est fortement lié à une certaine structuration de l'information et rend difficile une mise à jour des règles d'inférence. C'est le cas de l'ajout de nouvelles primitives.

Cette nécessité de connaissance approfondie de l'implémentation du système est un obstacle pour toute extension du système.

2.3. Conclusion

Plusieurs difficultés ont été ici rencontrées :

- difficultés pour l'intégration de nouvelles primitives (que ce soit dans l'ajout de nouvelles règles d'inférences voire même dans la modification de celles existantes),
- difficultés de représenter (ou décrire) plusieurs localisations d'un même événement
- difficultés dues à une intégration non formalisée des relations générales,
- difficultés dues aux restrictions de la négation

De plus, toute information sémantique doit être décrite à l'aide des primitives définies. Toute action relativement complexe devra toujours être décomposée en primitives. Ce qui signifie qu'une action plus complexe que la précédente est décomposée en un nombre plus important d'objets.

3. L'approche par les systèmes terminologiques de Brachman

Cette théorie fut élaborée en prenant en considération les problèmes que posait le manque de formalisme dans les systèmes à base de réseaux sémantiques ([Brachman, 1979]). A la suite de cette étude, Brachman proposa une théorie sur la base d'un formalisme bien défini opérant sur un niveau cognitif de représentation : le niveau épistémologique. Ces idées ont donné lieu à de nombreux systèmes de représentation. (KL-One, KL-Two, Krypton, BACK et le tout dernier Classic)

Notre choix s'explique par la présence d'un formalisme bien défini inexistant dans les systèmes à base de réseaux sémantiques, et par le nombre important de systèmes fondés sur cette théorie. De plus, les aspects d'auto-organisation de la connaissance, de recherche d'informations, les possibilités de descriptions incrémentales sont autant d'éléments que nous avons pris en considération.

3.1. La terminologie de Brachman

3.1.1. Unités cognitives et niveau épistémologique

Après avoir étudié les fonctionnalités des diverses primitives implémentées dans les systèmes à base de réseaux sémantiques, R.J. Brachman [Brachman, 1979] propose une vision de ces systèmes sous forme de niveaux.

NIVEAU LINGUISTIQUE
NIVEAU CONCEPTUEL
NIVEAU EPISTEMOLOGIQUE
NIVEAU LOGIQUE
NIVEAU IMPLEMENTATIONNEL

Figure 2.11

NIVEAU LINGUISTIQUE : les primitives sont prises comme des mots ou expressions spécifiques aux langages. Cela suppose une dépendance langagière de la détermination et la structuration de notre pensée.

NIVEAU CONCEPTUEL : Les primitives (fonctions) utilisées sont soit des concepts primitifs comme les types d'objets ou comme les types d'actions (les GRASP, INGEST, PTRANS de Schank), soit des relations conceptuelles primitives comme les "cas

sémantiques" (ACTEUR, OBJET, BENEFICIAIRE, DIRECTION, ETAT, INSTRUMENT de Schank).

NIVEAU EPISTEMOLOGIQUE : (*Epistemological Level*) Brachman le situe entre le niveau logique et le niveau conceptuel. Il déduit son existence à partir du constat suivant : l'héritage de propriétés ne peut être expliqué par les quatre niveaux linguistique, conceptuel, logique et implémentationnel. Il suppose dans ce niveau l'existence d'une structure interne d'un concept et de relations (différentes des relations logiques) qui rattacheraient les parties de cette description (de concept).

NIVEAU LOGIQUE : Le réseau est un ensemble de primitives logiques. Les nœuds représentent des prédicats et propositions, les liens représentent des relations logiques comme ET, OU, IL_EXISTE.

NIVEAU IMPLEMENTATIONNEL : Le réseau n'est qu'une simple structure de données où les liens sont des pointeurs, et les nœuds, des objets pointés.

3.1.2. Un formalisme à base de concepts, rôles et descriptions structurales

D'après Brachman, un concept est décrit par :

- les concepts le subsumant , c'est-à-dire ses superConcepts
- sa structure interne constituée de rôles et de descriptions structurales.

Un rôle décrit une relation entre les instances d'un concept et les instances d'autres concepts. C'est par son intermédiaire que les propriétés ou les parties d'un concept sont donnés.

Les descriptions structurales donnent aux rôles leur signification en montrant leurs relations.

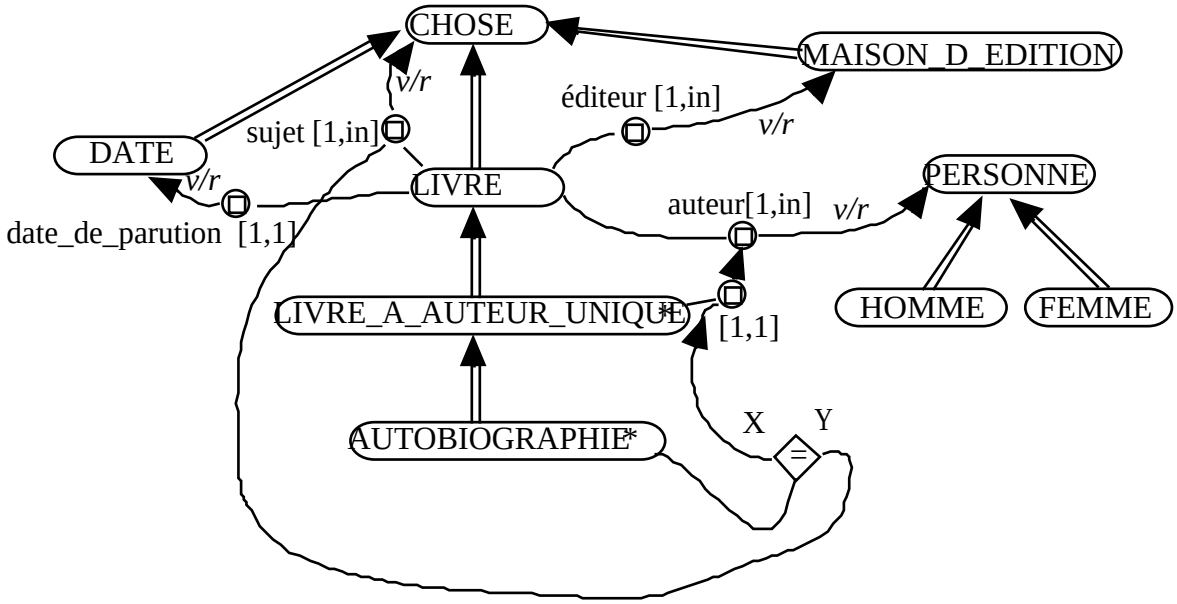


figure 2.12 : Un réseau du niveau épistémologique décrivant les concepts “livre”, “livre_à_auteur_unique”, “autobiographie”

La figure 2.12 est une représentation schématique d’un réseau décrivant trois concepts : livre, livre_à_auteur_unique, autobiographie.

Un livre est décrit par un superconcept “chose”, et quatre rôles “date_de_parution”, “auteur”, “éditeur” et “sujet”. Le concept “livre_à_auteur_unique” possède un superconcept livre et est lui-même superconcept de autobiographie. Le losange contenant “=” est une description structurale appartenant à la structure interne de “autobiographie”. De plus amples explications sont données dans les paragraphes qui suivent.

1) Le Concept et ses superConcepts

Chaque concept peut être défini en terme de ses superConcepts. Il est relié à ces derniers par la relation EST_UN. De cette relation de subsomption, découle une propriété importante : le concept hérite de toutes les propriétés (rôles, descriptions structurales) de ses superConcepts. Dans le réseau ci-dessus, les rôles date_de_parution, auteur, éditeur et sujet appartiennent aussi à livre_à_auteur_unique et autobiographie.

Un concept correctement décrit possède au moins un superConcept. Il peut exister des concepts sans restrictions et défini comme la conjonction de leurs superConcepts (ex : un homme est un être humain et un animal de sexe mâle).

La structure du réseau formé par les relations de subsumation entre les concepts est une sorte de taxinomie qui a pour origine directe cette relation d’ordre partielle. A l’intérieur de cette structure hiérarchique, on peut trouver deux types de concepts :

les concepts génériques et les concepts individuels.

Les concepts génériques désignent un type général au travers duquel plusieurs objets du monde peuvent être décrits (ex : homme, femme, animal). Tout concept générique peut être formé à partir d'autres concepts. Homme est un sous-concept générique de animal.

Les concepts individuels, comme leur nom l'indique, individualisent les concepts génériques et ne devraient pas posséder de sous-concepts. Le concept individuel dénote au plus un objet réel qui est une instance du concept générique.

Notion de concept défini et de concept primitif

Les concepts génériques sont classés en deux catégories :

les **concepts primitifs** et des **concepts définis**. [20]

Prenons le cas d'un concept dont on dit qu'il est le sous-concept à la fois de être_humain et du concept animal_de_sexe_mâle. Le système est capable de déduire que cette même instance est instance du concept homme. Un concept est défini s'il est entièrement déterminé par sa structure interne. Ses parties spécifient des conditions nécessaires et suffisantes.

Un concept primitif est partiellement déterminé par sa structure qui ne fournit que des conditions nécessaires.

Si on prend les deux définitions qui suivent :

- un être humain est un mammifère,
- un homme est un être humain de sexe mâle,

on peut dire que le concept homme est un concept défini car toute chose possédant à la fois le type être humain et le sexe mâle est considéré comme un homme. Par contre, tout mammifère n'est pas un être humain. Évidemment, il est possible de décrire l'homme comme n'étant qu'un être humain. Dans ce cas, être humain n'est pas suffisant pour toute chose pour devenir un homme.

Bien sûr, c'est à l'utilisateur seul que revient le droit de statuer le fait que la définition qu'il donne est complète ou non. Dans le réseau précédent, seuls deux concepts ont été déclarés comme définis, le concept livre_à_auteur_unique et le concept autobiographie qui ont d'ailleurs été soulignés dans le but de les distinguer des autres.

---

[20] Un concept individuel est toujours primitif.

[21] Il faut noter que le nom que nous associons au rôle n'est qu'une seule chaîne de caractère à aspect mnémonique. Ici, les SDs semblent rattacher un peu de sens à ces rôles.

2) Les Rôles

Dans les anciennes implémentations de réseaux sémantiques, la description d'un nœud se construit au fur et à mesure par l'adjonction de paires attribut-valeur. De cette notion d'attribut-valeur, on retient tout d'abord le fait qu'il n'existe aucune information sur la forme que doit prendre la valeur d'un attribut associé à un nœud. De plus, rien n'indique à quel type de nœud, un attribut donné peut être attaché. Le concept de rôle semble pallier à cette insuffisance et c'est ce qui lui a valu d'être appelé "description d'attributs généralisée" ([Brachman, 1977]). On décrit non seulement les "valeurs" possibles d'un attribut mais aussi le concept sur lequel cet attribut peut être appliqué.

Le rôle a pour but de capter "ce qui contribue" à la définition du concept. Il n'est pas suffisant de décrire un concept uniquement en donnant les entités qui le subsument. Il faut pouvoir le distinguer d'autres (en particulier, ceux qui posséderaient les mêmes subsumeurs).

Les Rôles permettent de spécifier les propriétés ou les composantes d'un concept. Plus précisément, ils décrivent des relations entre instances du concept (auquel le rôle est attaché) et d'autres instances d'autres concepts.

Dans notre cas, un livre est un objet. Pour le différencier des autres objets, on définirait alors la relation que le livre possède avec le concept date : c'est la date de parution. Tout livre possède une date de parution. Le rôle auteur relierait un livre à une ou plusieurs personnes.

Le rôle représente une relation binaire entre deux concepts qui sont appelés respectivement **concept domaine** et **concept rang** du rôle. Le rôle est dit "attaché" à son concept domaine.

La structure d'un rôle est la suivante :

- son nom rappelant sa fonctionnalité,
- un concept domaine qui est son domaine d'application,
- et une description des entités potentielles jouant ce rôle (Nous appellerons ces entités "**remplisseurs de rôles**", traduction du terme anglais *role fillers* utilisé par Brachman) pour chaque instance du concept domaine.

Ces remplisseurs de rôles sont décrits à l'aide

- d'une restriction de valeur (notée V/R) qui donne le type de ces entités,
- et d'une restriction en nombre, sous forme de paire de nombres [n1, n2] avec :

  $n1 <= n2$

  $n1 \in N$ (l'ensemble des nombres entiers)

  $n2 \in N \cup \{ in \}$

n1 (respectivement n2) est la cardinalité minimum (respectivement maximum) des individus (du concept rang) associés à un individu du concept domaine.

La valeur possible "in" (comme infini) pour n2 est due au fait que la cardinalité

maximum n'est pas connue. Si on considère par exemple le rôle AMI associé au concept HOMME, n2 ici vaudrait in, car on ne connaît pas le nombre possible d'amis de tout homme.

a) Héritage

Si un rôle est attaché à un concept donné, alors tout sous-concept subsumé hérite de ce rôle. Dans notre exemple, livre_à_auteur_unique subsumé par livre, possède tous les rôles de ce dernier, à savoir auteur, éditeur, sujet et date de parution. De plus, autobiographie possède non seulement les rôles de livre, mais aussi la restriction de rôle (que nous détaillerons par la suite) au niveau auteur appartenant à livre_à_auteur_unique. On peut se demander si certains rôles peuvent être ou non hérités par les sous-concepts subsumés. Ici, le problème dû aux exceptions ne se pose plus, tout simplement parce que Brachman considère, qu'y faire recours est dû à une mauvaise description de concepts. Leur usage n'est donc pas permis sur les implémentations de tels systèmes.

En ce qui concerne les rôles, certaines possibilités supplémentaires sont apportées ; comme le fait que tout rôle peut être restreint (un rôle restreint est appelé restriction de rôle) ou différencié en un (ou plusieurs) rôle(s) différentiateur(s).

b) La restriction de Rôle

Elle permet de créer de nouveaux concepts. En effet, un concept ne possédant qu'un superConcept est bien défini s'il en est différencié par certaines caractéristiques. Le rattachement d'un nouveau rôle est une solution. La restriction de rôle en est une autre, elle permet de restreindre les remplisseurs de rôle soit au niveau du type (le nouveau type est alors un sous-concept du concept rang), soit au niveau des deux cardinalités. Les sous concepts héritent bien sûr des autres rôles qui ne sont pas restreints. Dans l'exemple, le concept livre_à_auteur_unique se différencie du concept livre par la seule restriction du rôle auteur créée par affinement de la cardinalité maximum ramenée à 1. On peut décrire le concept "livre-écrit-par-une-femme" en utilisant le concept "livre-à-auteur-unique" et en restreignant les remplisseurs du rôles "auteur" au type "femme".

c) La différenciation de Rôle

Un rôle peut être différencié en un ou plusieurs rôles différenciateurs. Cette différenciation provient du fait que les remplisseurs de rôle pour un rôle donné peuvent être distingués entre eux. Considérons par exemple le rôle "enfant" attaché au concept "animal". Le type des remplisseurs de rôle est "animal". Or ces derniers peuvent être séparés en deux catégories : les fils et les filles . Le rôle enfants est différencié en deux : le rôle "fils" et le rôle "fille".

La différenciation de rôle crée ici une taxinomie entre les rôles. Dans l'arbre formé, le rôle différencié est père du rôle différenciateur. Cette différenciation est aux rôles ce que la spécialisation est aux concepts génériques. Les rôles différenciateurs héritent de la restriction

de valeur et de la cardinalité maximale de son rôle père. Tout remplisseur de rôle remplissant la fonction de “fils”, remplit aussi la fonction de “enfant”.

3) Les descriptions structurales (“SDs”)

Une autre solution pour différencier un concept d’un de ses superConcepts est de définir des relations non plus de concept à concept comme le font les rôles, mais des relations entre les rôles[21]. Ces super Rôles portent le nom de descriptions structurales.

Pour le concept autobiographie, il faudrait décrire la relation qui existe entre l’auteur de l’ouvrage et le sujet du livre.

Les descriptions structurales permettent également de rattacher les composantes d’un concept pour en former un tout. Si on reprend le concept traditionnel de l’arche, il ne suffit pas de décrire les composantes de l’arche comme l’ouverture, le linteau et les deux piliers, il faut aussi pouvoir dire que le linteau se trouve sur les deux piliers, que l’ouverture a une taille dépendante de la distance du linteau au sol.

a) La “*Role Value Map*” (“RVM”)

C’est la plus simple des SDs. Elle permet de donner l’égalité de deux ensembles de remplisseurs de rôles. Pour le cas de l’autobiographie, il suffit de dire que le sujet du livre est en fait l’auteur de ce même livre.

La RVM requiert alors à toute instance d’autobiographie la propriété suivante: l’ensemble des objets qui sont les sujets du livre est le même ensemble des personnes qui sont auteurs.

Les deux ensembles que l’on dit égaux peuvent être définis par l’intermédiaire de ce que Brachman appelle “chaînes de rôles” nommés respectivement X et Y. Pour décrire autobiographie, on utilisera les deux chaînes suivantes : X = [auteur] et Y = [sujet].

b) Les autres descriptions structurales :

Elles expriment les relations entre rôles d’un concept donné en termes d’autres concepts. Elles sont décrites dans [Brachman & Schmolze, 1985]. Malheureusement, elles n’ont pas été intégrées aux divers systèmes réalisés à partir de ce formalisme en raison de la difficulté de leur implémentation.

3.1.3. Inférences

Le classifieur est le mécanisme essentiel d’inférences de systèmes de type KL-ONE. Il s’appuie sur la notion de “concept défini” et permet d’établir les relations de subsomption entre un objet donné et d’autres objets par rapport à une structure de taxinomie gérée et mise à jour par le système.

En déterminant les relations hiérarchiques des concepts à partir de leurs propriétés

exprimées en terme de rôles, de restriction ou de RVM, le système place le concept à sa meilleure place.

De plus, dans BACK (un système de type KL-ONE), les instances de concept, peuvent bénéficier de ce classement. En effet, le système est capable de connaître le concept le plus spécifique qui dénote une instance donnée.

3.1.4. Systèmes terminologiques basés sur ce formalisme

Les concepts de BRACHMAN ont été à l'origine de plusieurs systèmes :

- KL-ONE ([Brachman, 1977]),
- KRYPTON ([Brachman et al. , 1983]),
- KL-TWO ([Moser, 1983]).
- Argon ([Patel-Schneider, 1984])
- SB-ONE ([Kobsa, 1991])
- BACK ([Von Luck, Nebel, Petalson & Schmiedel, 1986])
- CLASSIC ([Brachman & al., 1991])

On trouvera une description plus précise de trois de ces systèmes de représentation (Krypton, KL-TWO et BACK). Nous verrons que ces systèmes sont relativement différents malgré une origine commune des idées de départ.

Ces systèmes sont classés sous la dénomination "systèmes de type KL-ONE" en raison de la précédence du système KL-ONE par rapport aux autres.

Ces systèmes présentent un même point commun. Ils appartiennent tous à la famille des systèmes dits hybrides. En effet, ces systèmes se présentent en deux composantes distinctes :

- la TBox (*Terminological Box*) ou composante terminologique,
- la ABox (*Assertional Box*) ou composante assertionnelle.

Cette distinction provient du fait que la description de concepts est dissociée de l'assertion de faits à propos de ces concepts. Quand on décrit un concept, on ne fait pas d'affirmations à propos de l'existence d'objets réels dénotés. Ainsi, on peut décrire le concept d'un "homme de Mars" sans pour autant affirmer l'existence d'un tel homme.

1) Krypton

Krypton résulte de la combinaison d'un démonstrateur de théorème basé sur la logique du premier ordre avec un mécanisme fondé sur un raisonnement terminologique issu du formalisme proposé dans [Brachman, 1979] .

Krypton est composé de deux modules : la TBox et la ABox.

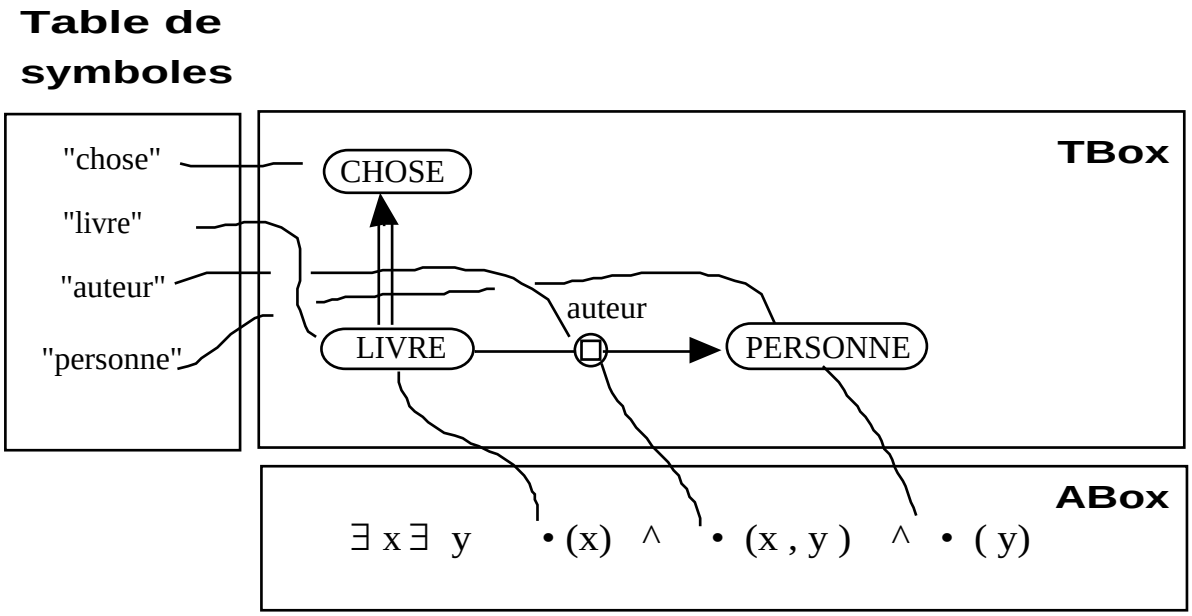


figure 2-13 : Le système Krypton.

La TBox de Krypton permet de décrire les concepts, les liens hiérarchiques entre ces derniers ainsi que les différents rôles entre concepts. Cette composante intègre une forme de raisonnement, appelée “raisonnement par classification”.

Le démonstrateur de théorèmes fait office de composante assertionnelle. C’est par son intermédiaire que l’on asserte l’existence d’objets réels dénotés.

La coopération entre ces systèmes se passe de la manière suivante. Chaque prédicat unaire et binaire (La ABox de KRYPTON ne travaille pas sur les prédicats possédant plus de deux arguments.) de la ABox peut être vu comme un pointeur vers un concept ou un rôle de la TBox. Les prédicats utilisés dans la ABox diffèrent de ceux utilisés dans le calcul des prédicats par le fait qu’ils ne sont pas indépendants entre eux. Ils entretiennent entre eux des dépendances issues des relations existant entre les éléments de la TBox.

2) KL-TWO

KL_TWO [Vilain, 1985] est issu de la combinaison de deux systèmes :

- NIKL (composante terminologique due à Moser et basée sur le formalisme de représentation proposé par Brachman),
- PENNI, la composante assertionnelle (une réimplémentation du système RUP - Reasoning Utility Package - de McAllister). Elle est essentiellement le sous-ensemble propositionnel du calcul de prédicats du 1er ordre. Le langage de

PENNI ne contenait aucune quantification.

NIKL produit, grâce au mécanisme de classification, des axiomes quantifiés similaires à :

$\forall$ x , (C2 x) =>(C1 x) qui signifie le concept C1 subsume le concept C2

à partir des expressions suivantes

(Let HOMME (CMEET ETRE_HUMAIN ANIMAL_MALE))

(Let FILS (VRDiff ENFANT ANIMAL_MALE))

correspondant sémantiquement aux axiomes

$\forall$ x (HOMME x) <=> (ETRE_HUMAIN x) & (ANIMAL_MALE x)

$\forall$ x, y (FILS x y) <=> (ENFANT x y) & (ANIMAL_MALE y)

3) BACK

A l'instar des bases de connaissance, ce système possède des fonctions capables d'ajouter de l'information (ce sont les fonctions TELL) et d'autres permettant d'en extraire (ce sont les fonctions ASK). Ces fonctions sont présentes dans les deux boîtes du système.

La TBox de BACK va nous permettre de décrire des concepts et des rôles. Comme dans beaucoup de boîtes terminologiques issues des idées de Brachman, les exceptions ne sont pas admises. Il faut noter que cette boîte ne possède qu'un seul type de "description structurale", la RVM. Elle reprend l'idée du marquage de la disjonction entre concepts existant chez KL-TWO et KRYPTON. Cette boîte permet la généralisation de concepts et introduit la notion de concept défini de manière extensionnelle grâce aux concepts "Ensembles d'attributs".

La ABox est la partie décrivant les faits du monde réel. Elle est en relation étroite avec la TBox. Les autres ABox des systèmes hybrides n'étaient ni plus ni moins que des démonstrateurs de théorèmes . La TBox et la ABox de Krypton et KL-TWO présentent l'inconvénient d'être conçus de manière indépendante, dès lors les assertions n'utilisent pas la potentialité des concepts définis dans la TBox. Ainsi, par exemple, la restriction en nombre de NIKL (TBox de KL-TWO) n'avait aucun impact sur PENNI (ABox de KL-TWO). Pour la ABox étudiée ici, il n'en est pas de même, les auteurs du système ayant cherché à respecter un principe qu'ils ont dénommé "expressivité équilibrée".

De plus, il est possible de représenter des faits incomplets. Ainsi, une connaissance comme « Pierre ou Jacques font partie du groupe formé par Jean et Anne ».

3.2. Systèmes type KL-One et problématique SYSQUEST

3.2.1. Représentations

1) Localisation spatiale d'objet "concret"

Un système utilisant le formalisme de Brachman s'appuie sur une organisation de connaissances à base de concepts et de rôles. Ici, le premier travail consiste à définir des éléments d'organisation des concepts de localisation spatiale.

La notion de localisation spatiale peut être captée par un concept appelé LOC_SPATIALE. Toute localisation spatiale appartenant à un monde réel ou au monde du discours se traduirait par une instanciation de ce concept.

Deux rôles CIBLE et SITE permettront de caractériser les éléments qui contribuent à la localisation spatiale. Les instances associées à l'instance de LOC_SPATIALE sont des instances du concept OBJET.

Sur cette base, deux organisations de concepts sont possibles.

❶

Une première organisation possible est décrite par la figure 2.14.

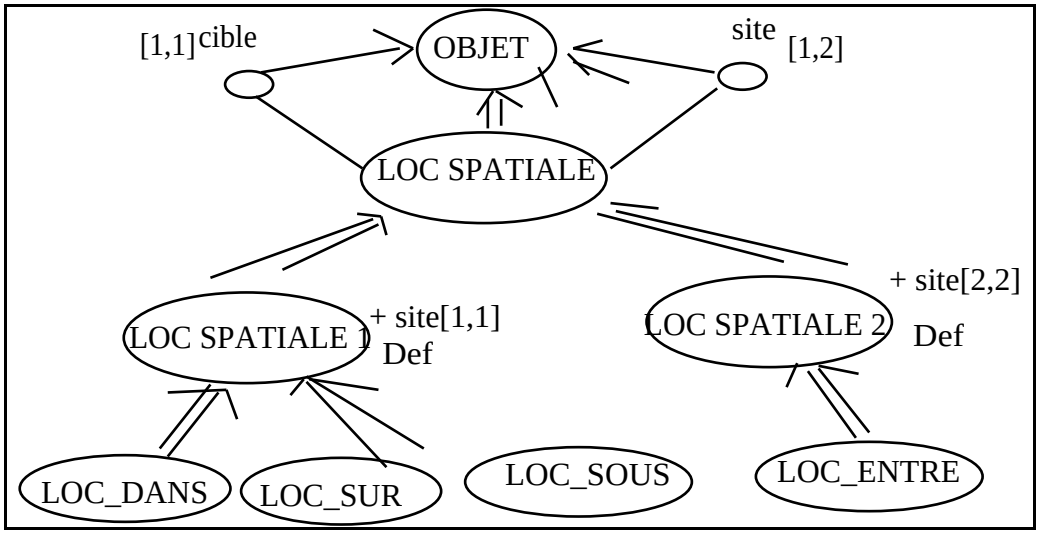


Figure 2.14

Le concept LOC_SPATIALE est un concept générique.

Associé à ce concept, deux rôles déterminants : la cible et le site de la localisation spatiale.

Le concept de LOC SPATIALE possède deux sous-concepts de type défini : LOC_SPATIALE1 et LOC_SPATIALE2 permettant de différencier les localisations spatiales possédant un ou deux sites distincts.

LOC_SPATIALE1 a pour sous-concepts primitifs :

LOC_DANS, LOC_SUR et LOC_SOUS.

LOC_SPATIALE2 a pour sous concepts primitifs : LOC_ENTRE.

Pour exprimer une information comme « Nicolas est dans la maison. », nous devrons créer un instance (idans1) du sous-concept LOC_DANS.

Les instances nicolas1 et maison1, respectivement associées aux entités réelles Nicolas et sa maison, seront respectivement les "*role fillers*" cible et site de idans1.

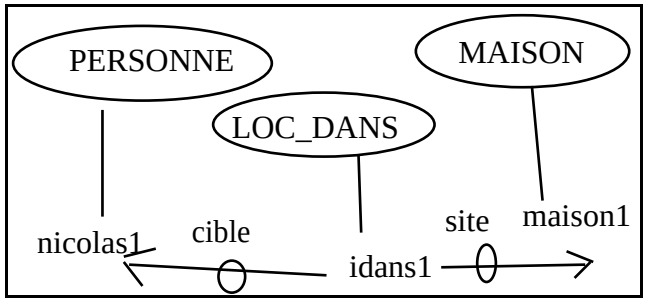


figure 2.15

Cette manière d'organiser les informations comporte quelques inconvénients.

A chaque représentation de localisation spatiale utilisant une relation spatiale (RS) différente, l'utilisateur ou le système doit savoir quel sous concept précis est associé à la RS utilisée.

❷

Une seconde organisation possible des concepts est donnée dans la figure ci-dessous.

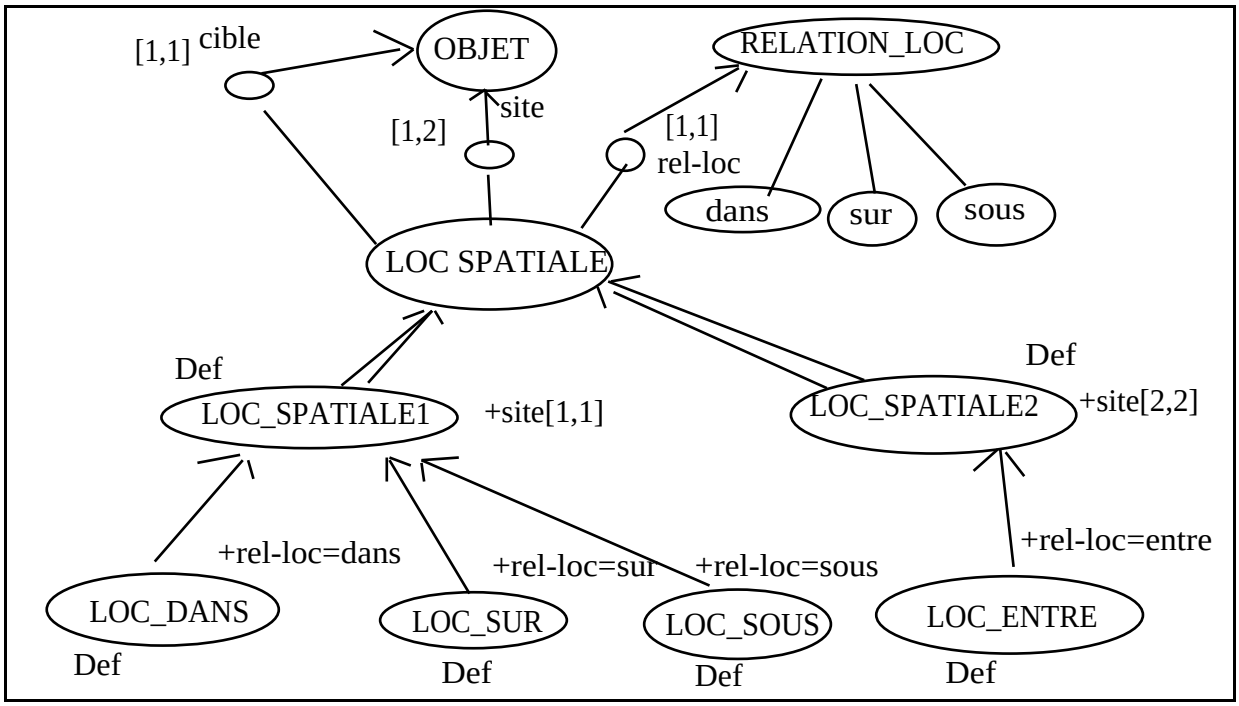


figure 2.16

Dans ce cas, le concept LOC_SPATIALE possède trois rôles : la cible, le site et la relation. Les sous-concepts de RELATION_LOC sont tous des concepts individuels.

Les sous concepts LOC_DANS, LOC_SUR, LOC_SOUS et LOC_ENTRE sont tous des concepts définis par une restriction du domaine rôle “rel-loc“ au sous-concept individuel (de RELATION_LOC) correspondant (dans, sur, sous, entre)

Cette seconde organisation permet grâce aux mécanismes de classification une relative souplesse d’utilisation.

Pour représenter « Nicolas est dans la maison », l’utilisateur a deux modes d’introduction d’informations :

- le premier mode consiste à créer une instance de LOC_DANS et à lui associer les instances jouant les rôle cible et site dans cette localisation.
- le second mode, plus souple, (voir figure ci-dessous) consiste en la création d’une instance du concept générique de LOC_SPATIALE et à lui associer les instances jouant les rôles cible, site et rel-loc. Le mécanisme de classification automatique présent dans les systèmes de type Brachman ira immédiatement classer cette instance comme une instance de LOC_DANS.

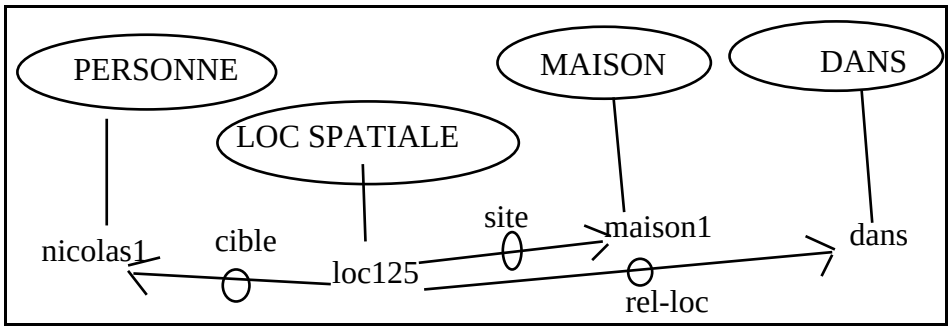


Figure 2.17

Cette classification automatique permet une plus grande rapidité lors de la recherche d'informations précises à propos d'une relation spatiale donnée.

Si l'on désire rechercher des informations à propos de l'endroit où se trouve Nicolas, le système parcourt toutes les instances du concept générique LOC_SPATIALE.

Dans BACK, il suffit d'interroger la base à l'aide de la requête :

:- abox-ask(X - all(X, loc_spatiale(cible = nicolas1))).

Cette efficacité diminue lors de la recherches d'informations spatiales à laquelle Nicolas participe. Dans ce cas, il faut préciser explicitement les différents rôles que peut prendre Nicolas.

2) Localisation spatiale d'une action

Un événement est représenté par une instance du concept ACTION. Ce dernier possède la structure suivante :

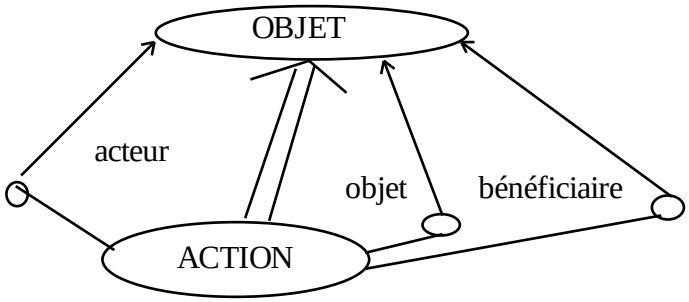


figure 2.18

La localisation d'une action s'effectue par la création d'une instance de localisation dont la cible est l'instance associée à cette action (figure 2.19).

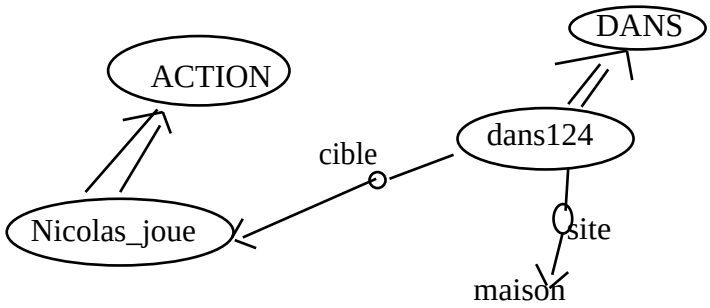


figure 2.19

Lorsque cet événement possède une autre localisation, nous avons une autre instance de localisation avec le même remplisseur de rôle cible.

Pour situer une action dans l'espace, il suffit de rechercher toutes les instances de LOC_SPATIALE ayant cette action pour remplisseur de rôle cible.

Un raisonnement tel que « Si X est l'acteur d'un événement E et que E est situé dans le lieu L, alors X est situé dans le lieu L » est possible autour de ces actions. Ce type de connaissance ne peut être décrit en termes de concepts, rôles et descriptions. Cette difficulté peut être résolue par un mécanisme qui devrait permettre la création d'instances correspondant à la conclusion donnée. Si ce mécanisme est systématique, le système risque de déborder sous le nombre important d'instances inférées.

3) Représentation de localisations quantifiées

Soit la phrase « une chambre de la maison est dans la maison ». Il s'agit ici d'une connaissance d'ordre terminologique puisqu'elle décrit des relations intrinsèques entre deux éléments.

En supposant qu'il existe une solution à ce problème, il faudrait, qu'au moment où l'on déclare qu'une instance de chambre donnée fait partie d'une instance de maison, que le système crée de lui-même une instance de LOC_SPATIALE mettant en relation ces deux instances.

4) Représentation de la négation dans la localisation

La négation n'est pas traitée dans le formalisme de Brachman.

Une solution est envisageable pour l'intégration de localisations spatiales négatives.

Nous pouvons supposer que le concept de LOC_SPATIALE utilisé dans les exemples est en fait séparable en deux sous concepts distincts LOC_SPATIALE_POS et LOC_SPATIALE_NEG.

Lors de l'introduction d'information, l'utilisateur /système doit préciser s'il s'agit d'une localisation positive ou négative.

Nous aurons alors le graphe suivant :

figure 2-20

Pour indiquer que « Nicolas n'est pas dans la maison. », il est nécessaire déclarer l'existence d'une instance de LOC_SPATIALE_NEG.

Si l'information négative doit être explicitée par la création d'une instance d'un concept négation d'un autre, on peut se demander s'il faut que toutes les informations négatives assertées ou déduites doivent se trouver dans la base d'instances. Dans l'affirmative, le problème qui surviendrait sera liée à la multiplication d'instances de concepts.

5) Représentation de la localisation et temporalité

Il faut ajouter deux rôles supplémentaires pour le concept de localisation. Chacun de ces rôles restreint le co-domaine au concept REPERE_TEMPOREL .

L'un permet d'indiquer le début d'un événement, l'autre la fin.

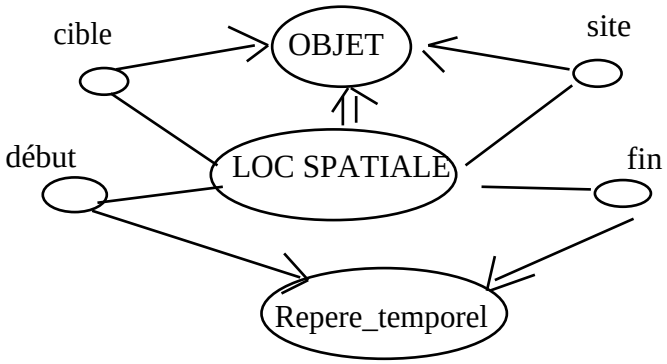


figure 2-21

Une grande précaution devrait être prise au moment du choix du concept de repère temporel. Dans le cas où des entités mathématiques seraient choisies, des problèmes peuvent être rencontrés, comme l'affirme Brachman.

« *Because of its object-centered nature, CLASSIC is likely to be cumbersome to use in cases where mathematical entities such as tuples, sequences, geometric entities, etc are the center of attention. This is because such entities usually have a notion of "equality" based on*

(recursive) component identity. For example, calendar dates are structured objects, and it seems natural to model them as CLASSIC individuals with trois attributes: day, month and year. However, object identity may provide surprising results.. »

3.2.2. Inférences autour de ces représentations

Les inférences principales fournies par le système se limitent principalement au mécanisme de classification présent dans toutes les boîtes terminologiques. La base de connaissances terminologique doit être décrite en tenant compte de ces fonctionnalités.

Hélas, ce mécanisme ne suffit pas pour déduire de nouvelles informations spatiales que l'on désirerait obtenir.

D'autres alternatives ont été proposées, comme les règles dans CLASSIC. Mais, elles ne se révèlent pas suffisantes pour exprimer les conditions complexes des antécédents.

La classification est la principale composante inférentielle dans la boîte terminologique. Cela entraîne que la base de connaissance doit être écrite en prenant en considération ce mécanisme.

3.3. Conclusion

Malgré un formalisme bien défini, de nombreuses difficultés ont été rencontrées dans la résolution des problèmes posés par SYSQUEST.

Ces limitations sont en partie dues, comme le constate Brachman, à une vue centrée objet des éléments d'information.

Nous terminerons cette conclusion par cette phrase de Brachman.

« *Finally, CLASSIC and its relatives have general (weak) reasoning procedures, and do not support the direct and efficient addition of specialized kinds of inference. This means that application needing to make intensive use of temporal reasoning or spatial reasoning, for example, would find it difficult to have CLASSIC deduce the desired relationships* . »

4. L'approche par les systèmes à base de graphes conceptuels de Sowa

Ce mode de représentation est intéressant par l'aspect sémantique généralisé proposé.

Il est basé sur un formalisme bien défini.

Ce modèle est utilisé pour représenter des énoncés en langage naturel (sens des mots, et des phrases). Il semble suffisamment puissant pour prendre en compte certains phénomènes linguistiques du langage naturel. (gestion de la notion de contexte, ...)

Ce formalisme est l'un des axes choisis par le programme de recherches coordonnées en Intelligence Artificielle (PRC-IA). Il bénéficie d'une large collaboration de recherches très récentes (formalisation d'un modèle étendu à travers des comparaisons par rapport à d'autres systèmes, construction de systèmes informatiques, ...)

4.1. Description du formalisme de Sowa

4.1.1. Définition d'un graphe conceptuel

Un graphe conceptuel est un graphe ayant deux types de nœuds :

- les concepts,
- les relations conceptuelles.

La phrase « Un homme casse une branche avec son pied » peut être représentée par le graphe conceptuel ci-dessous :

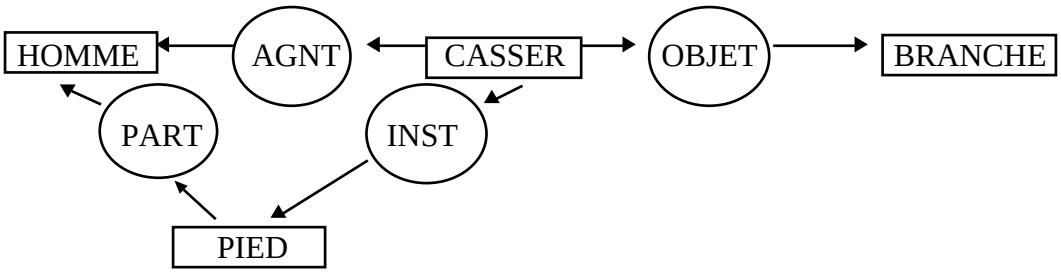


figure 2-22 : « Un homme casse une branche avec son pied »

Les concepts HOMME, CASSER, BRANCHE, PIED sont représentés par des rectangles dans lesquels figurent leur libellé. Les relations conceptuelles PART, AGNT, INST et OBJET sont représentées par des cercles.

Pour des raisons de commodité d'écriture, nous trouverons généralement des graphes écrits sous forme de texte linéaire. Ainsi, le graphe représenté par le diagramme ci-dessus

peut aussi être écrit de la façon suivante :

```
[CASSER] -
        (AGNT) -> [HOMME:*x],
        (OBJ) -> [BRANCHE],
        (INST) -> [PIED] ---> (PART) --> [HOMME:*x].
```

Les crochets sont utilisés pour marquer les concepts, tandis que les parenthèses délimitent les relations. Un concept tête est tout d'abord choisi. Il est suivi d'un tiret, pour indiquer que les relations conceptuelles qui lui sont reliées suivent de manière séquentielle. La virgule est utilisée pour terminer un niveau du graphe et le point termine le graphe. Le choix du concept tête, ici CASSER, dépend du point de vue où l'on se place.

Une autre forme linéaire peut être obtenue en changeant de concept tête. Les différentes formes linéaires obtenues sont équivalentes.

Le symbole *x est appelé variable et permet d'indiquer la coréférence entre l'homme qui casse la branche et le propriétaire du pied utilisé pour casser la branche.

L'orientation des flèches doit être guidée par l'association du sous-graphe [CONCEPT1] -> (REL) -> [CONCEPT2] à la phrase « La REL de CONCEPT1 est CONCEPT2 ».

Un graphe conceptuel possède les propriétés suivantes :

- il a un nombre fini de nœuds.
- il est connexe, car tous ses nœuds sont reliés. Si deux parties ne sont pas connectées, ces deux parties seront considérées comme deux graphes conceptuels.
- il est bipartite, dans le sens où il n'est composé que de deux types de nœuds.

Une relation conceptuelle est reliée à au moins un concept. Si cette relation a n arcs, elle est dite **n-adique** et ses arcs sont libellés de 1 à n. Le nombre de concepts (n) associés définit l'arité de la relation conceptuelle.

Un concept peut former à lui seul un graphe conceptuel, ce qui n'est pas le cas d'une relation conceptuelle.

1) Concept

Un concept est un objet ayant un **type** et éventuellement un **référent**.

Il s'écrit sous la forme [<type> : <référent>].

Exemple : [PERSONNE : 'Nicolas'], [MAISON : #14]

a) Type de concept

Le **type** indique la classe à laquelle appartient le concept.

Soit la fonction *type* de l'ensemble des concepts vers T, l'ensemble des noms de

concepts. Les concepts c1 et c2 sont dits de même type si *type*(c1) = *type*(c2).

Dans les graphes conceptuels, deux concepts ayant le même nom de type (le même nom entre crochets) sont deux concepts de même type.

La **dénotation** du nom de type t, notée δ**t**, est l'ensemble de toutes les entités qui sont des instances de tout concept de type t.

Sur les types de concepts, une relation d'ordre partiel "≤" définie permet de construire une hiérarchie de type.

Soient s,t et u des noms de type.

- Si s ≤ t, alors s est dit **sous-type** de t et t **surtype** de s (t ≥ s).
- Si s≤ t et s≠ t, s est dit **sous-type propre** de t, t est dit **surtype propre** de s.
- Si s ≤ t et ts ≤ u, alors s est appelé **sous-type commun** de t et u.
- Si s ≥ t et s ≥ u, alors s est appelé **surtype commun** de t et u.

Cette hiérarchie de type forme ce qui est appelé **un treillis de type** contenant

- un type universel noté ⊤,
- un type absurde ⊥

tels que pour tout nom de type t , ⊥ ≤ t ≤ ⊤

Dans ce treillis, toute paire (s,t) de noms de type possède un **surtype commun minimal**, noté **s** ∪ **t**. . Pour tout surtype u de s et t, u est aussi surtype de **s** ∪ **t**...

Toute paire (s,t) possède un **sous-type commun maximal**, noté **s** ∩ **t**, et tout sous-type u de s et t est aussi sous-type de s ∩ t.

Dans le treillis ci-dessous,

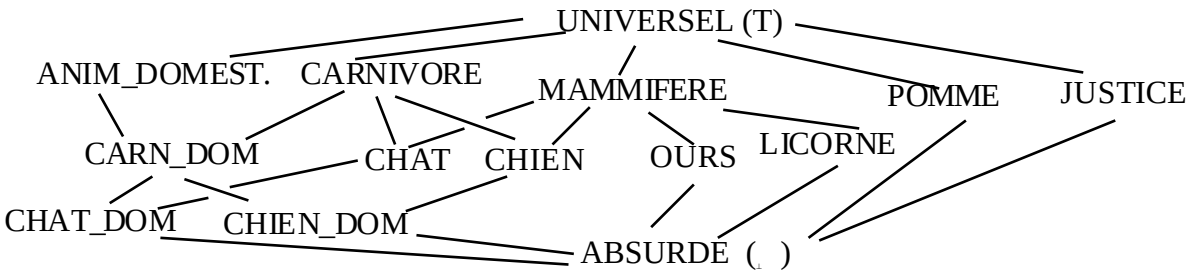


figure 2-23

CHAT, CHIEN, OURS, LICORNE, POMME, JUSTICE n'ont pas de sous-type commun. CHAT_DOM est le sous-type commun maximal de CHAT et ANIMAL_DOMESTIQUE. CARNIVORE_DOMESTIQUE est le surtype commun minimal de CHAT_DOM et CHIEN_DOM. MAMMIFERE est le surtype commun de LICORNE et CHAT.

b) Le référent

Des **marqueurs individuels** permettent de préciser l'individu du monde réel concerné. Soit l'ensemble des marqueurs individuels I = {#1, #2, #3, ...}.

Sowa définit de la manière suivante la fonction *referent* qui s'applique à un concept c :

- référent(c) ∈ I, le concept c est un **concept individuel**
- référent(c) = *, le concept c est un **concept générique**.

Le concept [CHAT:*] ou [CHAT] renvoie à un chat non spécifié.

Le concept [HOMME:*x] renvoie aussi à un homme non spécifié.

Le concept [CHAT:#9456] renvoie à un chat particulier auquel le système aurait affecté le numéro 9456.

On trouve aussi des concepts comme [PERSON:Judy].

Ce concept est issu de la contraction du graphe

[PERSON:#4567] -> (NAME) -> ["Judy"]

et renvoie à la personne appelé Judy.

NAME est le nom d'une relation conceptuelle qui associe **a** (sous concept du type ENTITY) à **b** (sous concept propre de WORD).

Le **référent** est dit **ensembliste** lorsqu'il renvoie à un ensemble d'entités réelles partageant le même type de concept .

Par exemple, le concept [PERSONNE:{Jean,Nicolas}] renvoie à l'ensemble de personnes composé de Jean et Nicolas.

Le symbole @ suivi d'un nombre sera utilisé pour indiquer le cardinal de l'ensemble des entités auxquels se réfère le concept ensembliste.

Ainsi, [PERSONNE:{Jean,Nicolas} @3] désigne 3 personnes dont Jean et Nicolas.

Le référent {*} est utilisé pour désigner plusieurs entités de même type.

Dans certains cas, le référent est lui même un graphe conceptuel. Le type de cette entité particulière sera soit PROPOSITION, soit SITUATION. Ces concepts sont appelés concepts du deuxième ordre. Ils forment un des thèmes principaux de recherches du groupe PRC-GDR qui s'est constitué autour de cette voie.

Un exemple d'utilisation d'un concept de second ordre est donné ci-dessous.

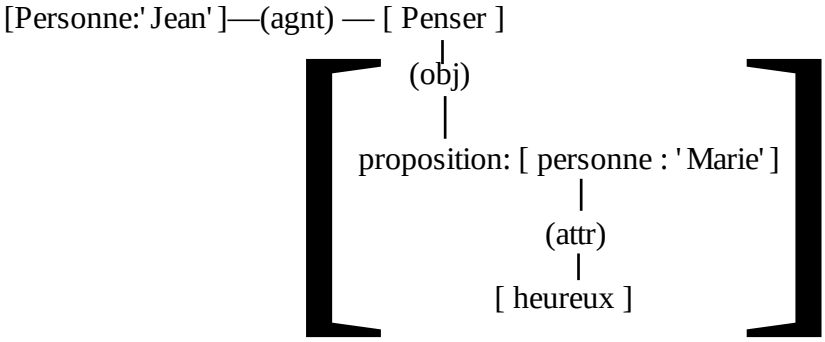


figure 2.24 : « Jean pense que Marie est heureuse »

c) La relation de conformité

Le marqueur individuel i est dit conforme au type t (on notera i :: t) si t::i est vrai.

La relation de conformité (“::”) est définie comme suit :

- ∀ c de l’ensemble des concepts, type(c) :: référent(c)
- Si s::i et s≤ t, alors t::i
- Si s::i et t::i, alors (s∩t)::i
- ∀ i de I, **T**::i, et non(i::⊥)
- ∀ t de l’ensemble des types, t ::*.

2) Relations conceptuelles

Une relation conceptuelle relie des concepts d’un graphe et permet de spécifier le rôle joué par chaque concept au niveau du graphe (ou sous-graphe).

a) Arité

Une relation conceptuelle est caractérisée par son arité. Elle indique le nombre de concepts que la relation lie. Elle est supérieure ou égale à 1.

La relation est **unaire** lorsque son arité est égale à 1. C’est le cas des relations NEG, qui exprime la négation d’un graphe conceptuel, PAST, qui permet de formuler la notion du passé, et PSBL(ou POSS), qui exprime la notion de possibilité.

La relation est dite **binaire** lorsque son arité est égale à 2. Sowa utilise notamment les relations AGNT et OBJ qui indiquent respectivement l’agent et l’objet d’une action.

La relation est dite **ternaire** lorsque son arité est égale à 3.

Les flèches sont orientées dans le sens concept-relation pour les arcs numérotés de 1 à n-1. Le $n^{ième}$ arc est orienté dans le sens relation-concept.

Par exemple, à la phrase « Entre deux briques, il y a un espace », nous obtiendrons le

modèle formalisé par le graphe conceptuel suivant :

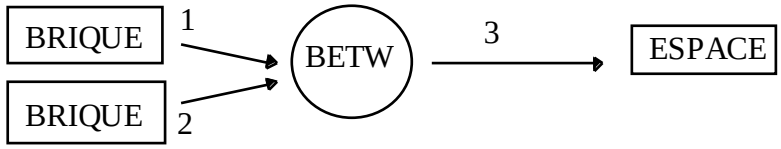


figure 2-25 : « Entre deux briques, il y a un espace. »

b) Hiérarchie de type de relation conceptuelle

La fonction type définie précédemment pour les concepts peut être étendue aux relations conceptuelles.

Deux relations conceptuelles r et s sont dites de même type si type(r) = type(s)

Le type d'un concept est toujours différent du type d'une relation.

Deux relations de même type auront alors le même nombre d'arcs.

Un ordre partiel existe aux niveaux des types de relations. Par contre, il n'y a pas de définition de surtype commun comme pour les concepts. Ainsi, la relation LOC possède des sous types comme DANS, SOUS, SUR.

4.1.2. Notion de graphe

Tous les graphes conceptuels se présentent sous forme d'ensemble de nœuds et de relations. Or certains ensembles, comme celle ci-dessous, n'ont pas de sens.

[DORMIR] -

(AGNT) -> [IDEE] -> (COULEUR) -> [VERT],

(MNR) -> [FURIEUX].

Cette combinaison pourrait provenir d'un analyseur conceptuel auquel on aurait fourni en entrée la phrase célèbre de Chomsky, « Des idées vertes dorment furieusement. ».

Les graphes, correspondant à une représentation réelle ou possible du monde réel, seront appelés par Sowa, **graphes canoniques**.

Si on applique sur des graphes canoniques des opérations comme la restriction, la copie, le joint et la simplification (que nous détaillerons dans les paragraphes suivants), les nouveaux graphes obtenus seront aussi canoniques.

Les graphes introduits par le cogniticien dans le système seront considérés comme canoniques.

Une grammaire de formation de ces graphes canoniques présente plusieurs avantages. Les programmes d'I.A. font un usage important d'opérations similaires. Une sorte de standardisation des modèles (*templates*) permettrait la dérivation de nouveaux graphes. Par ajout de ces contraintes, elle évite la dérivation de combinaisons sans sens. De plus, elle a un

rôle simplificateur au niveau des définitions et des preuves de la théorie.

4.1.3. Opérations sur les graphes

Il y a quatre opérations possibles sur les graphes :

la copie, la restriction, la jointure et la simplification.

Toutes ces opérations préservent la canonicité des graphes. L'application d'une de ces opérations sur un graphe canonique donne un graphe canonique.

La **copie** permet d'obtenir une copie conforme au graphe sur lequel s'applique l'opération.

La **restriction** permet de restreindre des concepts d'un graphe.

Deux types de restrictions sont possibles :

- Le remplacement du type d'un concept (ANIMAL) par un sous-type (CHAT) permet d'obtenir à partir du graphe [ANIMAL] --> (SUR) -> [TAPIS], le graphe dérivé canonique [CHAT] --> (SUR) -> [TAPIS].
- - L'individualisation d'un concept.
  Le concept [CHAT : *x] serait transformé en [CHAT:Chipie]. Dans ce cas, la relation de conformité doit être vérifiée. Ainsi, dans le cas où Chipie est un chat siamois, la restriction de [CHAT : Chipie] vers [SIAMOIS : Chipie] est acceptée, tandis que celle donnant [PERSAN : Chipie] n'est pas permise.
  On peut combiner les deux formes de restrictions.
  A partir de [ANIMAL], on obtiendrait [CHAT:Chipie].

La **jointure** s'applique sur deux graphes conceptuels G1 et G2 et le résultat R est issu de la fusion de tous les concepts communs identiques (même type et même référent).

Si G1 et G2 contiennent le concept [PERSONNE], alors le graphe R contiendra un unique concept [PERSONNE]. Par contre, [PERSONNE] et [PERSONNE:Nicolas] ne peuvent être fusionnés car ce ne sont pas des concepts identiques.

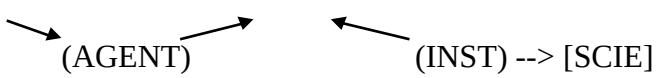
A partir des deux graphes :

[PERSONNE] - >(AGENT) -> [SCIER] -<- (OBJET) <-- [BRANCHE]

[PERSONNE] - >(AGENT) -> [SCIER] <-- (INST) <-- [SCIE]

on obtiendra par jointure le graphe

[PERSONNE] - >(AGENT) -> [SCIER] <-- (OBJET) <-- [BRANCHE]



La **simplification** permet d'éliminer des informations redondantes, en particulier, après une opération de jointure.

Soit R1 et R2, deux relations du graphe.

R1 et R2 sont redondantes si :

- type(R1) = type(R2) (leur arité A est donc la même)
- A tout i de 1 à A, l'arc i de R1 renvoie au même concept que l'arc i de R2.

Dans le graphe conceptuel précédent obtenu après jointure, la relation conceptuelle AGENT est redondante, la simplification supprime la redondance, le graphe devient :.

La combinaison des quatre opérations est possible.

Ainsi, l'opération appelée **joint maximal** consiste, à partir de deux graphes de départ, à effectuer des restrictions, puis une jointure, et une simplification.

Ces règles de formation ne sont pas des règles d'inférences. Elles ne peuvent indiquer la vérité ou la fausseté de la représentation dérivée.

4.1.4. Généralisation et spécialisation

1) Généralisation et spécialisation d'un graphe

Les règles de formations présentées ci-dessus sont dites **règles de spécialisation** car elles génèrent des informations supplémentaires provenant d'autres graphes qui contraignent les graphes initiaux.

Un graphe **u** est appelé **spécialisation d'un graphe v** (noté $u \leq v$) s'il a été dérivé de v à partir des opérations de copie, joint, restriction, simplification.

L'opération de généralisation est l'inverse de la spécialisation.

Le graphe **v** sera dit **généralisation de u** si $u \leq v$.

2) Hiérarchie de généralisation

La relation $\leq$ est une relation d'ordre partiel sur les graphes.

En effet, la relation est :

- réflexive : $u \leq u$
- transitive : $u \leq v$ et $v \leq w \Rightarrow u \leq w$
- antisymétrique : $u \leq v$ et $v \leq u \Rightarrow u = v$

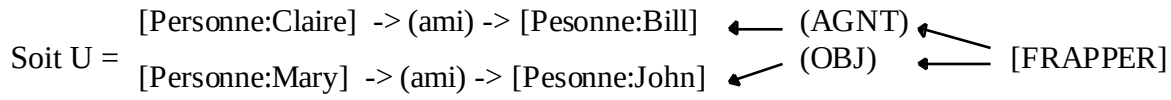
Ici, le graphe conceptuel réduit au seul concept [T] (Univers) est la généralisation de tous les autres graphes conceptuels.

3) Projection

Si u est une spécialisation de v, il doit exister un sous graphe u' de u qui représente le graphe v. Ce graphe u', appelé **projection dans u de v**, est créé par la fonction π de la manière suivante (on notera $u' = \pi v$) :

- tous les concepts c' de u' sont des restrictions des concepts c de v. (on notera $c' = \pi c$)
- les relations conceptuelles r' de u' sont identiques aux relations r de v (noté $r' = \pi r$)
- pour chaque arc i de r (dans v) liant un concept c (dans v), l'arc i de πr (dans u') doit être lui même relié à un concept c' tel que $c' = \pi c$.

Exemple :



le graphe correspondant à l'énoncé

« Bill, l'ami de Claire frappe John, l'ami de Mary. ».

Soit V, le graphe [Personne:*x] -> (ami) -> [Personne:*y]

alors deux projections U'1 et U'2 peuvent être obtenues.

U'1 : [Personne:Claire] -> (ami) -> [Personne:Bill]

U'2 : [Personne:Mary] -> (ami) -> [Personne:John]

L'exemple montre que la fonction de projection π n'est pas nécessairement unique.

4.1.5. Création de nouveaux types de concepts et relations

Le formalisme des graphes conceptuels permet la définition de nouveaux types de concepts ou de nouvelles relations conceptuelles.

Deux méthodes sont proposées par Sowa :

- la première, dérivée de la méthode dite du "*genus and differentia*" de Aristote, s'appuie sur des définitions formulées en termes de conditions nécessaires et suffisantes.

- la seconde, basée sur les schémas et prototypes, que l'on peut retrouver par exemple dans des systèmes de type KRL.

1) Définition de types à l'aide de conditions nécessaires et suffisantes

Une **abstraction n-adique** , noté $\lambda a_1, .., a_n u$, est formée d'un graphe canonique u,

appelé **corps de l'abstraction**, et d'une liste de concepts génériques a1, ..., an appelés **paramètres formels** appartenant à u. Cette liste permet de distinguer ces concepts génériques des autres concepts utilisés dans u.

Une définition de **type de concept** t sera définie par une **abstraction monadique** λa u, noté aussi "**type t(a) is u**". Le corps u est la *differentia* de t, et type(a) est le *genus* de t.

Ainsi, le type CHANTEUR serait défini par l'abstraction monadique :
type CHANTEUR(*x) is
[ETRE_VIVANT:*x] <- (AGNT) <- [CHANTER]

Pour éviter d'avoir à gérer une profusion de types, il est possible d'écrire
[λx [ETRE_VIVANT:*x] <- (AGNT) <- [CHANTER] :Jean]. au lieu de
[CHANTEUR:Jean].
Cette forme de notation rappelle l'écriture des fonctions dans les langages de type LISP.

Une définition de **type de relation conceptuelle** d'arité n est une abstraction n-adique. Elle est notée "**relation t(A1, ..., an) is u**".
Ainsi, la définition de la relation conceptuelle ARROSER pourrait s'écrire :
TYPE ARROSER(x,y) is
[ENTITE:*x] <- (AGNT) <- [VERSER] -> (OBJET) -> [LIQUIDE]
-> (DEST) -> [ENTITE:*y]

2) Définition de types à l'aide de schémas et prototypes

Une définition permet d'introduire des connaissances spécifiques au domaine relatives aux organisations plausibles. Un schéma représente en effet tout ce qui est plausible tandis qu'un graphe canonique représente tout ce qui est concevable.

Chez Sowa, la notion de graphe canonique évite la construction de graphes pour « Des idées vertes dorment furieusement » mais pas d'énoncés comme « des vaches de couleur mauve dorment. ». C'est la notion de schéma qui ajoutera des contraintes sémantiques à prendre en compte. Il définit une utilisation possible des concepts qu'il intègre.

Un type peut avoir plusieurs schémas associés, appelé **groupe de schémas** (*shematic cluster*). Un groupe de schémas pour un type t est un ensemble d'abstractions monadiques $\{ \lambda a_1 u_1, .., \lambda a_n u_n \}$ où chaque paramètre formel ai est de type t et chaque abstraction monadique $\lambda a_i u_i$ est appelée **schéma pour le type t.**

Exemple : SCHEMA FOR Bus(x) IS

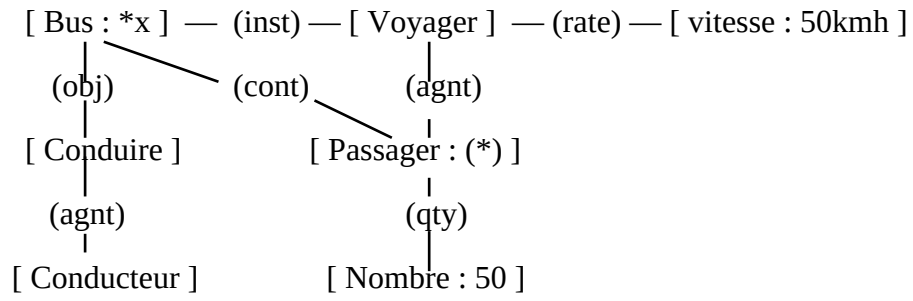


figure 2-26 : Un des schémas associé au type BUS

Pour déterminer la plausibilité d'un graphe donné, le système utilise l'opération appelée joint schématique. Elle se définit de la manière suivante:

Soit V un graphe canonique ayant le concept b et soit λa un schéma pour Type(b). Un **joint schématique** de λa et V est un joint maximal de U et V en liant le concept b et le paramètre formel a.

Si le joint obtenu n'est pas maximal, une présence de conflits est alors déduite.

Le graphe canonique à l'origine de ces conflits peut être considéré comme non plausible.

Un schéma permet de montrer les différents "usages" d'un concept, mais il ne décrit pas une instance typique d'un concept. C'est la notion de prototype qui nous permettra de décrire un individu typique.

Un prototype p pour un type t est une abstraction monadique vérifiant les conditions suivantes :

- le paramètre formel a est de type t
- le prototype p est le résultat d'un joint schématique entre un ou plusieurs schémas de l'ensemble de schémas pour t, avec une ou plusieurs restrictions des concepts génériques de P en des concepts individuels.

Exemple :

```
PROTOTYPE FOR Homme(x) IS
        [ Homme : *x ]
                (chrc) — [ Sexe : masculin ]
                (chrc) — [ Poids : 70kg ]
                (chrc) — [ Taille : 1m70 ]
                (part) — [ Main : @ 2 ]
                (stat) — [ Vivre ] —
                                        (dur) — [ Periode : 65ans ].
```

4.1.6. Opérations sur les nouveaux types

Le formalisme de Sowa propose des opérations utilisant ces définitions de nouveaux types. Ce sont les opérations de contraction et d'expansion.

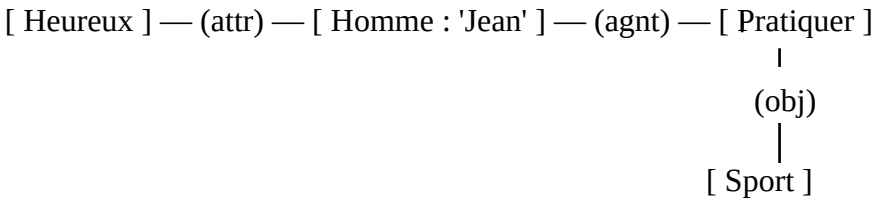
1) Opération de contraction

Elle consiste à remplacer dans un graphe U donné, le sous graphe V de U, tel que V est le corps d'une définition de type t=λaV par le concept qui correspond au paramètre formel *a*.

Soit TYPE Sportif(x) IS

[Personne : *x] — (agnt) — [Pratiquer] — (obj) — [Sport]

et soit le graphe suivant:



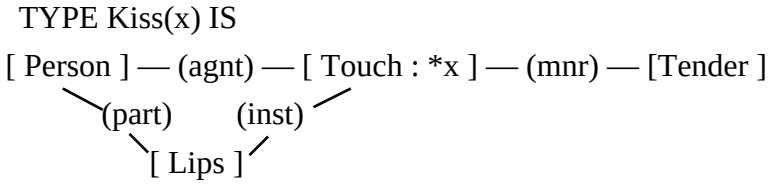
L'opération de contraction donnera le graphe :

[Sportif : 'Jean'] — (attr) — [Heureux]

2) Opération d'expansion

C'est l'opération inverse qui permet de remplacer un concept d'un graphe conceptuel par sa définition. Si "TYPE(a) IS U" est une définition de type, l'opération d'expansion de type sur un graphe V consiste à faire le joint de V et U sur un concept dont le type est t.

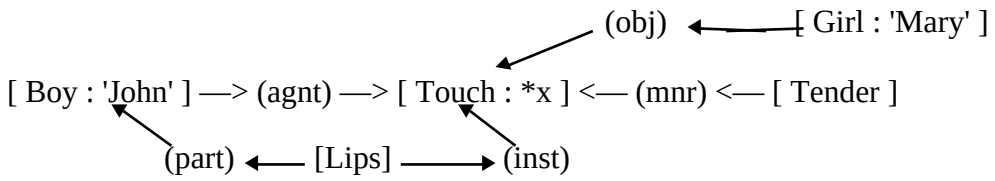
Soit la définition de type KISS:



et le graphe suivant :

[Boy : 'John'] — (agnt) — [Kiss] — (obj) — [Girl : 'Mary']

Le résultat de l'expansion sera :



3) Opération de contraction relationnelle

C'est l'analogue de l'opération de contraction. Ici, elle remplace le sous-graphe par une relation.

Soit le graphe :

[Garçon : 'Jean'] — (agnt) — [Donner] — (obj) — [Argent]
(rcv)
[Magasin] — (poss) — [Personne : 'Marie']

Avec la définition de la relation *Payer* définie de la manière suivante :

TYPE Payer(x) IS
[Person] — (agnt) — [Donner : *x] — (obj) — [Argent]
(rcv)
[Personne:*Y]

on obtient par contraction le graphe :

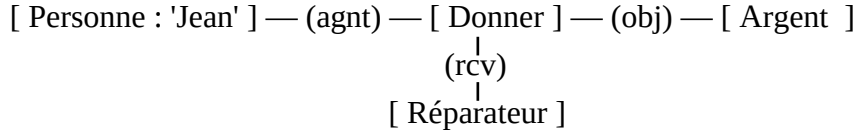
[Garçon] — (payer) — [Personne : 'Marie']<— (poss)— [Magasin]

4) Opération d'expansion relationnelle

C'est l'analogue de l'opération d'expansion. Ici, elle remplace la relation par le graphe équivalent.

[Personne] — (payer) — [Réparateur]

Avec la définition de la relation Payer, on aura par expansion le graphe :



4.1.7. Déductions, inférences

Une opération comme la jointure n'est pas une déduction logique. Comme le souligne Jean Fargues [Fargues, 1989], le joint "ne fait que propager certaines contraintes sélectives sur les graphes canoniques". Ces opérations sont des règles de formation plutôt que des règles d'inférence.

« *The canonical formation rules are the opposite of rules of inference. Whereas a rule of inference derives true graphs from true graphs, the canonical formation rules derive false graphs from false graphs. For this reason, they may be called refutation rules.* » [Sowa, 1984]

Sowa propose pour la déduction et la résolution sur les graphes conceptuels un ensemble de moyens. Il définit tout d'abord une fonction Φ qui associe à un graphe conceptuel une formule logique.

1) La fonction Φ

Sowa définit un opérateur Φ, qui associe à un graphe conceptuel u, une formule Φu en logique du premier ordre.

La formule Φu est construite de la manière suivante :

- A chaque concept c_i, est associé un prédicat unaire de nom type(c_i) avec un unique argument. Ce dernier, appelé identifieur de c_i, est une variable x_i si le concept c_i est générique, ou une constante égale à referent(c_i) si c_i est un concept individuel.
- A chaque relation conceptuelle n-aire r de u, est associé un prédicat n-aire de nom r et avec pour arguments les identifieurs des concepts reliés par r dans u
- Au graphe u, est associé la formule $\exists x_1, .., \exists x_n$ F
  où F est la conjonction des prédicats associés aux concepts ET aux relations de u et où les x_i sont les variables associées aux concepts génériques de u.
  Par exemple, la formule Φu associée au graphe
  [Personne : 'Jean'] — (agnt) — [Donner] — (obj) — [Argent]
  sera alors :

$$\exists x \exists y \, (\text{PERSONNE}(\text{Jean}) \wedge \text{AGNT}(x,\text{Jean}) \wedge \text{DONNER}(x) \wedge \text{OBJ}(x,y) \wedge \text{ARGENT}(y))$$

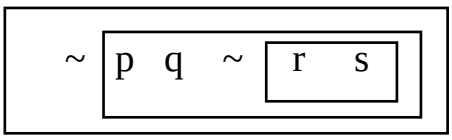
2) Déductions

a) Notation de Pierce

Pour exprimer des formules de logique sur les graphes conceptuels, Sowa utilise la notation de Pierce. Cette dernière se base sur la notion de contexte.
Un contexte est :
- soit un graphe,
- soit une conjonction de contexte,
- soit la négation d'un contexte.

Les contextes sont schématisés par des boîtes.

La formule $(p \wedge q) \supset (r \wedge s)$
qui est équivalente à $\sim(p \wedge q \wedge \sim(r \wedge s)$
est représentée dans la notation de Pierce
par une négation de contexte (voir figure ci-contre)



b) Formalisme "à la Prolog"

Pour Jean Fargues, la notation de Pierce est peu pratique pour plusieurs raisons :

- les opérations sur les contextes décrites par Sowa ne sont pas des méthodes applicables dans le cadre d'une implémentation,
- la notation se montre peu pratique pour décrire des déductions sous forme de règles ou de clauses

Il préfère à cette notation un formalisme de type Prolog (voir Annexe 3) qui s'inscrit dans l'optique résolution de problème et semble plus adéquat pour exprimer des requêtes ou des questions posées au système.
La règle « Une personne est membre du pays d'Oz si cette personne est née à Oz » sera ainsi représentée par
[CITIZEN : *x] <- (MEMB) <- [COUNTRY: 'Oz']
<= [PERSON : *X] <- (AGNT) <- [BORN] -> (LOC) -> [COUNTY : 'Oz']

4.2. Graphes conceptuels et problématique SYSQUEST

4.2.1. Représentations

1) Localisation spatiale d'objet "concret"

a) Première suggestion

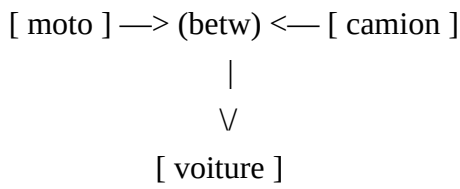
Les divers exemples de représentation des différentes relations spatiales que l'on rencontre dans les descriptions de systèmes à base de graphes conceptuels [Sowa, 1984] montrent des graphes dans lesquels les prépositions spatiales telles que "dans", "sur" interviennent comme relations conceptuelles.

Exemple 1 :

[Personne : 'nicolas'] ---> (dans) ---> [chambre :#ch12]
[Personne : 'nicolas'] ---> (près) ---> [chat : chat#2]

La relation spatiale "entre" faisant intervenir trois objets ne pose à priori pas de problèmes. Ici, l'arité de cette relation est 3.

Exemple 2 :



Le problème soulevé par ce formalisme de représentation réside dans l'utilisation de cette information dans les inférences.

Dans l'exemple 1, ces deux graphes sont traduits par la fonction Φ dans les deux formules suivantes :

$$\text{personne(nicolas)} \wedge \text{dans(nicolas,ch12)} \wedge \text{chambre(ch12)}$$
$$\text{personne(nicolas)} \wedge \text{près(nicolas,ch12)} \wedge \text{chat(ch2)}$$

Si le système a besoin de rechercher une localisation spatiale où intervient Nicolas en tant que cible, il lui faudra rechercher dans tous les graphes les formes suivantes :

RS1(nicolas,X) et RS2(nicolas,X,Y)
RS1 = {dans, sur, sous, près-de, ...}
RS2 = {entre, ...}

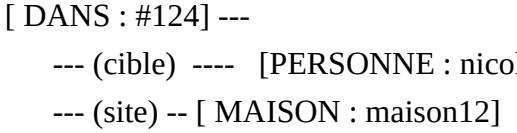
Si les graphes sont traduits dans une logique des prédicats d'ordre 1, la recherche de cette information nécessite l'énumération de toutes les relations spatiales existantes.

Une logique d'ordre 2 rendrait plus aisée ce type de recherche puisqu'il faudrait à première vue utiliser une forme comme RS1(nicolas, X) sans nécessiter d'énumérer les relations spatiales utilisées. Mais un inconvénient subsiste. L'ensemble de prédicats où Nicolas intervient comme premier argument ne se limite pas à des prédicats représentant une information spatiale. En effet, Nicolas peut jouer le rôle d'acteur dans une action de "payer", par exemple.

Les problèmes sont identiques si l'on recherche une localisation spatiale où intervient 'Nicolas' en tant que site.

b) Deuxième suggestion

Pour éviter les problèmes dus à l'utilisation d'une relation conceptuelle pour représenter la relation spatiale, on pourra représenter la localisation spatiale par une instance d'un sous-concept de localisation spatiale. Par exemple, la phrase « Nicolas est dans la maison. » est représentée par le graphe :



Dans cette solution, une relation spatiale est traduite sous forme de concept. Une localisation spatiale fait alors intervenir un nouveau référent d'un concept LOC.

La traduction logique de cette nouvelle représentation correspondrait à :

dans(#124) et
personne(nicolas) et
maison(maison12) et
cible(nicolas,#124) et
site(maison12,#124).

Cette solution a l'avantage de trouver plus aisément une localisation spatiale où Nicolas joue le rôle de cible. Malheureusement, le fait que la relation spatiale soit mise dans le champ "nom du prédicat" rend difficile l'utilisation de cette localisation spatiale. Pour y remédier, nous proposons une autre solution qui consiste à indiquer le nom de la relation spatiale dans le référent du concept RS. Cette instance de concept est liée par la relation conceptuelle "rel-spatiale" à l'instance de concept de LS (localisation spatiale).

Ainsi, la phrase « Nicolas est dans la maison. » est représentée par

[LS : #l125] -
-- (cible) -- [HOMME : nicolas]
- (rel-spatiale) -- [RS : dans]
- (site) -- [MAISON : maison12]

Traduite en logique des prédicats, cela donne :

LocalisationSpatiale(l125) et
cible(nicolas, l125) et
rel_spatiale(dans, l125),
site(maison12, l125).

Pour savoir où se trouve nicolas1 en tant que cible, nous utiliserons les trois prédicats :

ls(X), cible(nicolas, X), rs(LARELATION, X).

Pour la recherche de Nicolas en tant que site, un problème subsiste. En effet, il faut impérativement utiliser des prédicats différents suivant qu'il s'agit du premier site, ou du second site dans une relation spatiale à 3 arguments. Par exemple, nous aurions pour « la voiture se trouve entre la moto et le camion. », le graphe conceptuel traduit en logique des prédicats donne :

LocalisationSpatiale(l126) et
cible(voiture1, l126) et
rel_spatiale(entre, l126),
site1(moto2, l126).
site2(camion3, l126).

Si l'on recherche une localisation où Nicolas intervient en tant que site, il faudra utiliser successivement les prédicats site, site1, site2, siten. Une solution intermédiaire, plus lourde à gérer, mais permettant ce type de recherches, peut être implantée. Il faudrait alors que le prédicat site ait un argument supplémentaire. Cet argument permet de savoir s'il est le premier, second, ou troisième site de la relation spatiale. On obtient alors une structure similaire à la suivante :

LocalisationSpatiale(l126) et
cible(voiture1, l126) et
rel_spatiale(entre, l126),
site(moto2, l126, 1).
site(camion3, l126, 2).

2)Localisation spatiale d'une action

Chacune des suggestions données ci-dessus peut être appliquée pour représenter ce type d'information. Nous retrouvons les mêmes problèmes que ceux présentés ci-dessus dans la recherche de la localisation spatiale d'une situation.

3) Représentation de localisations quantifiées

a) 1ère suggestion

Pour indiquer le fait qu'une chambre de maison se trouve dans la maison qui la contient, il faut une autre représentation que celle donnée ci-dessous :

[chambre_de_maison] --- (dans) --- [maison]

Pour représenter cette information quantifiée, le formalisme de Sowa nous permet l'adoption de deux solutions :

- soit utiliser des abstractions
- soit utiliser une règle logique.

Cas 1.

On utilise chez Sowa ce que l'on appelle une définition de type de relation conceptuelle (abstraction).

Toute chambre composante d'une maison est dans cette maison.

[(λx) [CHAMBRE :*x] -- (partie) -- [MAISON:*y] : ∀] --- (dans) -- [MAISON:*y] .

(La partie soulignée correspond à toute chambre composante d'une maison)

Cas 2.

On définira la clause règle suivante :

```
clause(règle)
  [chambre:*x] <- (dans) -- [maison:*y]
    <=  [chambre :*x] -- (partie) -- [maison : *y]
```

Dans les deux cas, le système garde le défaut provenant de l'utilisation d'une relation conceptuelle pour représenter le type de la relation spatiale.

b) 2ème suggestion.

On retrouve les deux cas présentés ci-dessus :

Cas 1

En tenant compte du fait que l'on crée à chaque localisation des instances du concept LS, nous obtiendrions une représentation similaire à :

[LS : ls1] -
-- (cible) -- [(λx) [CHAMBRE :*x] -- (partie) -- [MAISON:*y] : ∀]
- (rel-spatiale) -- [RS : dans]
- (site) -- [MAISON : *y]

(La partie soulignée correspond à toute chambre composante d'une maison)

Le problème ici réside dans le fait que nous utilisons une instance particulière (ls1) d'un concept pour décrire toutes les localisations possibles indiquant qu'une chambre fait partie d'une maison. Associer à une instance d'un concept un ensemble de situations différentes peut apparaître problématique en termes d'homogénéité.

Cas 2

On définira la clause règle suivante :

clause(règle)
[LS: ls1] --
- (rel-spatiale) -- [RS : dans]
- (site) -- [MAISON : *y]
- (cible) [chambre :*x]
<= [chambre :*x] -- (partie) -- [maison : *y]

4) Représentation de la localisation et temporalité

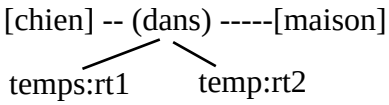
La relation conceptuelle (*past*) pourrait être utilisée pour exprimer la notion du passé. Son utilisation nécessite de fréquentes mises à jour de la base de connaissances à chaque introduction d'informations postérieures aux autres. Il est plus intéressant d'utiliser des entités cognitives de type temps que l'on intègre dans les graphes conceptuels afin de définir l'intervalle temporel durant lequel la localisation spatiale est valide.

a) 1ère suggestion

Prenons le schéma [chien] -- (dans) -----[maison]

Si l'on veut rajouter le temps, la première hypothèse consiste à utiliser l'hypothèse d'une relation "dans" comme relation quaternaire, possédant 4 arguments :

- Les deux premiers sont les entités jouant le rôle cible et site.
- Les deux derniers permettraient de localiser cette situation dans le temps.



La structure temporelle de la localisation spatiale est alors portée par la relation de localisation.

Cette première suggestion pose le problème de la généralisation de cette association de deux entités temporelles à d'autres relations.

Si l'on reprend l'exemple

on peut se demander duquel de la relation "agnt" ou "mnr" peut porter les deux entités temporelles.

Est-ce "agnt" ou "mnr" qui aura 4 "paramètres" ?

b) 2ème suggestion

Elle permet d'associer à toute instance de concept de localisation spatiale, deux entités par l'intermédiaire de deux relations conceptuelles : repère_temporel_début et repère_temporel_fin.

Nous obtiendrons le graphe suivant :

```
[ LS : #l125] -
    -- (cible) --    [HOMME : nicolas ]
    - (rel-spatiale) -- [RS : dans]
    - (site) -- [ MAISON : maison12]
      - (rt-debut) -- [RT : rt1]
       - (rt-fin) -- [RT : rt2]
```

On rencontre ici le même problème de généralisation de cette association aux autres types de représentations.

5) Représentation de la négation dans la localisation

La relation conceptuelle (neg) est utilisée pour la négation d'un graphe conceptuel.

On aura alors :

(neg) ---- [SITUATION : [.....]]

Nous pouvons proposer deux suggestions pour représenter « Nicolas n'est pas dans la maison. ».

a) Première suggestion

(neg) -- [SITUATION : [[Personne: 'nicolas'] --> (dans) --> [chambre:#ch12]]]

b) Deuxième suggestion

```
(neg) -- [SITUATION :
                [ LS : #l125] -
                    - (cible) --    [HOMME : nicolas ]
                    - (rel-spatiale) -- [RS : dans]
                    - (site) -- [ MAISON : maison12]
                ]
        ]
```

4.2.2. Inférences autour de ces représentations

Le formalisme à la Prolog proposé par Fargues permet une facilité d'écriture de règles

qui permettent la déduction de nouvelles localisations spatiales. Les antécédents et les conclusions des règles sont des graphes conceptuels.

Comme le mécanisme considère que le graphe [A] <- (R) <- [B] -> (S) -> [C] est équivalent à la conjonction des sous-graphes [A] <- (R) <- [B:*x] et [B:*x] -> (S) -> [C], le nombre de graphes "atteints" par une règle est important.

Malgré cette expressivité, l'écriture des règles est complexe car il faut donner les formes multiples de graphes des représentations spatiales dues aux nombres différents de sites.

4.3. Conclusion

Les représentations d'informations spatiales à partir de graphes conceptuels posent des difficultés. Les problèmes posés proviennent de la structure même d'une relation. Une relation a en effet une arité A (de 1 à A concepts liés) bien définie. C'est elle qui permet de relier plusieurs concepts entre eux. Chacun des concepts liés joue un rôle dans la relation et ce rôle n'est pas explicité. Lorsque l'on a besoin d'effectuer des recherches par rapport aux rôles pris par les concepts, on constate une augmentation de la complexité des structures de graphes manipulées. Le domaine d'exemples a montré aussi la difficulté de la recherche simultanée d'une relation à 2, 3 ou N sites à cause des multiples structures prises par chaque type de relation spatiale.

Chapitre 3 :
Proposition d'un système de représentation des connaissances : les DCAS

1. Introduction

Dans le chapitre précédent, nous avons vu que chacun des systèmes examinés présente à la fois des aspects intéressants par rapport à la problématique SYSQUEST, mais aussi des points de difficulté.

Deux choix étaient alors possibles :

- élaborer une forme de coopération entre les différents systèmes de représentation existants.
- élaborer un nouveau formalisme qui serait en mesure de respecter nos contraintes.

Nous avons retenu cette deuxième solution. Nous décrirons tout d'abord les difficultés rencontrées dans l'élaboration d'une forme de coopération entre les systèmes de représentations existants, puis nous proposerons un formalisme répondant à nos besoins.

2. La coopération de différents systèmes de représentations

Cette idée de coopération est l'objet de préoccupation des chercheurs en IA.

Elle est issue du constat suivant. Certains mécanismes d'inférences sont plus adéquats à certaines formes de représentations que pour d'autres.

Une multiple représentation d'une même connaissance donnée permet alors l'accès à plusieurs formes de raisonnement. Cette connaissance sera manipulée par plusieurs mécanismes inférentiels.

Cette idée conduit à la notion de systèmes hybrides combinant plusieurs systèmes de raisonnement en un tout complexe.

Les représentants les plus connus de cette famille de systèmes sont KRYPTON [Brachman, Fikes & Levesque, 1983] et KL_TWO [Vilain, 1985].

De ces approches hybrides, nous pouvons mentionner les avantages suivants :

- gain d'efficacité,
- inférences spécialisées,
- élimination de recherches inutiles.

Le gain d'efficacité est généralement l'avantage le plus recherché. D'ailleurs, il est même parfois à l'origine du choix des systèmes hybrides.

Vilain [Vilain, 1985] décrit des systèmes partant d'un démonstrateur de théorème en logique du premier ordre auquel viennent s'ajouter des composantes inférentielles. Ces dernières sont spécialisées pour un type particulier de raisonnement et se montrent, sur ce point précis, plus efficaces que le démonstrateur de théorèmes. Ces ajouts permettent de contourner la procédure normale de preuve et évite des recherches combinatoires coûteuses.

Cependant, des difficultés apparaissent lors de l'élaboration de tels systèmes : en particulier, bien qu'il y ait accroissement d'efficacité, l'expressivité ne change pas. ([Vilain, 1985])

Lorsque les composantes additionnelles sont restreintes (comme dans les systèmes décrits par Vilain), le système résultant devient efficace. Si les composantes additionnelles ne restreignent en aucune manière l'espace de recherche dans les autres parties du système, le système résultant devient inefficace puisque chaque sous-système travaille de son côté pour son seul compte.

Il est alors nécessaire d'ajouter des protocoles de communication entre les systèmes leur permettant ainsi de s'informer de l'intérêt de leur travail à un moment donné. Soient deux systèmes S1 et S2, chacun essayant de trouver une solution à un problème particulier. Supposons que chaque système ait ses propres structures de représentation du problème. Si S1 trouve la solution du problème, il vaudrait mieux qu'il informe S2 de sa réussite pour éviter à ce dernier de rester dans une recherche de solution qui risquerait de lui être très coûteuse en temps.

D'autres difficultés surgissent dans la mise en place de tels systèmes. Il n'est pas trivial de définir les modalités de collaboration.

Il faut tout d'abord décider quel type de représentation est plus adéquat pour tel type de connaissance et formuler des critères de choix d'un formalisme par rapport à un autre.

Dans le cas où une multiple représentation est choisie en raison de types de raisonnements recherchés, quelles seront les conséquences d'une déduction dans un système de représentation donné ? Faudra-t-il traduire cette connaissance dans les autres formalismes ? Si oui, un procédé de traduction doit être défini.

Le problème de l'utilisation d'une structure commune d'échange d'informations apparaît alors.

Les difficultés à faire coopérer les différents systèmes de représentation sont loin d'être négligeables. La problématique posée nous amène au second choix envisagé qui consiste à proposer un nouveau formalisme intégrant des fonctionnalités permettant des évolutions futures.

3. Description du formalisme des DCAS

Nous commencerons ce chapitre en introduisant les idées à l'origine du formalisme des DCAS (**D**escription **C**omposite **à S**ignification). Cette introduction sera suivie par la définition d'une DCAS et par la description de la syntaxe de construction de DCAS.

Ensuite, nous présenterons les différentes entités cognitives mises en jeu. Ces entités cognitives sont regroupées par type. Dans une DCAS, un nombre fini d'entités cognitives est utilisé. Nous décrirons les moyens d'expliciter pour chacune de ces entités cognitives utilisées leur participation au sens associé à cette DCAS.

Puis, nous parcourrons les diverses utilisations de ces descriptions pour représenter certaines connaissances rencontrées dans le monde du discours.

Enfin, nous présenterons l'étude du mécanisme de production de nouvelles informations dans une base de DCAS.

3.1. Introduction

La problématique SYSQUEST nous a permis de mettre en évidence des difficultés rencontrées par des systèmes de représentation, en particulier :

1) L'utilisation de systèmes à base de primitives sémantiques est rendue difficile lorsque les structures déjà proposées sont inadéquates pour les traitements recherchés. En effet, avec la dépendance conceptuelle, nous avons pu voir les difficultés à intégrer de nouvelles primitives avec leur prise en compte dans des raisonnements. De plus, le codage procédural des règles d'inférence pose des problèmes de maintenance lorsqu'il s'agit d'intégrer les spécificités des nouvelles connaissances. Par exemple, l'intégration des connaissances spatiales à deux sites dans la dépendance conceptuelle n'a pas été aisée.

Un système utilisant des règles explicites est intéressant dans la mesure où ces dernières restent modifiables. Bien entendu, certaines règles peuvent être codées en interne lorsqu'il y a nécessité d'accélérer le raisonnement. Dans un système à base de DCAS, le raisonnement sera mené par des règles qui seront explicites, dans la plupart des cas. Dans des situations

particulières, et pour des raisons de rapidité, certaines règles pourront être rendues internes au système et optimisées.

2) Dans la plupart des systèmes de représentations, la connaissance est décrite en donnant un rôle prépondérant à une donnée particulière de l'information et cette donnée rigidifie les éléments qui la composent.

Avec le formalisme de Brachman, un rôle important est donné au concept de localisation spatiale dans la problématique SYSQUEST. Cela signifie que les accès se font par ce concept. Pour rechercher, par exemple, l'endroit où se trouve Nicolas, le système ne se contentera pas uniquement de la seule donnée Nicolas (instance d'être humain) ; il faudra préciser au système, le rôle joué par Nicolas avec l'instance du concept localisation. Ainsi, rechercher une instance du concept localisation à laquelle participe Nicolas nécessite au système un parcours de toutes les instances et l'expression de tous les cas possibles de rôle joué par cette instance.

Avec les graphes conceptuels, nous avons décrit les deux points de vue qui peuvent être mis en avant pour la localisation : une localisation spatiale en tant qu'instance de concept ou en tant que graphe avec pour relation de liaison, la relation spatiale utilisée. Ces deux choix ont été analysés et les défauts sont principalement dus à l'orientation des données dans le graphe.

Retrouver des informations quelconques à propos d'une entité donnée (ici, Nicolas) est difficile. Exploiter ces données en évitant de décrire toutes les permutations possibles n'est pas chose aisée. Plus généralement, toute structure de connaissance ayant implicitement ordonné les éléments qui la composent est difficile à gérer.

Évidemment, s'il n'y a pas d'ordre des éléments composants, il faudra trouver un moyen d'exprimer leur participation à l'ensemble de la structure formée. Ces éléments ne se distinguent pas uniquement au niveau de leur catégorie d'appartenance, mais aussi par le rôle joué dans la sémantique de l'ensemble. Cette idée peut être rapprochée de la théorie des cas (Fillmore) qui bâtit autour d'une phrase comme « le chat mange la souris. » un squelette dont l'élément central est le verbe, et où le chat et la souris possèdent des rôles sémantiques différents. Dans notre formalisme, par rapport à l'idée globale précédente, aucun élément n'est considéré comme élément fondamental. Chacun contribue à sa manière à créer la signification globale de la structure et chaque contribution est explicitée à l'intérieur de la structure.

3) L'accès à l'information est rendue difficile par l'impossibilité de décrire de manière incomplète la connaissance que l'on recherche.

Cette difficulté est d'ailleurs fortement liée à la difficulté précédente.

Même en utilisant des variables, pour la majorité des systèmes de représentation, il faudra décrire toutes les variations possibles de la structure recherchée. Si l'on prend la

logique des prédicats, la connaissance étant structurée grâce à des termes, il faut décrire à travers des inconnues les éléments (dont le nombre est fixe) du terme qui manquent et connaître les rôles joués par chaque élément du terme.

Si la possibilité de description incomplète de l'information recherchée était fournie, cela rendrait plus souple la recherche de données introduites par un utilisateur de la base de connaissances. De plus, les règles d'inférence que l'on sera amené à produire peuvent bénéficier de cette puissance expressive puisqu'elles sont, elles-aussi, utilisatrices des procédures de recherche.

Avec les DCAS, nous proposons des structures qui permettent un accès aisé à l'information par l'utilisation de descriptions incomplètes et nous représentons les règles en conformité avec les structures proposées.

3.2. Définition d'une DCAS

3.2.1. Unité de base

L'unité de base du raisonnement est la DCAS, la **D**escription **C**omposite **A** **S**ignification.

Le terme composite est due à la disparité des éléments qui la composent.

Chaque description composite est un ensemble non-ordonné de quadruplets. Le cardinal de cet ensemble est fini et généralement supérieur à 1.

Leur forme est la suivante :

DCAS = { <quadruplet-1> , ... , <quadruplet-n> }

N'importe quel ensemble de quadruplets n'est pas obligatoirement une DCAS.

Le sens d'une DCAS est formé par les différentes entités cognitives (**EC**) manipulées et les différentes contributions de ces entités à la création du sens.

Dans un processus d'inférences, il n'est pas possible de raisonner uniquement à partir d'un fragment d'une DCAS car ce fragment n'a à priori aucun sens associé. C'est toute la DCAS qui doit avoir un sens.

Le sens est indépendant d'un ordre que l'on pourrait assigner à ces quadruplets. Par exemple, si l'ensemble de quadruplets {(e1, e2, e3, e4), (f1, f2, f3, f4)} a un sens, alors l'ensemble {(f1, f2, f3, f4), (e1, e2, e3, e4)} a le même sens.

Exemple :

A la phrase « Le chat est dans la maison », correspond une DCAS représentant le sens de cette phrase. Cette DCAS fait intervenir un ensemble d'entités cognitives dans lequel figurent trois éléments associés respectivement au chat, à la maison, et à la relation spatiale "dans".

3.2.2. Structure des quadruplets d'une DCAS

Tout quadruplet d'un DCAS contient des informations relatives à une seule entité cognitive et à la contribution de cette entité cognitive au sens de la DCAS.

Un quadruplet n'a de sens que par rapport à la DCAS qui la contient.

Chacun des quadruplets se présente sous la forme suivante :
(<type-entité>, <rôle-entité>, <numéro-rôle>, <référent-entité>).

1) Le type et le référent de l'entité

Les deux informations <type-entité> et <référent-entité> sont relatives à l'entité cognitive manipulée tandis que <rôle-entité> et <numéro> portent sur une[22] contribution de cette entité au sens de la DCAS incluante.

Lorsque nous avons besoin de nous référer à une entité parmi d'autres, il nous faut donner les deux informations suivantes: <type-entité> et <référent-entité>.

<type-entité> désigne la catégorie conceptuelle à laquelle l'entité appartient.

A titre d'exemple, nous évoquerons brièvement quelques types d'entités utilisés : le type-entité concept , le type-entité relation et le type-entité instance.

La description des catégories conceptuelles que nous trouverons dans les paragraphes suivants sera limitée à celle des catégories qui nous semblent pertinentes par rapport à notre problématique.

<référent-entité> permet de distinguer des entités d'une même catégorie conceptuelle.

A cet emplacement, le système peut trouver une description. Dans ce cas, ce référent sera dit de nature complexe.

---

[22] Nous insistons bien sur le fait qu'il s'agit d'une contribution de cette entité au sens. Si cette dernière apportait un nombre N de contributions au sens d'une DCAS, il y aurait N quadruplets contenant des informations similaires au niveau des <type-entité> et <référent-entité>. Les raisons de ce choix sont en grande partie dues au fait d'un besoin de clarté au niveau des écritures.

Par abus de langage, il nous arrive fréquemment d'appeler cette entité :

- l'entité <type-entité, référent-entité>
- l'entité '<référent-entité>'
- ou même le <type-entité> '<référent-entité>'.

Exemple :

Pour la phrase « Le chat est dans la maison », la DCAS fait intervenir l'entité instance chat12 qui est associé, dans notre cas, au chat cible de la relation spatiale donnée.

2) Le rôle et le numéro de rôle de l'entité

Les renseignements concernant la participation de l'entité au sens d'une DCAS sont donnés par : <rôle-entité> et <numéro-rôle>.

Cette contribution s'exprime au travers d'un rôle que joue cette entité à l'intérieur de la DCAS. <rôle-entité> permet de préciser cette donnée et <numéro-rôle> la complète dans certaines situations que nous préciserons plus tard.

Exemple :

Dans la DCAS associée à la phrase « Le chat est dans la maison », l'entité instance chat12 joue le rôle 'cible' dans la DCAS. Le numéro de rôle est '*' en raison de l'unicité du rôle cible pour la relation spatiale 'dans'.

Pour la DCAS de la phrase « Le chat est entre le lit et le siège. », il y aura deux entités jouant le même rôle : celui de site. Le numéro de rôle associé à l'entité instance représentant le lit est 1, celui de l'entité instance représentant le siège est 2.

3.3. Syntaxe de construction d'une DCAS

Nous utiliserons la syntaxe décrite dans la figure ci-dessous pour écrire une DCAS.

Les caractères utilisés pour exprimer l'optionnalité sont : { et }.

Les symboles terminaux sont mis en gras.

Les symboles non terminaux sont marqués par les séparateurs < et >.

```
<description> ::=
            {<type-entité/rôle/numéro-d-ordre/référent-entité>
               { , <type-entité/rôle/numéro-d-ordre/référent-entité> }
            }

<type-entité/rôle/numéro-d-ordre/référent-entité> ::=
            (<type-entité>, <rôle-entité>,
             <numéro-d-ordre>, <référent-entité>)

<type-entité> ::= <type-entité-constant> |
                  <type-entité-variable>
<type-entité-constant> ::= <identificateur-fixe>
<type-entité-variable> ::= <identificateur-variable>

<rôle-entité> ::=  <rôle-entité-constant> |
                   <rôle-entité-variable>
<rôle-entité-constant> ::= <identificateur-fixe> | * (nil)
<rôle-entité-variable> ::= <identificateur-variable>

<numéro-d-ordre> ::=  <numéro-d-ordre-constant> |
                      <numéro-d-ordre-variable>
<numéro-d-ordre-constant> ::= <nombre> | * (nil)
<numéro-d-ordre-variable> ::= <identificateur-variable>

<référent-entité> ::= <référent-entité-constant> |
                      <référent-entité-variable> |
                      <référent-entité-complexe>
<référent-entité-constant> ::= <nombre> | <identificateur-fixe>
<référent-entité-variable> ::= <identificateur-variable>
<référent-entité-complexe> ::= <description>
<identificateur-fixe> ::= <lettre-minuscule> {<lettre-ou-chiffre> }
<identificateur-variable> ::= <lettre-majuscule> {<lettre-ou-chiffre> }
```

figure 3.1 : Syntaxe de construction des DCAS

Quelques exemples de DCAS :

```
1)    { (concept, r1, *, chien) , (concept, r2, *, chat) }

2)    { (VARIABLE_TE, r5, *, 150)}

3)    { (concept, subsumé, *, chien),
        (relation, prédicat, *, est-un),
        (concept, subsumeur, *, animal)
        (vdv,*,*,vrai)  }
```

Tous les objets que nous pouvons créer à partir de cette grammaire ne sont pas obligatoirement des DCAS, car ils ne sont pas nécessairement porteurs de sens.

3.4. Les entités cognitives

3.4.1. Introduction

Avant de décrire chacune des entités utilisées, il nous paraît opportun de rappeler la nécessité d'employer des "objets abstraits" indépendamment de tout formalisme.

Frost [Frost, 1986] introduit la notion de "vue de l'univers". Mis à part le formalisme utilisé, il paraît nécessaire, voire indispensable, à tout modéliseur (personne chargée du travail de représentation de connaissances percevant un univers donné et conceptualisant la partie de cet univers qu'il veut représenter) d'utiliser une notation à base de termes tels que "entité", "relation", "ensemble", "négation" s'il veut communiquer ses connaissances à une tierce personne.

Frost [Frost, 1986] identifie un ensemble de concepts sémantiques fréquemment utilisés (figure 3.2).

entity
entity set
attribute
attribute set
relation
function
the truth-values 'true', 'false', 'unknown'
time
proposition
name
variable
logical negation
conjonction
disjunction
->
possible world
possible truth
necessary truth
strict implication
agent
proof
rule

figure 3.2 : Liste de concepts sémantiques donnée par Frost

Cependant, Frost reconnaît lui-même que cette liste doit être améliorée. Malgré son imperfection, il lui paraît utile de présenter cette liste pour que les personnes qui abordent de nouveaux formalismes aient en tête les différents concepts sémantiques sur lesquels ces formalismes reposent.

Avant d'aller plus avant dans la liste des entités cognitives que nous utilisons, nous décrirons les entités cognitives utilisées par d'autres formalismes.

La dépendance conceptuelle de Schank (voir chapitre précédent) introduit les entités cognitives suivantes :

- les PPs (*Picture Producer*)
- les Acts
- les LOCs
- les Ts
- les AAs (*Action Aiders*)
- les PAs (*Picture Attribute*)

Le regroupement de ces entités à l'aide des notions d'acteur, d'objet, de bénéficiaire, de direction, d'état et d'instrument, donne dans la terminologie schankienne des conceptualisations qui sont les unités de base de son niveau conceptuel. L'association de ces différentes entités en respect avec des règles syntaxiques et sémantiques de combinaison procure des objets sur lesquels des inférences peuvent être réalisées.

Les systèmes de représentation dits de type KL-ONE manipulent les entités suivantes :

- les concepts (génériques, individuels, primitifs, définis),
- les rôles, ainsi que les différentes opérations de restriction, différentiation applicables sur ces derniers,
- les descriptions structurales,
- les instances.

3.4.2. Les différents types d'entités

Parmi les entités cognitives que nous décrirons, il en existe qui sont décrites au moyen d'autres. Ce qui nous amène à la répartition en deux classes d'entités :

- les entités simples,
- et les entités composées.

Nous ne prétendons pas décrire l'ensemble de toutes les catégories conceptuelles existantes. Nous décrirons seulement celles qui nous paraissent les plus pertinentes.

L'ensemble des entités cognitives que nous retenons est le suivant :

- entités simples :

  concept, instance, nombre, chaîne, relation, repère-temporel (rt), valeur-de-vérité (vdv), symbole.

- entités composées :

  situation, proposition, mesure, règle, iattribut, censemble, iensemble.

3.4.3. Les entités simples

1) L'entité concept

Ces entités sont utilisées dans la grande majorité des systèmes de représentation.

Compte tenu des différents points de vue sur les concepts, que ce soit sur leur nature (intensionnelle ou bien extensionnelle), ou que ce soit au niveau des objets qu'ils dénotent (unique ou non), il est important de préciser la notion de concept que nous adopterons.

Woods [Woods, 1975] soulignait la nécessité d'utiliser deux entités intensionnelles correspondant aux deux expressions "Étoile du Soir" (*Evening Star*) et "Étoile du Matin" (*Morning Star*) n'ayant pas les même significations, mais dénotant le même objet extensionnel. (figure 3.3)

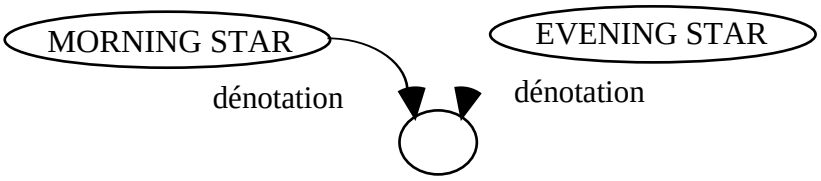


figure 3.3 : Deux entités intensionnelles dénotant un même objet extensionnel

Nous adopterons une définition de l'entité concept similaire à celle de [Brachman, 1979] :. les concepts représentent des objets intensionnels et ne doivent pas être utilisés pour représenter directement des extensions.

Par contre, il considère les concepts comme éléments de base de son formalisme. Ils permettent, en tant qu'objets formels, la représentation d'objets, d'attributs et de relations du domaine que l'on cherche à modéliser. Dans notre formalisme, l'entité concept ne joue pas un rôle aussi primordial que dans des systèmes de type KL-ONE. Même si beaucoup de ces éléments sont présents dans les DCAS, leur rôle n'en est pas pour autant central. Ils sont utilisés dans les descriptions au même titre que d'autres entités de catégorie conceptuelle différente.

Tout comme dans les systèmes de type KL-ONE, les entités concepts de notre

formalisme ne sont que des objets abstraits et peuvent décrire des objets n'ayant pas d'existence. On pourra prendre l'exemple du concept de vénusien : "la personne habitant la planète Vénus". L'existence de ce concept dans la base de connaissances ne suppose pas l'existence d'un ou plusieurs vénusiens.

Brachman distingue deux types de concepts : le **type générique** et le **type individuel**.

Le concept générique est un type au travers duquel plusieurs objets du monde peuvent être décrits (ex : homme, femme, animal). Un concept peut être formé à partir de concepts : le concept formé est appelé **sous-concept**. "Homme" est un sous-concept générique de "animal".

Les concepts individuels individualisent les concepts génériques et ne possèdent pas de sous-concepts. Le concept individuel dénote au plus un objet réel qui est une instance du concept générique.

Dans notre formalisme, cette distinction se fait au niveau d'un attribut (nature) que l'on peut associer à ce concept via une DCAS et non au niveau du type de l'entité.

Chez Brachmann, tous les objets intensionnels sont représentés par des concepts. Ici, dans le formalisme que nous proposons, des entités autres que les entités concepts peuvent être utilisées.

On utilisera l'entité-concept lorsque les deux conditions ci-dessous sont vérifiées :

- il s'agit de décrire un objet intensionnel,
- toute dénotation de cet objet intensionnel est décrite par une entité-instance.

Nous verrons par la suite quels sont les objets intensionnels représentés par les entités concepts.

Dans le formalisme proposé, ces entités participent essentiellement à la construction des différentes hiérarchies existant entre classes.

2) L'entité instance

Les entités-concepts sont liées directement à des objets abstraits qui peuvent n'avoir aucune existence réelle. Il s'agit par exemple de concepts tels que "cyclope" ou '"vénusien".

Pour parler d'un objet donné appartenant à l'univers du discours, il nous faudra utiliser une entité de type instance. Cette dernière permet de déclarer au système l'existence (dans un

univers donné, comme celui du discours) d'un objet donné.

Ainsi, pour décrire une personne que nous connaissons, nous devrons créer une entité de type instance. Ensuite, il nous sera possible de dire qu'il existe une relation entre cette entité et le concept personne en créant une DCAS.

Pour représenter l'enfant appelé Nicolas dans l'histoire, il nous faudra créer une entité-instance dont le référent sera n. Lorsque l'on voudra donner des informations à propos de Nicolas, il faudra créer des descriptions intégrant l'entité-instance n.

Si l'histoire traitée relate des faits à propos d'un cyclope particulier, il existera dans le système une description utilisant une instance du concept cyclope.

3) L'entité nombre

L'utilisation d'entités cognitives de ce type est fréquente.

Brachman [Brachman & Schmolze, 1985] utilise des nombres dans son formalisme pour décrire la cardinalité minimum et maximum de l'ensemble des instances remplissant un rôle d'un concept.

Leur usage n'est parfois pas justifié formellement.

Schank [Schank, 1975] estime que certains états d'objets peuvent être décrits par des échelles avec valeur numérique. Même s'il pense que les êtres humains n'ont pas cette manière de décrire ces états, il justifie néanmoins ce choix pour cause de nécessité d'implémentation informatique et de l'usage possible de ces états dans les inférences.

Par exemple, l'état SANTÉ est décrit par un ensemble de valeurs allant de -10 à +10. Le nombre -10 était associé à l'état de mort, et le nombre +10 à l'état de parfaite santé.

Les entités nombres dans ce formalisme sont utilisées pour décrire principalement des cardinalités et des valeurs numériques associés ou non à une dimension.

Lorsque nous voudrons décrire des connaissances comme « Un homme possède deux jambes. », l'utilisation de l'entité-nombre 2 est inévitable.

D'autres exemples peuvent être donnés :

- 2 et 2 font 4 (Cet exemple met en œuvre trois valeurs numériques sans dimensions)
- Un groupe de 3 personnes.
- Pierre mesure 1 mètre 80. (L'entité nombre 1.8 est utilisé. La dimension "mètre" sera représentée par une entité de type symbole.)

4) L'entité chaîne

Cette entité est employée de manière implicite dans la majorité des systèmes.

Un des rares systèmes à l'utiliser explicitement est celui de Maida et Shapiro [Maida & Shapiro, 1982]. En effet, elle apparaît explicitement dans leurs réseaux sémantiques propositionnels (figure 3.4). Les concepts M1 et M2 sont reliés par l'arc LEX aux mots "Morning Star" et "Evening Star"considérés comme des entités appartenant à la mémoire lexicale de leur système.

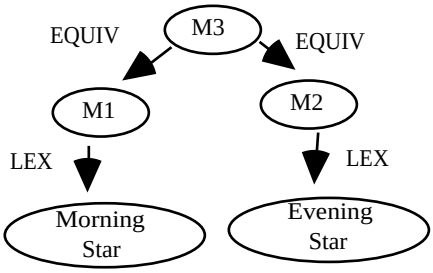


figure 3.4 : Un réseau sémantique propositionnel

Dans les DCAS, ces entités servent dans la majeure partie des cas à associer un nom à une instance donnée. L'entité-instance p associée à Pierre, par exemple, pourra elle-même être associée à l'entité-chaîne "Pierre".

Exemples d'entités chaînes

- la chaîne "Nicolas"
- la chaîne "Titi"

5) L'entité relation

Les entités relation sont généralement utilisées pour lier des entités entre elles.
Sabah cite dans [Sabah, 1988] les relations les plus couramment employées. (figure 3.5)

Lien de hiérarchie	entre classes oiseau — EST_PARTIE —> canari entre classe et individu coco — EST_UN —> canari
Lien de particularisation	chien — PARTICULIER —> setter briller — PARTICULIER —> soleil
Lien d'équivalence	amusant <— EQUIVALENT —> comique
Lien de contraste	bas <—CONTRASTE —> haut,
Lien partie	membres — EST_PARTIE —> oiseau, ailes — EST_PARTIE —> membres, pattes — EST_PARTIE —> membres
Relations spatiales	nicolas — SUR —> canapé
Scénario	entrer_dans_un_restaurant — scénario —> scénario_restaurant
Succession[23]	scène_1 -- PRECEDE -> scène_2 -- PRECEDE —> scène_3
Fonction	pelle — FONCTION —> creuser
Autres	comparaison coyote — COMME —> loup

figure 3.5 : **: Relations les plus couramment utilisées.**

Dans les réseaux sémantiques, ces relations sont représentées par des arcs orientés et relient un nœud à un autre.

[23] Il s'agit d'une approche simplifiée des aspects temporels.

Dans les graphes conceptuels de Sowa, une relation peut être reliée à un ou plusieurs concepts. Le nombre de concepts auxquels elle est reliée, définit son arité. (figure 3.6).

La relation unaire "PSBL"[24]	(PSBL) ↓ [PROPOSITION : [PERSON: Tom] <- (AGNT) <- [PARTIR]] « Il est possible que Tom parte. »
La relation binaire "sur"	[livre:#125] — (SUR) —[table:#126] « le livre est sur la table. »
La relation tertiaire "entre"	[moto] — (betw) — [camion] | [voiture] « la moto est entre le camion et la voiture. »

figure 3.6 : Exemples de relations rencontrées dans le formalisme de Sowa

La relation d'ordre que définit Sowa s'étend aussi sur les différents types de relations conceptuelles. La relation "on" peut être vue comme un sous-type de la relation "loc".

Chez Brachman, les relations qui existent entre les concepts sont définies dans la structure d'un concept et sont représentées par ce qu'il appelle des rôles. Ces rôles décrivent les relations entre les instances de ce concept et les instances d'autres concepts.

Les entités-relations du formalisme proposé permettent de relier des entités distinctes entre elles.

Elles permettent de relier des entités de catégorie cognitive aussi bien similaires que différentes. Par exemple, l'entité-relation "est_un" est utilisée pour décrire aussi bien la relation entre entités-concepts que entre entité-concept et entité-instance. Ces deux relations qui n'ont pas le même statut représentationnel sont différenciées dans le raisonnement.

Les entités -relations "dans", "sur", "sous" prennent part dans des DCAS représentant des localisations spatiales où interviennent les relations spatiales "dans", "sur", "sous".

[24] PSBL : c'est la relation "Possibilité".
Elle est proposée par Sowa pour intégrer des raisonnements proches de la logique modale.

6) L'entité repère-temporel

Ce sont des objets qui s'utilisent généralement par paires. Ils permettent de situer dans le temps des événements. Les relations existant entre ces objets permettent de situer des événements entre eux dans le temps.

La forme de raisonnement qui s'applique à ces objets est appelée raisonnement temporel. De nombreuses théories ont découlé de l'étude de l'aspect temporel de la représentation des connaissances.

L'étude de ces formalismes dépasse le cadre actuel de cette thèse. Un travail relativement important a été fait dans ce sens dans la thèse de Myriam Bras [Bras, 1990]. Cependant, nous présenterons rapidement les différentes théories pour permettre de situer la gestion du temps dans notre formalisme par rapport aux théories existantes.

La théorie de J.F. Allen ([Allen, 1981], [Allen, 1983], [Allen, 1984]) :

Cette théorie se base sur des éléments temporels primitifs qui sont des intervalles et des éléments atemporels qui sont de trois types : les propriétés, les événements et les processus. La liaison entre un éléments atemporel (une proposition A, par exemple) et un élément temporel (l'intervalle I) est assurée par des prédicats (HOLDS, OCCUR, OCCURING) ayant une sémantique bien précise.[25]

Allen définit 13 relations binaires possibles entre ces intervalles : égale, avant, après, touche, touché-par, recouvre, recouvert-par, débute, débuté-par, contenu-dans, contient, termine, terminé-par.

Ces relations sont toutes mutuellement exclusives. (Une seule de ces 13 relations est possible entre deux intervalles).

Lorsque les extrémités d'un intervalle ne sont pas connues avec certitude, il est possible d'utiliser des relations complexes. Chaque relation complexe est formée par une disjonction de 2 à 13 relations de base. Lorsque l'on ne connaît aucune information sur les intervalles, la disjonction utilisant toutes les relations de base est alors appliquée.

A partir de relations explicitement décrites entre intervalles, il est possible de déduire de nouvelles relations entre les intervalles. Si pour les trois intervalles A, B et C, A est avant B, et B lui même est avant C, on peut en déduire alors que A est avant C.

---

[25] HOLDS(P,I) est vrai si et seulement si la propriété P est vraie pendant tout sous-intervalle de I. Et HOLDS(P,I) est vérifié si et seulement si, pour tout intervalle J tel que J contenu dans I, HOLDS(A,J) est vérifié.

OCCUR(E,I) signifie que l'événement E survient pendant l'intervalle I.

OCCURING(P,I) n'est vrai que si le processus P se déroule pendant un nombre suffisant de sous-intervalles de I..

Allen [Allen, 1983] décrit 144 propriétés de composition des relations. Cette fermeture transitive est le principal outil de déduction.

La théorie de D. McDermott ([McDermott, 1982])

Elle est fondée sur l'existence d'une structure temporelle composée d'instants. Ici, l'objet temporel de base est un état qui représente un instant, un moment de l'univers.

Contrairement à la représentation de J. F. Allen, celle de D. McDermott permet de manipuler l'aspect numérique du temps. Il est en effet possible de gérer des contraintes numériques sur les durées.

Nous n'entrerons pas plus dans le détail des entités manipulées ici car les visées du système de raisonnement de McDermott ne concernent pas directement les applications traitant du langage naturel.

La théorie de Shoham ([Shoham, 1987])

Shoham rejette l'utilisation d'intervalles en tant que primitive temporelle.

Comme chez McDermott, la primitive temporelle est le point du temps.

Cette primitive lui permet de considérer des intervalles de temps au niveau de sa structure temporelle ; chaque intervalle étant défini par deux points.

Shoham définit un type unique d'assertions atemporelles, les propositions. Il ne propose ni la dichotomie faits-événements de McDermott, ni la trichotomie de Allen.

L'association d'une proposition atemporelle (par exemple, p) à un élément de la structure temporelle (par exemple, l'intervalle [t1,t2]) se fait par un type unique de lien. Elle se fera par la formule : TRUE(t1,t2,p)

TRUE ne doit pas être considéré comme un prédicat, ni comme un opérateur modal. Il sert simplement à la visualisation de l'association. [Bras 90]

Le langage de Shoham est donc composé de deux sortes de formules de base :

- les formules TRUE(t1,t2,p)
- les formules purement temporelles comme $t1 \leq t2$.

Le temps n'est pas un argument du prédicat de la proposition p.

Dans la formule TRUE(t1,t2,p), p n'est pas manipulée comme une constante (on dit que p n'est pas réifié). Ici, on a accès à chaque composante de la proposition p. Des variables utilisées comme argument de la relation constituant p peuvent alors être quantifiées. [Bras, 1990]

Par exemple, la phrase « Chaque fois que quelqu'un bat son âne, il brait. » pourrait être traduite en

$\forall x \forall y \forall t1 \forall t2$ TRUE(t1,t2, possède(x,y)) $\wedge$ TRUE(t1,t2, âne(y)) $\wedge$ TRUE(t1,t2,bat(x,y)) => $\exists$ t3 TRUE (t2,t3, brait (y))

Dans p, l'interprétation des constantes ne dépend pas du temps.

Même si p ne contient pas d'objets temporels, l'interprétation de la relation et de toutes les fonctions contenues dans p dépend de l'intervalle de temps auquel p est rattaché.

En effet, le terme "Président(USA)" renvoie à des entités différentes suivant l'intervalle de temps considéré. Par contre, la constante "USA" ne varie pas avec le temps.

Les repères temporels dans les DCAS.

L'entité repère temporel que nous utilisons dans notre formalisme peut être considéré comme la primitive temporelle proposée. Une entité repère temporel renvoie à un point du temps. Pour indiquer l'intervalle de temps durant lequel une situation est valide, nous utilisons deux entités de ce type. Une de ces entités définit le début de l'intervalle, l'autre la fin. Lorsqu'une DCAS contient deux entités repère temporel, cela signifiera que cette DCAS est temporalisée. Comme une DCAS est considérée comme un tout, tout raisonnement sur une DCAS temporalisée tient donc compte du temps.

7) L'entité vdv

L'information négative est très souvent représentée de manière implicite.

Les théories telles que :

- l'hypothèse de monde clos [Reiter, 1978a], (l'absence d'une information donnée dans la base de connaissances entraîne alors la négation de cette information)
- la "négation par l'échec" [Clark, 1978] (non A est vrai si il y a toujours échec pour prouver A)

permettent la représentation implicite des informations négatives. Elles se révèlent suffisantes pour certains types d'applications (comme la robotique) mais peu efficace pour la représentation de phrases du langage naturel.

La nécessité de représenter explicitement des informations négatives pour pouvoir les manipuler ne nous paraît donc pas inutile.

C'est grâce à l'incorporation d'entités de type vdv (valeur de vérité) que nous associerons une valeur de vérité à une situation ou à une proposition. A l'intérieur d'une description, cette entité nous permet d'assigner une valeur de vérité à toute description.

Nous utiliserons les trois entités suivantes :

- la vdv 'vrai'. (voir Maida et Shapiro)
- la vdv 'faux'.

- la vdv 'inconnu'.

8) L'entité symbole

Cette entité est utilisée conjointement à la notion d'attribut. Les valeurs des attributs ne se limitent pas à des nombres, ni à des chaînes de caractères.

Nous décrirons dans les parties qui suivent d'autres utilisations de ce type d'entité ; comme par exemple, l'association d'une description DCAS à une entité de type composé.

3.4.4. Les entités composées

Le regroupement d'entités simples dans une DCAS est une connaissance en elle-même : c'est l'entité de type composé. Les connaissances constituées par ces regroupements d'entités présentent des régularités qui permettent la constitution d'une classification de ces connaissances. Deux connaissances d'une même classe renvoient à des entités de même catégorie cognitive.

1) L'entité iattribut

Soient les quatre phrases suivantes :

La longueur du terrain est de 500 mètres. (1)
La longueur du terrain est supérieure à 500 mètres. (2)
La longueur du terrain est supérieure à la largeur du terrain. (3)
La longueur du terrain est inférieure à la longueur du stade. (4)

Pour (1), la première représentation à base de réseaux sémantiques pourrait être la suivante (figure 3.7) :

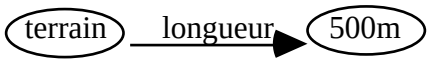


figure 3.7

Fixer à l'extrémité de l'arc la valeur de l'attribut longueur du terrain n'est pas une bonne solution. Elle rend difficile l'élaboration de représentations pour (2), (3) et (4).

On ne pourra envisager un réseau comme celui ci-dessous (figure 3.8):

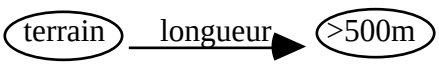


figure 3.8

La solution la plus adéquate part de l'idée suivante : chacune de ces quatre phrases traite d'une entité commune ; c'est l'entité iattribut "longueur du terrain".

Une entité iattribut dans le formalisme des DCAS est le regroupement dans une DCAS

de deux entités : une entité concept relative à l'attribut décrit et une entité (instance, en général) à qui appartient cet attribut.

En (1), l'entité iattribut "longueur du terrain" est en relation d'égalité implicite avec une mesure. La variation en (2) se fait uniquement au niveau de l'opérateur utilisé. Ce n'est plus l'égalité, mais la relation de supériorité qui est ici employée. Pour (3), il s'agit d'une comparaison entre deux entités distinctes (figure 3.9) mais néanmoins proches car elles présentent deux aspects (longueur, largeur) d'une même entité : le terrain.

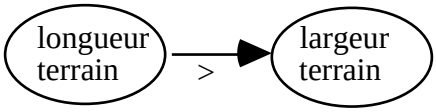


figure 3.9

Quand à (4), nous nous trouvons devant la comparaison d'un même aspect de deux entités différentes (figure 3.10).

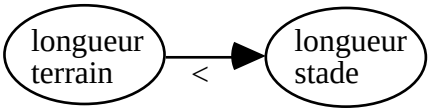


figure 3.10

L'entité associée à la longueur du terrain est composée de deux informations :

- celle relative à l'entité terrain qu'elle décrit
- et celle relative à l'attribut longueur.

Les entités de ce type ne sont pas toutes comparables entre elles. Le poids de John et la longueur du terrain ne peuvent être comparés entre eux. Par contre, on peut trouver des relations entre la longueur du terrain et la taille de Pierre.

2) L'entité mesure

Reprenons la phrase :

La longueur du terrain est de 500 mètres. (1)

Nous avons eu besoin de créer une entité correspondant à "longueur-du-terrain".

L'information "500 mètres" peut être vue comme la combinaison d'une entité nombre pour "500" et une entité symbole pour "mètres".

Une entité mesure dans le formalisme des DCAS est le regroupement dans une DCAS de deux entités : une entité nombre correspondant à la valeur numérique de la mesure représentée et une entité symbole relative à l'unité de mesure utilisée.

Exemples d'entités de type mesure.

- 25 degrés
- 1 kilogramme
- 80 kilomètres/heure

Les entités symboles utilisées sont respectivement "degré", "kg" et "km/h".

3) Les entités composées situation et proposition

a) L'entité situation

La plupart des phrases, que nous rencontrons dans le langage naturel, donnent des informations sur des événements, des états ou des processus.

La situation apparaît en tant qu'objet à la suite des travaux de Barwise et Perry [Barwise & Perry, 1983].

Wilensky [Wilensky, 1991a] utilise ce terme de manière informelle pour désigner une catégorie (*superordinate category*) regroupant les actions, les événements, les états et les processus et tout ce qu'une phrase peut décrire. Pour des raisons aussi bien théoriques que pratiques, il lui semble utile de concevoir des situations comme des entités auxquelles on se référerait ou à propos desquelles on pourrait discourir.

Pour appuyer son affirmation, il montre l'insuffisance de la forme logique (2) pour représenter la phrase (1)

Jan donna à Lynn un cigare .	*(1)*
Donner(jan, Lynn, cigare1)	*(2)*
Cela rendit Teddy furieux.	*(3)*
Rendre(Donner(jan, Lynn, cigare1), furieux)	*(4)*
Le fait que Jan donne à Lynn un cigare rendit Teddy furieux.	*(5)*

En effet, il lui apparaît nécessaire de fournir un objet associé à sa sémantique si l'on voulait décrire (3). La structure logique (2) se montre insuffisante car elle ne fournit aucun objet désignant l'événement rapporté. Même si l'on est tenté par (4) pour représenter (3), on n'est pas sûr qu'il s'agit bien du même événement qui rendit Teddy furieux.

La seconde raison invoquée provient des variation linguistiques. Dans la phrase (5), le fait que « Jan donne à Lynn un cigare » semble similaire à (1). Il y a malgré tout des nuances. Dans (1), le locuteur asserte l'existence de cette situation. Dans (5), on présuppose cette action.

De même, pour pouvoir rajouter des informations ultérieures à propos d'une situation, il nous faut utiliser à un moment ou un autre, une référence à l'événement décrit.

Notre définition (tout aussi informelle) de situation est beaucoup moins large que celle de Wilensky. Nous utiliserons ce terme pour décrire des actions, des événements, des états.

Le référent d'une entité situation est un objet qui renvoie à la DCAS qui la décrit.

Nous utiliserons une entité-situation (dont le référent est l-oiseau-vole) pour décrire le vol d'un oiseau particulier.

Des éléments d'informations supplémentaires comme le lieu de l'action, les liens entre situation (cause-conséquence, précédence, succession, etc.) peuvent être donnés au sujet des situations.

b) L'entité proposition

Nous retrouvons cette entité dans les réseaux appelés "**réseaux sémantiques propositionnels**" (*propositional semantic networks*) chez [Schubert, L., Goebel, R., Cercone, N. , 1979].

Tous les nœuds de ce type de réseau ne sont pas nécessairement des concepts. Le statut de certains est celui de propositions (figure 3.11).

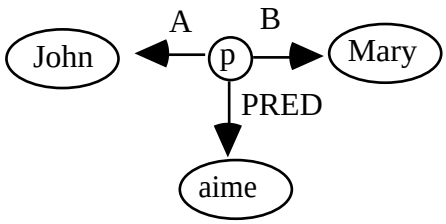


figure 3.11 : **Réseau sémantique propositionnel représentant « John aime Mary » de [Schubert, L., Goebel, R., Cercone, N. , 1979]**

Le nœud ⓟ est un nœud propositionnel central, tandis que John Mary et aime ont le statut de nœud concept.

c) Les différences entre les entités situation et proposition

Au vu de l'exemple ci-dessus, la différence entre situation et proposition n'est pas très explicite.

Wilensky [Wilensky, 1991b] remarque qu'à certaines occasions, les situations furent même confondues avec les propositions. Pour un système utilisant des situations, il établit

qu'il reste toujours une nécessité de faire appel à des objets représentant des propositions. En effet, selon lui, la phrase (1) fait appel à un élément de type proposition.

John sait que Jan donna à Lynn un cigare. (1)

<< A situation is just the wrong kind of entity to be known or believed, just as a person or physical object cannot be known or believed (in the pertinent sense). More formally, if we tried using situations for this purpose, exactly what is known or believed about the situation becomes difficult to determine.>> [Wilensky, 1991b]

Il indique aussi une propriété des propositions : elles sont fermées, contrairement aux situations, et donc reconnaissables par leur contenu.

<<Unlike situations, propositions are completely bounded in extent and are thus identical identical given given the same contxt. That is, there is only one proposition that Jan went to New York, and that Jan went to New York is its entire contents; however, the event of Jan's going to New York might have any number of other components..>> [Wilensky, 1991b],

Pour nous aider à distinguer une proposition d'une situation, nous dirons que :

- l'objet de ce que pense ou croit une personne est généralement une proposition
- tandis que ce que veut ou craint une personne est plutôt une situation.

Une situation décrite n'est pas nécessairement réelle puisqu'elle peut être crainte ou voulue. Une situation (fictive ou réelle) peut être localisée dans l'espace. Par contre, l'objet de ce que l'on croit l'est difficilement.

Prenons les énoncés ci-dessous :

Georges casse une chaise (1)
Nicolas pense que Georges casse une chaise. (2).
Nicolas veut que Georges casse une chaise. (3)
Nicolas pense que Georges casse une chaise dans la maison. (4)
Nicolas pense que c'est dans la maison que Georges casse une chaise. (5)

La phrase (1) sera traduite par une situation.

Dans (2), l'objet pensé est la proposition associée à (1) Dans l'énoncé (3), c'est la situation associée à (1) qui est voulue. La phrase (4) correspond selon nous à l'union de (2) et (5). Elle se traduit par deux descriptions : l'une mettant en œuvre une proposition correspondant à (1), l'autre une proposition indiquant que la situation correspondant à (1) est localisée dans une maison particulière.

4) L'entité règle

La plupart des systèmes de représentation utilise des règles ; principalement, les systèmes basé sur un formalisme logique, et les systèmes experts à base de règles de production.

Les règles ne sont pas toujours codées aussi explicitement que dans les systèmes experts. Dans certains cas, elles sont codées procéduralement.

Elle permettent :

- soit de créer de nouvelles informations dans la base de connaissances.
  Dans le micro-monde des blocs, on peut trouver par exemple diverses règles de mise à jour de la position des objets après l'exécution d'une action de base.
- soit de réaliser des raisonnement du type modus-ponens.

Dans le formalisme des DCAS, les entités règles sont utilisées pour le raisonnement en modus-ponens. Elles se présentent sous le forme d'une DCAS contenant elle-même d'autres DCAS. Chaque (sous-)DCAS représente soit un antécédent de la règle, soit un conséquent de la règle.

Nous analyserons ultérieurement les raisonnements autour d'une base de DCAS.

5) Les entités iensemble et censemble

Si nous avions à représenter la phrase (1),

Jean et Marie possèdent ensemble une maison. *(1)*

deux solutions sont envisageables :

- représenter les deux phrases (1') et (1").
  Jean possède une maison. *(1')*
  Marie possède une maison, et c'est la même que Jean. (1")
- utiliser la notion d'ensemble. Les propriétaires de la maison sont ici regroupés dans un ensemble contenant Jean et Marie.

Dans une représentation à graphes conceptuels au sens de Sowa [Sowa, 1984], nous aurions obtenu le graphe (2) pour représenter (1).

[PERSONNE : set('jean', 'marie')] <- (POSS) <- [MAISON:#m12] (2)

Ici, [PERSONNE:set('jean', 'marie')] est l'ensemble d'individus [PERSONNE, 'jean'],

[PERSONNE, 'marie'] ainsi que d'autres individus non spécifiés. Cet ensemble est dit partiellement spécifié. Sowa propose d'indiquer la cardinalité d'un ensemble à l'aide du symbole @. Un ensemble de 4 personnes sera manipulé au travers de la structure [PERSONNE: @4]. Tous les éléments d'un ensemble peuvent participer de différentes manières à une relation :

- collective (participation de tous dans (1))
- distributive (participation séparée dans « Jean et Marie mangent »)
- et respective (« Jean et Mary possèdent respectivement une maison et une voiture »).

Hendrix [Hendrix, 1979] utilise aussi la notion d'ensemble dans ses réseaux partitionnés. Des relations comme l'appartenance (les relations "e" et "de") et l'inclusion (les relations "s" et "ds") ensemblistes y étaient prédéfinies. La relation "e" signifie "élément de l'ensemble", "de" "élément distinct de l'ensemble", "s" "sous-ensemble de l'ensemble" et "ds" "sous-ensemble distinct de l'ensemble".

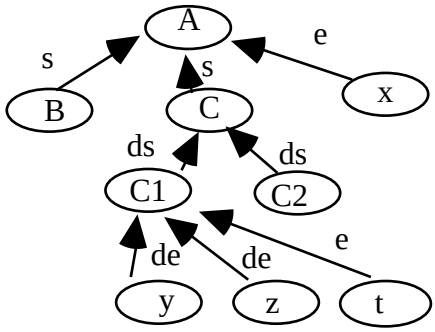


figure 3.12 : Un Réseau partitionné de Hendrix

La figure 3.12 indique l'existence de 5 ensembles (A, B, C, C1, C2) et de 4 éléments (x, y, z, t). En termes mathématiques, elle s'interprète ainsi :

$B \subset A, C \subset A, C1 \subset C, C2 \subset C, x \in A, y \in C1, z \in C1, t \in C1$

et $y \neq z$, $C1 \cap C2 = \varnothing$.

L'intersection entre B et C n'est pas forcément vide.

Rien n'empêche à l'élément x d'appartenir à B ou C ou C1 ou C2. x ne peut appartenir en même temps à C1 et C2. L'élément t n'appartient pas à C2, mais appartient à C et à A et pourrait être élément de B. Les éléments y et z n'appartiennent pas à C2. L'élément t peut être égal à y ou z, mais pas aux deux en même temps.

L'idée centrale dans les réseaux partitionnés était de permettre le regroupement de nœuds et arcs dans un objet que Hendrix appelle "espace". Ces objets lui permettent d'aborder des notions peu clarifiées dans les réseaux sémantiques : la représentation de

structures logiques (conjonction, disjonction, négation et implications) et de quantificateurs (∀, ∃).

Dans le formalisme que nous proposons, nous utiliserons deux entités relatives à la notion d'ensemble : les entités **censemble** et **iensemble**.

Dans le formalisme des DCAS, une entité iensemble décrit un ensemble d'instances **réelles** (le "i" de iensemble provient de "instance") en indiquant les propriétés vérifiées par la cardinalité de cet ensemble, les propriétés communes (concept dénoteur, etc...) ou particulières (en indiquant explicitement les instances membres) de ses instances. Avec une entité iensemble, on peut par exemple décrire un ensemble de chemises de couleur blanche.

Dans le formalisme des DCAS, une entité censemble décrit aussi un ensemble d'instances. A la différence de l'entité iensemble, les instances décrites sont hypothétiques. Cela permet de décrire des connaissances dans le cas de l'existence de ce groupe d'instances.

Ainsi, pour représenter (1), nous utiliserons un iensemble. Ce dernier sera décrit en donnant entre autre des informations sur les caractéristiques des différents éléments : leur type, le concept commun qui les décrit, leur nombre minimum, maximum ou effectif.

des personnes attendent. *(3)*
des personnes malades attendent. *(4)*
le livre et la table sont dans la chambre. *(5)*

Dans (3), l'iensemble correspondant indiquerait que le nombre d'éléments est 2, et que ce sont toutes deux des instances de personnes. Dans (4), on retrouvera le même iensemble avec en complément l'information supplémentaire sur l'état de santé des personnes. Avec (5), nous aurons un iensemble décrit uniquement par les instances qui en sont membres : l'instance de livre "livre150" et l'instance de table "table12".

Ces entités se révèlent pratiques car il n'est pas nécessaire de donner les éléments de manière précise ; une description un peu vague peut se révéler tout à fait suffisante. Ainsi, pour décrire (6), il n'est pas indispensable de dire que jean possède les instances de chemises "chemise1" à "chemise20".

Jean a 20 chemises. (6)

La phrase (6) est équivalente à « Jean possède un ensemble de chemises, dont le nombre est 20 ». Si une des chemises est blanche, on trouvera dans la base de connaissances des descriptions relatives à l'existence d'une chemise blanche et à son appartenance à l'iensemble

décrit ci-haut.

L'entité de type censemble est utilisée pour décrire des faits génériques.
Lorsque nous pensons à des phrases comme (7), (8)

Chaque père de famille a 1 enfant. (7)
Chaque père de famille a 1 devoir (celui de bien élever ses enfants). (8)

on conçoit bien que le type iensemble, suffisant pour (8) car il s'agit du même devoir pour tous les pères, est inadéquat dans (7) pour représenter "1 enfant". Dans (7), il est donné en fait, une description générique de tous les iensembles de un enfant.

Cette notion de censemble nous permettra aussi de représenter des expressions quantifiées universellement. A "chaque père de famille", on pourra associer un censemble décrivant tous les pères de familles. La phrase (7) pourra finalement donner lieu à une description composite associant deux censembles, (1 censemble pour "chaque père de famille", et 1 censemble pour "1 enfant").

3.4.5. Les rôles des entités dans les descriptions

Toute description DCAS fait appel à des entités. La contribution de chaque entité au sens associé s'exprime à travers les rôles associés.

Un rôle permet de dire comment les différents éléments interviennent dans le sens associé à une description.

Soit la phrase suivante :

Nicolas est dans la maison. (1)

Pour représenter (1), nous utiliserons trois entités.

- l'instance 'personne1' pour l'instance de personne qui s'appelle Nicolas
- l'instance 'maison1' pour l'instance de maison
- la relation 'dans'.

La description (2)

```
{     (instance, *, *, personne1),
      (relation, *, *, dans),
      (instance, *, *, maison1)}      (2)
```

se révèle à première vue suffisante.

Sachant que personne1 est une personne, et que maison1 est une maison, on pourra affirmer sans hésitation que cette description correspond à (1) et non à (3). En effet, la situation (3) n'est pas possible.

La maison est dans Nicolas. (3)

Par contre, si on examine attentivement (4),

Le livre est sur le cahier. (4)

nous ne saurions dire avec la description (5)

```
{     (instance, *, *, livre1),
      (relation, *, *, sur),
      (instance, *, *, cahier1)}      (5)
```

qui est sur quoi.

Ici un "raisonnement à partir du sens commun" ne pourra pas nous aider comme il l'avait fait dans l'exemple ci-dessus.

Dans toutes les situations décrivant des relations spatiales entre éléments, nous utiliserons les rôles cible et site pour les différencier dans une description.

Les définitions de cible et de site retenues sont celles de [Vandeloise, 1986].

La description (6)

```
{     (instance, cible, *, livre1),
      (relation, *, *, sur),
      (instance, site, *, cahier1)}    (6)
```

est plus adéquate que (5) pour représenter (4).

Il apparaît néanmoins difficile d'assigner un rôle à chaque entité participante. La relation "sur" dans (6) n'a en effet aucun rôle. Cela provient du fait qu'on choisit généralement une entité dominante dans la description (la relation spatiale "sur") et on décrit les rôles des autres entités par rapport à cet objet focus.

Il est fréquent de trouver des situations dans lesquelles l'acteur d'une action et l'objet de cette action sont les mêmes.

Dans (7),

Pierre se nettoie. (7)

l'entité-instance associée à Pierre est aussi bien acteur et objet de l'action.

Il est parfois nécessaire de distinguer des entités partageant le même rôle dans une description. Aussi, un numéro peut être associé à ces rôles. Il évite la création de rôles dérivés.

La phrase (8), si l'on utilise les rôles site et cible nous amène aux représentations (9) et (10).

Bernard est entre Albert et Claire. (8)

```
{     (instance, cible, *, b), (relation, *, *, entre),
      (instance, site1, *, a), (instance, site2, *, c)}     (9)

{     (instance, cible, *, b), (relation, *, *, entre),
      (instance, site, 1, a), (instance, site, 2, c)}       (10)
```

La description (9) présente l'inconvénient de mentionner deux rôles 'site1' et 'site2' qui semblent complètement différent de celui de (6). Cela nous obligera alors à définir ces deux nouveaux rôles. Nous interpréterons (10) de la manière suivante : l'instance a possède le rôle site et en est le premier, l'instance c possède le rôle site et en est le second.

Ce troisième champ de quadruplet permet de créer un ordre parmi tous les éléments ayant le même rôle.

L'utilisation du champ nombre dans un quadruplet implique alors l'existence d'autres entités jouant le même rôle.

3.4.6. Les référents des entités dans les descriptions

Le référent d'une entité cognitive peut être :

- soit un élément atomique qui se présente sous la forme :
  - d'un nombre (125)
  - ou d'une chaîne de caractères ("Nicolas")
  - ou d'un symbole ('homme32', 'canari12')
  - ou d'une variable (REF2, ...)
- soit une description ({(instance, *, *, I), (relation, *, *, R), (concept, *, *, C)}, ..).

Le référent d'une entité de type simple ne peut être une description.

Seuls les référents d'entités de type composé peuvent avoir comme référent des descriptions, mais ce n'est pas une condition nécessaire. En effet, nous autorisons les quadruplets suivants : (proposition, *, *, p125), (règle, *, *, r12).

3.5. Quelques exemples de descriptions utilisées pour modéliser les connaissances liées au monde du discours

Dans les paragraphes précédents, nous avons vu les diverses entités cognitives que notre formalisme se propose de manipuler, ainsi que la notion de rôle que nous associons à ces objets.

Avant de décrire précisément les mécanismes d'inférences qui sont appliquées sur les descriptions, il nous paraît utile de donner des exemples de DCAS utilisées pour modéliser les connaissances liées à un monde donné.

Tout d'abord, nous présenterons quelques définitions préliminaires que nous emploierons par la suite .

Pour chaque série de descriptions, nous décrirons leurs formes générales. Ces dernières devront être utilisées lors de la création de connaissances similaires Le non respect de ces formes conduit à l'impossible application de certaines règles internes du mécanisme d'inférences.

3.5.1. Quelques définitions préliminaires

Avant de parcourir les différentes descriptions, nous verrons :

- les notions d'ordre et de niveau de profondeur associées aux quadruplets et aux descriptions,
- la notion de type de description.

1) Notion d'"ordre" de quadruplets et de descriptions

Un **quadruplet** est dit d'**ordre 1** (ou dit du premier ordre) si et seulement si la partie référent est un élément atomique (nombre, chaîne de caractères, symbole ou variable).

Ex :

(instance, *, *, h125), (concept, *, *, homme), (relation, *, *, est_un)

sont des quadruplets d'ordre 1.

Une **description** est dite d'**ordre 1** si et seulement si tous les quadruplets qui la composent sont d'ordre 1.

Ex :

{(concept, *, *, homme), (instance, *, *, h15), (relation, *, *, est_un)}

est une description d'ordre 1.

Un **quadruplet** est dit d'**ordre n** (> 1) si et seulement si la partie référent est une description d'ordre n-1.

Ex :

(mesure, *, *, {(nombre, *, *, 100), (symbole, unité, *, mètre)}),

(iattribut, *, *, {(instance, *, *, terrain12), (concept, attribut, *, longueur)}) et

(proposition, objet, *, { (instance, *, *, h15), (concept, *, *, homme),
(relation, *, *, est_un)}

Tous ces exemples sont des quadruplets d'ordre 2.

Une **description** est d'**ordre n** (> 1) si et seulement si

- elle possède au moins un quadruplet d'ordre n et
- elle ne possède aucun quadruplet d'ordre supérieur ou égal à n+1.

Ex :

{(iattribut, *, *, {(instance, *, *, terrain12), (concept, attribut, *, longueur)}),
(symbole, opérateur, *, =),
(mesure, valeur, *, {(nombre, *, *, 100), (symbole, unité, *, mètre)}),
(vdv , *, *, vrai)
}

est une description d'ordre 2.

2) Notion de "niveau de profondeur" de quadruplets et de descriptions

La base de connaissances sur laquelle travaille le système est en fait un ensemble de descriptions. Chacune de ces descriptions est d'ordre quelconque.

Nous dirons de ces descriptions qu'elles se situent au premier niveau de profondeur, que nous numéroterons 1.

De plus, nous considérons que chacun des quadruplets d'une description de niveau de profondeur 1 appartient aussi au niveau de profondeur 1.

Lorsqu'un quadruplet de niveau de profondeur n possède comme référent une description (ce qui est le cas des entités cognitives de type complexe), on dit que cette dernière description est de profondeur n + 1.

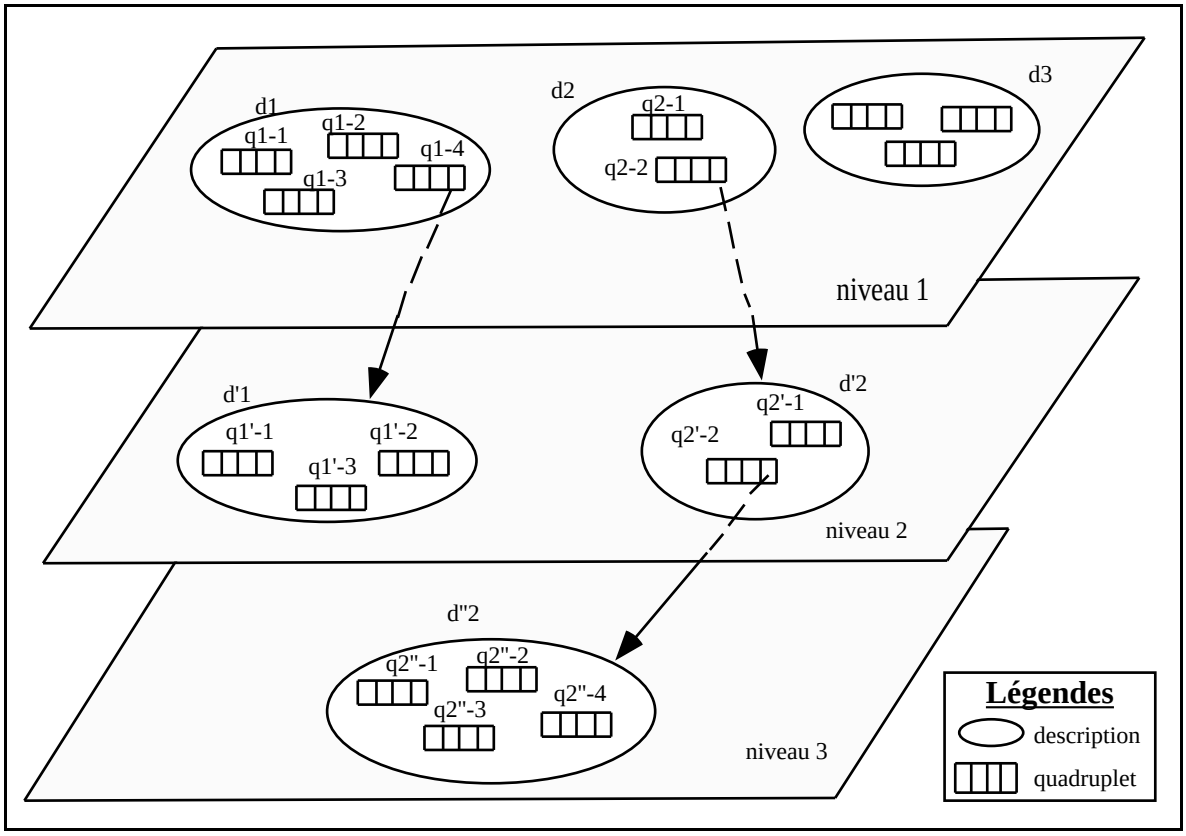


figure 3.13

A partir de la base de connaissances de la figure 3.13 contenant les trois descriptions d1, d2 et d3, il nous est possible de générer le tableau suivant (figure 3.14) :

élément	composants	niveau de profondeur	Ordre
d1	q1-1, q1-2, q1-3, q1-4	1	2
q1-1, q1-2, q1-3		1	1
q1-4	?, ?, ?, d'1	2	2
d'1	q1'-1, q1'-2, q1'-3	2	1
q1'-1, q1'-2, q1'-3	?, ?, ?, ?	2	1
d2	q2-1, q2-2	1	3
q2-1	?, ?, ?, ?	1	1
q2-2	?, ?, ?, d'2	1	3
d'2	q2'-1, q2'-2	2	2
q2'-1	?, ?, ?, ?	2	1
q2'-2	?, ?, ?, d''2	2	2
d"2	q2'"-1, q2"-2, q2"-3, q2"-4	3	1
q2'"-1, q2"-2, q2"-3, q2"-4	?, ?, ?, ?	3	1

figure 3.14

3) Notion de "type" (et référent) de description

Les descriptions que nous créons dans la base de connaissances en vue de leur traitement ne sont pas uniquement différenciables entre elles par leur ordre et leur niveau de profondeur. En effet, tout ensemble de quadruplets, qu'une description regroupe, forme une connaissance. Parmi toutes les descriptions qui peuvent se former, on peut regrouper certaines d'entre elles car les connaissances qu'elles renferment possèdent les même régularités : cela nous amène à la notion de type de description.

Lorsqu'une description est le référent d'une entité de type composé, on considère que le type de cette description est le même que celui de l'entité composée. Par exemple, nous dirons que la description

{(instance, possesseur, *, pierre1), (instance, possédé, *, maison1)}

présente dans le quadruplet

{(proposition, *, *, {(instance, possesseur, *, pierre1),
(instance, possédé, *, maison1)}

est de type proposition.

Pour des descriptions, telles que

{(instance, possesseur, *, pierre1), (instance, possédé, *, maison1)},

situées à un niveau de profondeur 1, c-a-d ne pouvant pas être le référent d'une entité composée, on peut s'interroger sur la manière de spécifier leur type.

Avant d'apporter quelques éléments de réponse, il nous paraît opportun de donner un rapide aperçu des descriptions que l'on rencontrera dans ce premier niveau de profondeur. Comme nous l'avions dit, ce premier niveau constitue l'intégralité de la base de connaissances du système. Pour des raisons de commodité d'écriture de la base de descriptions, nous avons choisi de représenter à ce niveau tous les éléments permettant au système de mettre en œuvre le mécanisme de raisonnement.

Les DCAS appartenant à une base de connaissances se partagent en deux groupes :

- un groupe de connaissances admises par le système (**GCAS**) qui se décompose en trois sous-groupes :
  - les propositions :
    « Les oiseaux sont des animaux. », « Toute instance de oiseau possède 2 ailes. »
  - les situations :
    "Nicolas est assis sur le canapé."
  - les règles :
    « Si I1 aime I2 et I2 aime I3 et I1 ≠ I3, alors I1 est malheureux. »
- un groupe de connaissances que le système ne considère pas comme sien. (**GCNAS**). Elles existent pour des fins de simplification d'écriture. Leur utilité se rapproche de celle des macro-expressions dans les langages informatiques.

Soit la phrase « Jean pense que Pierre a donné un livre à Marie. ».

Il se révèle utile de pouvoir considérer dans une premier temps la proposition « Pierre a donné un livre à Marie » à laquelle on associe l'entité-proposition 'p125', et d'indiquer que cette entité cognitive de référent 'p125' joue le rôle d'objet dans une situation où Jean est l'acteur.

proposition 125 = proposition « Pierre a donné un livre à Marie. »
situation 23 = jean pense 'p125'.

On peut observer que nous n'indiquons pas au système d'admettre la proposition 'p125', mais plutôt que la proposition 125 est liée au fait que Pierre a donné un livre à Marie.

Cela nous permet de souligner ici un fait très important : pouvoir parler de propositions quel que soit leur degré de véracité ou de fausseté.

Rien ne doit nous empêcher d'associer un référent à la proposition « Les oiseaux ne sont pas des animaux. » même si le contraire est admis par le système.

Ces propositions seraient plutôt l'objet de certaines croyances. Même si le système ne les admet pas, il peut utiliser ces connaissances dans un raisonnement donné. Si la proposition « La Terre n'est pas ronde. » fait l'objet d'une croyance d'un être humain, le système pourrait inférer sans peine que l'être en question n'est pas au courant de certaines découvertes en matière d'astronomie.

Ces connaissances que le système ne considère pas comme siennes ne sont là que pour faciliter une écriture, afin de ne pas alourdir la base de connaissances.

Dans ce groupe de connaissances, nous trouverons des propositions, des situations, des règles, des ensembles et même des mesures et iattributs.

Une **description du niveau de profondeur 1** est dite **de type <T>** si et seulement si elle contient le quadruplet suivant (symbole, type-description, *, <T>).

Ex :

La description

{(concept, *, *, oiseau), (relation, *, *, est_un), (instance, *, *, o1),
(symbole, type-description, *, proposition)}

est de type proposition.

Une **description du niveau de profondeur 1** a pour **référent <R>** si et seulement si elle contient le quadruplet suivant (symbole, référent-description, *, <R>).

Ex :

La description

{(concept, *, *, oiseau), (relation, *, *, est_un), (instance, *, *, o1),
(symbole, type-description, *, proposition), (symbole, référent-description, *, p1)}

est une proposition qui a pour référent p1.

Nous savons associer un type et un référent à une description de niveau de profondeur 1. Mais, il reste un problème à résoudre : Comment distinguer une proposition (ou situation ou règle) admise par le système d'une proposition (ou situation ou règle) non admise par le système ?

Nous signifierons qu'une **description de niveau de profondeur 1** est **admise par le**

système si elle contient le quadruplet : (*, *, *, *)

4) Notion de "type" d'un quadruplet

Par extension et pour éviter des lourdeurs dans le texte, nous dirons que tout **quadruplet** de la forme (<type-entité>, <rôle-entité>, <numéro>, <référent-entité>) est **de type <type-entité>.**

Ex :
Le quadruplet (instance, *, *, personne1) est dit de type instance.`

3.5.2. Les descriptions relatives à l'organisation des concepts

A l'intérieur d'une base de connaissances, les entités de type concept sont liées entre elles par des relations.

1) La relation est-un

Nous trouverons chez Brachman [Brachman , 1983] dans « *What ISA Is and Isn't : An analysis of taxonomic links in Semantic Networks* » une analyse intéressante des différentes utilisations de cette relation.

Cette relation induit une structure de taxonomie dans l'ensemble des entités-concepts que nous manipulons. La figure 3.15 illustre de manière graphique une structure de taxonomie entre les concepts : chose, plante, animal, minéral, mammifère, oiseau, être humain, homme et femme. L'arc qui relie deux concepts entre eux représente la relation est-un, et l'arc barré la relation "n'est-pas-un".

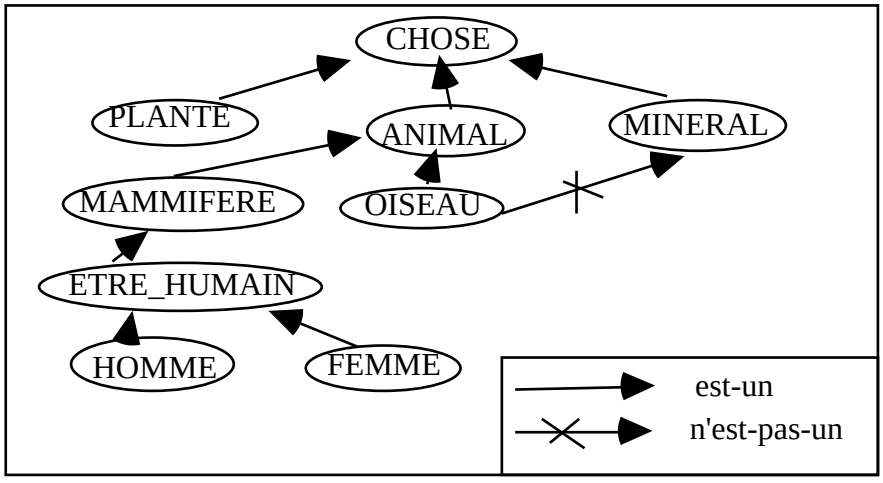


figure 3.15

Pour pouvoir décrire de telles taxonomies, nous utiliserons une forme de description permettant de relier deux concepts entre eux.

Cette description contient les 4 quadruplets suivants :

- les deux entités concepts mises en relation : ces deux entités concepts se distinguent par des rôles différents : le subsumeur et le subsumé.

- l'entité-relation "est-un".

- une entité vdv. Cette dernière sera celle de référent 'vrai' s'il s'agit de représenter la phrase « Un oiseau est un animal. », ou celle de référent 'faux' s'il s'agit de représenter « Un oiseau n'est pas un minéral. »

Nous donnerons en encadré (ci-dessous) la forme générale puis la sémantique qui lui est associée.

Forme générale :

{ (concept, subsumé, *, C1),
  (relation, *, *, est-un),
  (concept, subsumeur, *, C2),
  (vdv , *, *, V_OU_F),
}

Sémantique associée :

« Il est V_OU_F que le concept C1 est subsumé par le concept C2. »

La base de descriptions relative à la hiérarchie de concepts contient des descriptions qui ont la structure suivante :

```
{ (concept, subsumé, *, plante) , (relation, *, *, est-un),
  (concept, subsumeur, *, chose) , (vdv, *, *, vrai)
}
{ (concept, subsumé, *, animal) , (relation, *, *, est-un),
  (concept, subsumeur, *, chose) , (vdv, *, *, vrai)
}
{ (concept, subsumé, *, minéral) , (relation, *, *, est-un),
  (concept, subsumeur, *, chose) , (vdv, *, *, vrai)
}
{ (concept, subsumé, *, mammifère) , (relation, *, *, est-un),
  (concept, subsumeur, *, animal) , (vdv, *, *, vrai)
}
{ (concept, subsumé, *, oiseau) , (relation, *, *, est-un),
  (concept, subsumeur, *, animal) , (vdv, *, *, vrai)
}
{ (concept, subsumé, *, être_humain) , (relation, *, *, est-un),
  (concept, subsumeur, *, mammifère) , (vdv, *, *, vrai)
}
```

```
{ (concept, subsumé, *, homme) , (relation, *, *, est-un),
  (concept, subsumeur, *, être_humain) , (vdv, *, *, vrai)
}
{ (concept, subsumé, *, femme) , (relation, *, *, est-un),
  (concept, subsumeur, *, être_humain) , (vdv, *, *, vrai)
}
{ (concept, subsumé, *, oiseau) , (relation, *, *, est-un),
  (concept, subsumeur, *, minéral) , (vdv, *, *, faux)
}
```

2) La disjonction de concept

Parmi les relations existant entre concepts, on trouve aussi la disjonction de deux concepts. Deux concepts C1 et C2 sont disjoints quand toute instance de C1 ne peut être instance de C2, et réciproquement.

Pour représenter ces connaissances, nous emploierons des DCAS mettant en jeu :

- une entité-relation dont le référent est 'disjoint'
- une entité-vdv : vrai ou faux
- deux entités-concepts .

La relation de disjonction est symétrique car aucun rôle n'est associé à ces deux concepts.

Forme générale :

```
{(concept, *, *,        C1),
 (relation, *, *,       disjoint),
 (concept, *, *,        C2),
 (vdv , *, *,           V_OU_F),
}
```

Sémantique associée :

« Il est V_OU_F que le concept C1 est disjoint du concept C2. »

Ces connaissances nous permettront d'établir le fait que, dans le cas où V_OU_F est vrai, toute instance de C1 ne peut être également instance de C2 (et réciproquement).

Il ne faut pas confondre la disjonction de concepts avec la relation "n'est-pas-un".

En effet, si l'on considère les phrases suivantes :

Oiseau est disjoint de Minéral (ou Minéral est disjoint de oiseau) (1)
Un oiseau n'est pas un minéral. (2)
Un minéral n'est pas un oiseau. (3)
Un animal n'est pas un oiseau. (4)
Animal est disjoint de Oiseau (ou Oiseau est disjoint de Animal) (5)

on constate effectivement que (1) entraîne (2) et (3) et que par contre, (4) n'entraîne pas la proposition incorrecte (5) du fait qu'un oiseau est un animal.

3.5.3. Les descriptions relatives à la liaison instance-concept

Les entités-concepts, n'étant que des entités purement abstraites, ne nous permettent pas de décrire des objets réels, comme la maison de Nicolas ou son chat. Pour déclarer l'existence de ces objets réels, nous associerons à chacun d'eux une entité instance. Une instance en elle-même n'est pas intéressante. Il nous faut connaître le (ou les concepts) qui la dénotent.

La forme générale que nous allons décrire comprend principalement 4 éléments :

- 1 entité-instance, dénotée.
- 1 entité-relation : est-un
- 1 entité-concept, dénoteur.
- 1 vdv : vrai ou faux.

Forme générale :

{(instance, dénoté, *, I),
(relation, *, *, est-un),
(concept, dénoteur, *, C),
(vdv , *, *, V_OU_F),
}

Sémantique associée :

« Il est V_OU_F que I est une instance du concept C. »

Exemple :

La connaissance d'un téléphone réel et celle d'une personne (nommée Nicolas) dans la base de connaissances se traduiront par les descriptions ci-dessous :

```
{ (instance, dénoté, *, téléphone125), (relation, *, *, est-un),
  (concept, dénoteur, *, téléphone),  (vdv, *, *, vrai),
}
{ (instance, dénoté, *, personne1), (relation, *, *, est-un),
  (concept, dénoteur, *, personne),  (vdv, *, *, vrai),
}
```

3.5.4. Le temps dans les descriptions

Dans les descriptions précédemment étudiées, nous n'avons pas tenu compte des informations d'ordre temporel.

Dans le système proposé, les informations relatives au temps sont portées dans les

DCAS par deux quadruplets de type "repère temporel" :

- l'un a le rôle "début".
- l'autre a un rôle "fin".

Une description présentant des informations d'ordre temporel a la forme suivante:
{ .... , (rt, début, *, RT1) , (rt, fin, *, RT2) }

Ce couple de quadruplets accompagne toute description incluant des informations temporelles. La position relative de deux événements distincts dans le temps pourra se faire grâce à la connaissance des relations existant entre les quadruplets de type repère temporel respectifs.

Exemple :

Soient deux description D1 et D2 relatives à deux événements distincts.

D1 contient les quadruplets : (rt, début, *, rt1) et (rt, fin, *, rt2).

D2 contient les quadruplets : (rt, début, *, rt3) et (rt, fin, *, rt4).

Si on est sûr que rt2 est antérieur à rt3, il n'y aucune raison de ne pas croire que l'événement décrit par D1 s'est passé avant D2. Les descriptions permettant d'exprimer les relations entre les différents repères temporels utilisés, ont la forme suivante :

Forme générale :

```
{(rt, sujet, *,              RT1),
 (relation, prédicat, *,     RELATION1),
 (rt, valeur_référence, *,   RT2),
 (vdv , *, *,                V_OU_F),
}
```

Sémantique associée :

« Il est V_OU_F que RT1 est RELATION1 RT2. »

Les valeurs possibles de RELATION1 sont avant, après, égal.

3.5.5. Les descriptions permettant d'attacher une valeur à un attribut d'une entité donnée

Ces descriptions sont susceptibles de faire intervenir les deux types d'entités suivants :

- le type iattribut
- et le type mesure.

Dans un premier temps, nous donnerons les informations nécessaire à la formation de

quadruplets de ce type. Ensuite, nous décrirons la manière d'attacher une valeur à un attribut donné.

1) Forme de quadruplets de types iattribut et mesure

a) Forme d'un quadruplet de type iattribut

Un quadruplet de type iattribut aura la forme suivante :

```
(iattribut, ROLE, NUM,
        {   (TYPE_ENTITE, détenteur, *, REF_ENTITE),
            (concept, attribut, *, REF_ATTR )
        }
)
```

Un iattribut est entièrement décrit par une description-référent contenant uniquement deux entités :

- une entité de type TYPE_ENTITE et de référent REF_ENTITE dont on décrit l'attribut. Il aura le rôle de détenteur.
- et une entité concept qui joue le rôle d'attribut dont le référent est le nom de l'attribut.

Ex :

Le quadruplet associé à "la longueur du terrain t125" est

(iattribut, UN_ROLE, UN_NUMERO,
{(instance, détenteur, *, t125), (concept, attribut, *, longueur)})

Le quadruplet associé à "la couleur du téléphone tél34" est

(iattribut, UN_ROLE, UN_NUMERO,
{(instance, détenteur, *, tél34), (concept, attribut, *, couleur)})

Le quadruplet associé à "la nature du concept c12" est

(iattribut, UN_ROLE, UN_NUMERO,
{(concept, détenteur, *, c12), (concept, attribut, *, nature)})

b) Forme d'un quadruplet de type mesure

Un quadruplet de type mesure aura la forme suivante :

```
(mesure, ROLE, NUM,
        {   (nombre, *, *, N),
            (symbole, unité, *, U )
        }
)
```

Un quadruplet de type mesure est décrit dans sa totalité par une description-référent contenant deux entités :

- une entité de type nombre et de référent N,
- une entité symbole de référent U jouant le rôle d'unité.

Ex :

Le quadruplet associé à "100 mètres" est :

(mesure, ROLE, NUM, { (nombre, *, *, 100), (symbole, unité, *, mètre) })

2) La liaison attribut/valeur

Pour un attribut donné (la longueur d'un terrain, la couleur d'un téléphone), dans le meilleur de cas, la valeur exacte nous est donnée. Par exemple, la longueur du terrain est égale à 100 mètres ; la couleur du téléphone est noire. Dans ces deux cas, les descriptions se composent de 5 éléments :

- 1 entité-iattribut
- 1 entité jouant le rôle de valeur
- 1 entité vdv (vrai ou faux)
- les deux entités temporelles donnant l'intervalle de temps dans lequel la description est réalisée.

Mais, généralement, une relation existe entre l'attribut d'un objet et une valeur jouant le rôle de référence (Ce n'est pas obligatoirement la relation d'égalité). Lorsque nous déclarons « La longueur du terrain est supérieure à 100 mètres. », nous définissons une relation entre la mesure (100 mètres) et l'iattribut (longueur du terrain).

Les descriptions, permettant de faire les liaisons attribut-valeur, mettent en œuvre 5 entités :

- 1 entité-relation REF_OP jouant le rôle de prédicat : Les référents possibles seraient '=' (dans le meilleur des cas) ou '>' ou '<' ou '≥' ou '≤'.
- les deux entités que l'on compare : pour l'exemple ci-dessus, il s'agira de l'entité iattribut "longueur du terrain" et l'entité-mesure "100 mètres". Nous associons le même rôle à ces deux entités ; c'est le rôle d'argument. Pour les distinguer, nous utiliserons les rôles sujet et valeur_référence. L'entité qui a pour rôle sujet est l'entité dont on parle, et l'autre aura pour rôle valeur_référence.
- 1 entité vdv : vrai ou faux
- deux entités temporelles permettant d'indiquer à quel moment cette déclaration est valide.

Forme générale :

{(TYPE_ENTITE1, sujet, *, REF_ENTITE1),
(relation, prédicat, *, REF_OP),
(TYPE_ENTITE2, valeur_référence, 2, REF_ENTITE2),
(vdv , *, *, V_OU_F),
(rt, début, *, RT1),
(rt, fin, *, RT2)
}

Sémantique associée :

« Il est V_OU_F que l'entité <TYPE_ENTITE1, REF_ENTITE1> soit REF_OP à l'entité <TYPE_ENTITE2, REF_ENTITE2> entre les repères temporels RT1 et RT2. »

Exemple :

A la phrase « La longueur du terrain dépasse 100 mètres. », nous associerons la description :

```
{ (iattribut, sujet, *,
    {(instance, détenteur, *, t125),
     (concept, attribut, *, longueur)}),
  (relation, prédicat, *, >),
  (mesure, valeur_référence, *,
    {(nombre, *, *,  100),  (symbole, unité, *, mètre ) }),
  (vdv , *, *, vrai),
  (rt, début, *, rt32),
  (rt, fin, *, rt33)
}
```

A la phrase « La couleur du téléphone téléphone34 est noire. » nous associerons la description :

```
{ (iattribut, sujet, *,
    {(instance, détenteur, *, téléphone34 ),
     (concept, attribut, *, couleur)}),
  (relation, prédicat, *, =),
  (symbole, valeur_référence, *, noir),
  (vdv , *, *, vrai),
  (rt, début, *, rt34),
  (rt, fin, *, rt35)
}
```

Pour dire qu'une personne donnée s'appelle Nicolas, nous utiliserons :

```
{ (iattribut, sujet, *,  {(instance, détenteur, *, personne12),
                          (concept, attribut, *, nom)}),
  (relation, prédicat, *, =),
  (chaîne, valeur_référence, *, "Nicolas"),
  (rt, début, *, rt36),
  (rt, fin, *, rt37)
}
```

Nous noterons que le rôle de valeur_référence peut être aussi joué par une entité de type

iattribut. Le fait que Nicolas soit plus grand que Pierre sera représenté par :

```
{  (iattribut, sujet, 1,
      {(instance, détenteur, *, personne12),
       (concept, attribut, *, taille)}),
   (relation, prédicat, *, >),
   (iattribut, valeur_référence, 1,
      {(instance, détenteur, *, personne13),
       (concept, attribut, *, taille)}),
   (rt, début, *, rt38),
   (rt, fin, *, rt39)
}
```

3.5.6. Les descriptions relatives aux termes de la forme prédicat-arguments

Elles ont la forme générale suivante :

Forme générale :

{(TYPE_ENTITE0, **prédicat**, *, REF_ENTITE0),
(TYPE_ENTITE1, **argument**, 1, REF_ENTITE1),
(TYPE_ENTITE1, **argument**, 2, REF_ENTITE2),

(TYPE_ENTITE1, **argument**, n, REF_ENTITEn),
(vdv , *, *, V_OU_F),
}

Sémantique associée :

« Il est V_OU_F que REF_ENTITE0(REF_ENTITE1, REF_ENTITE2,.., REF_ENTITEn). »

Les descriptions de cette forme sont en fait des généralisations des formes précédentes.

Dans les descriptions relatives à l'organisation des concepts, rien n'empêche d'écrire

```
{     (concept, argument, 1, mammifère),
      (relation, prédicat, *, est-un),
      (concept, argument, 2, animal) ,
      (vdv , *, *, vrai)
}
```

au lieu de

```
{     (concept, subsumé, *, mammifère),
      (relation, prédicat, *, est-un),
      (concept, subsumeur, *, animal) ,
      (vdv , *, *, vrai)
}
```

Néanmoins, il y a peu d'intérêt à utiliser la forme basée sur les rôles "argument" dans les descriptions dont nous avons parlé précédemment. En effet, les numéros des rôles "argument" seront les seuls éléments qui permettront de connaître la place que prennent les

entités dans l'expression prédicative équivalente.

De plus, du fait que chaque entité argument tient une place bien précise dans la forme prédicative, il nous faudrait pour écrire l'équivalent de

```
{     (concept, *, *,          oiseau),
      (relation, prédicat, *, disjoint),
      (concept, *, *,          minéral),
      (vdv , *, *,             vrai),
}
```

les deux formes suivantes :

```
{     (concept, argument, 1, oiseau),
      (relation, prédicat, *, disjoint),
      (concept, argument, 2, minéral) ,
      (vdv , *, *, vrai)
}
{     (concept, argument, 1, minéral),
      (relation, prédicat, *, disjoint),
      (concept, argument, 2, oiseau),
      (vdv , *, *, vrai)
}
```

En effet, il nous faut dire que « oiseau est disjoint de minéral » et aussi que « minéral est disjoint de oiseau ». Il est possible d'éviter ces doubles écritures en postulant que le prédicat "disjoint" doit être traité de manière symétrique. Cette symétrie est d'ailleurs implicite à l'intérieur de la première forme proposée, car les rôles et numéros de rôles sont identiques.

3.5.7. Les descriptions relatives à des actions

Dans une action donnée, différents éléments interviennent. Pour distinguer chacun de ces éléments, nous utiliserons des éléments d'identification proposés par la théorie de Fillmore [Fillmore, 1968]. Il est possible d'identifier un ensemble de cas sémantiques mettant en évidence les relations de sens qui existent entre des groupes nominaux et le verbe dans une phrase donnée.

Dans les phrases suivantes qui décrivent un même événement sous différents angles :

- *Jean casse la branche avec une pierre. (1)*
- *La pierre casse la branche. (2)*
- *La branche casse. (3)*

les sujets du verbe casser sont trois participants différents. Chacun de ces trois

participants occupe le même rôle syntaxique dans la phrase (la même fonction de surface) et ont des rôles différents dans la sémantique. Ici, Jean est en effet l'initiateur de l'action de casser, la branche en est l'objet et la pierre en est l'instrument (l'outil).

Il apparaît difficile de définir l'ensemble de tous les cas possibles existant dans une phrase donnée. Nous ne prétendrons pas donner, comme Schank, un ensemble de primitives qui permettraient de représenter le sens de toutes les phrases indépendamment du langage source. Nous évitons volontairement la description d'un ensemble complet (qui d'ailleurs pourrait ne pas être réellement complet, vu la difficulté du travail), nous pouvons limiter à une liste de cas bien précis.

Nous n'utiliserons que les cas suivants pour représenter une action :

- L'**acteur** ou **agent**. C'est le protagoniste de l'action.
- L'**objet** de l'action.

  On peut distinguer deux types d'objets :
  - l'objet . C'est l'élément, l'être ou la chose qui subit l'action faite par l'acteur.
  - le résultat.
- L'**instrument** de l'action, c'est à dire l'élément qui contribue à l'action.

  On peut distinguer 5 types de rôle d'instrument :
  - l'outil,
  - la méthode,
  - l'adjuvant,
  - l'ingrédient
  - et le composant.
- Le **bénéficiaire** de l'action.

  C'est celui à qui profite l'action. L'entité instance de personne liée à "Marie" joue ce rôle dans la DCAS associée à la phrase « Jean donne un livre à Marie ».
- La **manière** dont s'est déroulée l'action.

Une description relative à une action contiendra alors les quadruplets suivants :

- une entité-concept qui jouera le rôle appelé "act". La place de référent de cette entité est occupée par le verbe (à l'infinitif) relatif à l'action décrite. La description de la phrase (1) comportera l'entité-concept "casser".
- une entité jouant le rôle d'acteur de cette action. Le type de l'entité qui joue ce rôle est soit instance, soit iensemble, soit censemble.
- une entité jouant le rôle d'objet de l'action.
- une entité jouant le rôle d'instrument.
- une entité jouant le rôle de bénéficiaire.

- une entité-symbole décrivant la manière dans laquelle s'est réalisée l'action. Elle représente une modalité de l'action.
- une entité-vdv indiquant si oui ou non l'action s'est (ou a été ou sera) réalisée effectivement ou non

Forme générale :

{(concept, act, *, REF_ACT),
(TYPE_ENTITE_1, acteur, *, REF_ENTITE_1),
(TYPE_ENTITE_2, objet, *, REF_ENTITE_2),
(TYPE_ENTITE_3, instrument, *, REF_ENTITE_3),
(TYPE_ENTITE_4, bénéficiaire, *, REF_ENTITE_4),
(symbole, manière, *, REF_ENTITE_5),
(vdv , *, *, V_OU_F),
(rt, début, *, RT1),
(rt, fin, *, RT2)
}

Sémantique associée :

« Il est V_OU_F que l'action REF_ACT s'est réalisée avec pour acteur REF_ENTITE_1, pour objet REF_ENTITE_2, pour instrument REF_ENTITE_3, pour bénéficiaireREF_ENTITE_4 et de la manière REF_ENTITE_5. Il existe de plus une relation spatiale REL_SPATIALE entre l'action décrite et l'entité REF_ENTITE_5. »

Exemples :

A la phrase « Jean fait une omelette avec un œuf. », on associera la description

```
{ (concept, act, *,       faire),
  (instance, acteur, *, jean1),
    (instance, objet, *, omelette1),
    (instance, ingrédient, *, œuf1),
  (vdv , *, *,            vrai),
  (rt, début, *, rt34),
  (rt, fin, *, rt35)
}
```

Pour « Jean donne à Marie un livre », on associera

```
{ (concept, act, *,       donner),
  (instance, acteur, *, jean1),
  (instance, objet, *, livre23),
  (instance, bénéficiaire, *, marie1),
  (vdv , *, *,            vrai),
  (rt, début, *, rt36),
  (rt, fin, *, rt37)
}
```

3.5.8. Les descriptions relatives aux situations/propositions

Il existe un certain nombre de relations entre les situations. Les relations de succession dans le temps ne sont pas les seules relations qui existent. Il en existe d'autres, parmi lesquelles, on peut retrouver celles décrites par Schank.

Deux situations peuvent jouer entre elles les couples de rôles suivant :

- cause/résultat.
- prétexte/résultat
- nécessaire/résultat

Notons qu'une description décrivant l'existence de ce type de relation contiendra toujours le quadruplet (vdv,*,*, vrai). Évidemment, le quadruplet (vdv, *, *, faux) sera donné pour indiquer l'existence de non-relation.

Par exemple, dans la phrase « John fait pousser les plantes », la situation qui joue le rôle de cause est celle décrivant John faisant quelque chose, et la situation résultat décrit la croissance des plantes.

Par extension, il existera le même type de description pour les propositions.

3.5.9. Descriptions décrivant des concepts d'ensembles et s'y référant

Nous avons stipulé que les IENSEMBLEs et CENSEMBLEs permettaient de décrire des connaissances à propos d'ensemble d'entités, et aussi d'introduire des connaissances de type générique. Nous limiterons le type des entités recouvertes par ces deux notions au type instance. Une amélioration ultérieure du système consisterait à envisager le recouvrement d'autres types d'entités, comme par exemple, les repères temporels.

Généralement, un ensemble se décrit par :

- les caractéristiques de son cardinal
- la caractérisation de ces différents éléments
- les différents éléments possibles qu'il peut contenir (lorsqu'il s'agit d'un ensemble réel, bien précis)

- un ordre éventuel qui existe dans ces éléments.

1) Référents de CENSEMBLE

Le référent d'une entité de type CENSEMBLE est décrit par plusieurs quadruplets :

- une entité-instance qui jouera le rôle (mr) de membre représentatif. Le référent de cette entité est un référent variable qui peut être utilisé à l'intérieur des autres quadruplets constituant le censemble.
- une entité nombre qui jouera le rôle de cardinal. Cette entité nombre a un référent variable et elle participe généralement à d'autres relations à l'intérieur des autres quadruplets (entre autres, les quadruplets qui jouent le rôle de condition_respectée).
- une entité symbole jouant le rôle de quantificateur. Les référents possibles sont : "tout"(⇔chaque) ou "existe". Si le référent est "existe", cela signifie qu'une entité qui vérifie les conditions à respecter ne jouera pas nécessairement le rôle indiqué.
- une entité symbole jouant le rôle de type de participation. Cette entité indique la manière dont participe tout élément décrit dans le censemble. Le référent est soit **collectif** (tous les éléments participent tous ensemble à la relation), soit **distributif** (tous les éléments peuvent participer seuls au rôle).
- une ensemble de propositions jouant le rôle condition_respectée (cr). Ces dernières permettent la description de relations existant entre le membre représentatif, le cardinal et d'autres entités.

Forme générale du référent d'un CENSEMBLE :

```
{(instance,mr, *,                    I),   <- mr = membre_représentatif
  (nombre, cardinal, *,              N),
   (symbole, quantificateur, *,      QUANTIF),
  (symbole, type_participation, *,   TPARTICIP),
  (proposition, cr, *,               REF_ENTITE_P_1),  <- cr = condition_respectée
  (proposition, cr, *,               REF_ENTITE_P_2),
          ...
  (proposition, cr, *,               REF_ENTITE_P_N),
}
```

Exemple:

Pour exprimer une connaissance générale comme « Toute voiture est composée de deux portes (au moins) », nous utiliserons la description suivante :

```
{  (censemble, composé, *,
           {   (instance, mr, *, IVOITURE),
               (nombre, cardinal, *, N1),
               (symbole, quantificateur, *, tout)
               (symbole, type_participation, *, distributif)
               (proposition, cr, * ,
                     {     (instance, subsumé, *, IVOITURE),
                           (relation, prédicat,*,est_un),
                           (concept, subsumeur,*, voiture),
                           (vdv , *,*, vrai)})
               (proposition, cr, * ,
                     {     (instance, sujet, *, N1),
                           (relation, prédicat,*,=),
                           (nombre, valeur_référence,*, 1),
                           (vdv , *,*, vrai)})})
   (censemble, composant, *,
           {   (instance, mr, *, IPORTE),
               (nombre, cardinal, *, N2),
               (symbole, quantificateur, *, existe)
               (symbole, type_participation, *, distributif)
               (proposition, cr, * ,
                     {     (instance, subsumé, *, IPORTE),
                           (relation, prédicat,*,est_un),
                           (concept, subsumeur,*, porte),
                           (vdv , *,*, vrai)})
               (proposition, cr, * ,
                     {     (instance, sujet, *, N2),
                           (relation, prédicat,*, >=),
                           (nombre, valeur_référence,*, 2),
                           (vdv , *,*, vrai)
                     })}
   (vdv , *, *, vrai)
}
```

Si le censemble relatif aux portes comportait le quantificateur "tout", la description aurait alors pris le sens erroné suivant : « Toute porte est composante d'une voiture ».

La traduction de la description ci-dessus en logique des prédicats la plus proche pourrait se présenter sous la forme suivante:

$\forall$ x, $\exists$ y1, $\exists$ y2, voiture(x) et porte(y1) et porte (y2) et y1 $\neq$ y2
=> composant-composé(x,y).

L'écriture de la DCAS donnée en exemple paraît lourde par rapport à la forme logique ci-dessus. Néanmoins, cette forme d'apparence complexe permet à partir uniquement du quadruplet (instance, R,*, I) où I est une instance de voiture, de calculer une description où cette entité joue le rôle de composé et où un iensemble de portes (de cardinal supérieur ou égal à 2) y jouera un rôle de composant.

L'implémentation effective de ce même type de déduction en logique des prédicats à partir de la forme donnée plus haut est plus difficile.

2) IENSEMBLE

Les référents d'une entité de type IENSEMBLE sont décrits par des quadruplets semblables à ceux vus pour le CENSEMBLE.

Nous retrouverons :

- une entité-instance qui jouera le rôle (mr) de membre représentatif.
- une entité nombre qui jouera le rôle de cardinal.
- une entité symbole jouant le rôle de type de participation. Les référents possibles sont soit **collectif,** soit **distributif.**
- un ensemble de propositions ayant le rôle condition_respectée (cr).

Il n'y a plus de symbole jouant le rôle de quantificateur.

Il peut exister toutefois des quadruplets dans le référent, dont la forme est :

"(instance, membre, I, INSTANCE_MEMBRE)".

Elles permettent d'indiquer la présence dans cet ensemble d'instances données.Forme générale du référent d'un IENSEMBLE :

```
{(instance,mr, *,                    I),   <- mr = membre_représentatif
 (nombre, cardinal, *,               N),
  (symbole, quantificateur, *,       QUANTIF),
 (symbole , type_participation, *,   TYP_PARTICIP)
 (instance,membre, 1,                INSTANCE_MEMBRE_1),
 (instance,membre, M,                INSTANCE_MEMBRE_M),

 (proposition, cr, *,                REF_ENTITE_P_1),  <- cr = condition_respectée
 (proposition, cr, *,                REF_ENTITE_P_2),
        ...
 (proposition, cr, *,                REF_ENTITE_P_N),
}
```

Exemple 1 :

Le quadruplet associée à "Pierre et Nicolas" est le suivant :

```
(iensemble, ROLE_D__IENSEMBLE, *,
   {  (instance, membre, 1, pierre1),
      (instance, membre, 2, nicolas1),
      (symbole, type_participation, *, TYPE_PARTICIP)
      (proposition, cr, * ,
            {     (instance, sujet, *, N1),
                  (relation, prédicat,*, =),
                  (nombre, valeur_référence,*, 2),
                  (vdv , *,*, vrai)
  })})
```

Exemple 2 :

La description, associée à « La cage cage12 est composée de 2 portes. », est la description suivante :

```
{  (instance, composant, *, cage12)
        (iensemble, composé, *,
          {   (instance, mr, *, IPORTE),
              (nombre, cardinal, *, N2),
              (symbole, type_participation, *, distributif)
              (proposition, cr, * ,
                    {     (instance, subsumé, *, IPORTE),
                          (relation, prédicat,*,est_un),
                          (concept, subsumeur,*, porte),
                          (vdv , *,*, vrai)})
              (proposition, cr, * ,
                    {     (instance, sujet, *, N2),
                          (relation, prédicat,*, =),
                          (nombre, valeur_référence,*, 2),
                          (vdv , *,*, vrai)
                    })}
        (vdv , *, *, vrai)
}
```

Ici, le type de la participation pour un ensemble de 2 portes est distributif, car chacune des portes est dite composante de la porte.

3) Des liens de coréférence dans les CENSEMBLEs

Il apparaît parfois, dans certaines descriptions utilisant deux entités de type censemble, le besoin d'exprimer une condition à respecter de l'un des deux censembles à partir de l'instance membre de l'autre censemble.

Par exemple, dans la phrase « Tout père d'enfant aime son enfant. », le censemble correspondant à "père d'enfant" contient dans son référent une relation entre père et enfant. Hors, l'enfant de ce père apparaît dans le censemble correspondant à "son enfant".

Pour gérer ces liens de coréférence, nous intégrerons dans le censemble dont le membre représentatif est utilisé dans la partie condition respectée d'un autre censemble, un quadruplet de la forme "(instance, **coref_mr**,*, <variable>)".

La description ci-dessous correspond à la phrase « Tout père d'un enfant aime son enfant. »

```
{  (censemble, acteur, *,
           {   (instance, mr, *, IPERE),
               (nombre, cardinal, *, N1),
               (symbole, quantificateur, *, tout)
               (symbole, type_participation, *, distributif)
               (proposition, cr, * ,
                     {     (instance, subsumé, *, IPERE),
                           (relation, prédicat,*,est_un),
                           (concept, subsumeur,*, personne),
                           (vdv , *,*, vrai)})
               (proposition, cr, * ,
                     {     (instance, père, *, IPERE),
                           (instance, fils,*, COREF_IENFANT),
                           (vdv , *,*, vrai)})
               (proposition, cr, * ,
                     {     (instance, sujet, *, N1),
                           (relation, prédicat,*,=),
                           (nombre, valeur_référence,*, 1),
                           (vdv , *,*, vrai)
                     })}),
   (censemble, objet, *,
           {   (instance, mr, *, IENFANT),
               (instance, coref_mr, *, COREF_IENFANT),
               (nombre, cardinal, *, N2),
               (symbole, quantificateur, *, tout)
               (symbole, type_participation, *, distributif)
               (proposition, cr, * ,
                     {     (instance, subsumé, *, IENFANT),
                           (relation, prédicat,*,est_un),
                           (concept, subsumeur,*, personne),
                           (vdv , *,*, vrai)})
               (proposition, cr, * ,
                     {     (instance, sujet, *, N2),
                           (relation, prédicat,*, =),
                           (nombre, valeur_référence,*, 1),
                           (vdv , *,*, vrai)
                     })}),
   (concept,act,*,aimer),
   (vdv , *, *, vrai),
   (rt,début, *, rth1),
   (rt, fin,*,rth2),
}
```

4. Introduction de descriptions dans la base

4.1. Syntaxe utilisée pour l'introduction d'informations

L'entrée d'une information dans la base de connaissances se fait par l'intermédiaire de la commande tell.

La syntaxe de cette commande est la suivante :

tell (<Description>, X).

La variable X contient un numéro de référence que le système a associé à la description introduite. Ce numéro de référence est unique. Il permet une identification plus aisée des descriptions utilisées pour un raisonnement produit ; le système est en effet capable de donner la trace de son raisonnement.

Notre système de représentation étant implémenté en Prolog, la description s'écrira à l'aide de la syntaxe suivante :

<Description> ::= **[** <Quadruplet> { <Liste de quadruplets> } **]**
<Liste de quadruplets> ::= <Quadruplet> { <Liste de quadruplets> }
<Quadruplet> ::= **[** <Type entité> , <Rôle entité> ,
<Numéro Rôle Entité> , <Référent Entité> **]**
<Type entité> ::= <constante alphanumérique> | <variable>
<Rôle entité> ::= <constante alphanumérique> | <variable> | *****
<Numéro Rôle Entité> ::= <constante alphanumérique> | <variable> | *****
<Référent Entité> ::= <constante alphanumérique> | <variable>
<constante> ::= <constante chaîne> | < constante nombre> |
< constante alphanumérique>
< variable> ::= { <lettre majuscule> } { < suite de lettres ou chiffres > } |
<souligné**>**{ < suite de lettres ou chiffres > }
< suite de lettres ou chiffres > ::= <lettre> { < suite de lettres ou chiffres> } |
<chiffre> { < suite de lettres > }
<constante chaîne> ::= **"**< suite de lettres >**"**
< constante nombre> ::= <chiffre> { < suite de chiffres > }
< constante alphanumérique> ::= < suite de lettres ou chiffres >
< suite de chiffres > ::= **0** | **1** | .. | **9**
<lettre> ::= **a** | .. | **z** | **A** | .. | **Z**
<lettre majuscule> ::= **A** | **B** | ... | **Z**
<souligné**> ::=** _

Exemple 1:

L'instruction ci-dessous permet d'introduire la connaissance « Un chat est à côté d'une chaise. ».

```
:- tell(
 [ [iensemble, cible, *,
       [[instance, mr, *, ICHAT],
        [nombre, cardinal, *, N1],
        [symbole, type_participation, *, distributif],
        [proposition, cr, * ,
             [[instance,subsumé,*,ICHAT],[relation,prédicat,*,est_un],
              [concept, subsumeur,*, chat],[vdv , *,*, vrai]]],
        [proposition, cr, * ,
             [[instance, sujet, *, N1],[relation, prédicat,*, =],
              [nombre, valeur_référence,*, 1],[vdv , *,*, vrai]]]]]
   [relation,*,*,à_côté_de],
   [iensemble, site, *,
       [[instance, mr, *, ICHAISE],
        [nombre, cardinal, *, N2],
        [symbole, type_participation, *, distributif]
        [proposition, cr, * ,
             [[instance, subsumé, *, ICHAISE],
              [relation, prédicat,*,est_un],
              [concept, subsumeur,*, chaise],
              [vdv , *,*, vrai]]]
        [proposition, cr, * ,
             [[instance, sujet, *, N2],
              [relation, prédicat,*, =],
              [nombre, valeur_référence,*, 1],
              [vdv , *,*, vrai]]]]]
   [rt,début,*,rt1],[rt,fin,*,rt2],
   [vdv, *, *, vrai], [*,*,*,*]
 ] , X) .
```

=> X = 35.

Le quadruplet [*,*,*,*] indique au système que la description introduite est une description assertée. La valeur de X retournée est un numéro unique attribué par le système à cette description. Ce numéro pourra être utilisé dans l'analyse du chemin construit pendant un raisonnement. En effet, le chemin se présente comme une série structurée de nombres correspondant à des numéros de descriptions utilisées dans le raisonnement.

4.2. Processus de vérification de validité d'une description

Lors de la phase d'introduction de nouvelles descriptions dans la base, des processus internes de vérification sont enclenchés.

Soient

DI, la description à introduire et

EQDIsv, l'ensemble de quadruplets de DI amputé du quadruplet possible

(vdv, *, *, VI).

1 - Si DI contient un quadruplet (vdv,*, * VI) , le système vérifie la non présence d'une

description DJ telle que :

$DJ = EQDJ_{sv} + \{(vdv, *, *, VJ)\}$

et $VI \neq VJ$

Si cette description DJ existe, le système signale l'existence de cette description DJ et refuse d'introduire DI.

2 - S'il existe une description DG telle que

$DG = EQDG_{sv} + \{(vdv, *, *, VG)\}$

et $EQDG_{sv} = EQDIsv$

et $VG \neq VI$

et DG est déductible de la base

alors le système indique l'existence d'un problème éventuel avec la description DI et continue le processus de vérification.

3 - Si la description DI est une description dont le référent est défini,

et s'il existe une autre description DJ de même type et de même référent,

alors le système demande l'introduction de la jointure[26] de ces deux descriptions (tout en demandant au système de retirer momentanément de la base la description DJ). Si la jointure est acceptée, le système demande le retrait définitif de la base de DJ. Si la jointure n'est pas acceptée, le système signale la non-introduction de DI et de sa jointure. Si la jointure est acceptée dans la base, le système signale que DI est tenu en compte par l'introduction de sa jointure. Le système, quel que soit le résultat de la recherche, n'introduira jamais DI en tant que tel.

26

Lorsque 2 descriptions donnent des informations à propos de la même situation, ou de la même proposition, il est possible de regrouper les informations contenues dans ces 2 descriptions pour n'en former qu'une.

Par exemple, il est possible d'écrire :2 descriptions correspondant à :

- Georges mange un steak.
- Georges le mange avec un couteau.

La jointure se traduirait en l'expression suivante : "Georges mange un steak avec un couteau."

4 - Les quadruplets admis pour les valeurs de vérités sont :

- (vdv, *, *, vrai)
- (vdv, *, *, faux)
- (vdv, *, *, inconnu)
- (vdv, *, *, X)

5 - Pour des raison d'uniformisation, il ne doit y avoir

- qu'un quadruplet au plus de type vdv
- qu'un quadruplet au plus de type concept qui joue le rôle act
- qu'un quadruplet au plus qui joue le rôle objet
- qu'un quadruplet au plus qui joue le rôle acteur (s'il y a plusieurs instances jouant ce rôle dans un événement, ces instances sont regroupées dans un iensemble)
- qu'un quadruplet au plus de type symbole jouant le rôle de référent-description
- qu'un quadruplet au plus de type symbole jouant le rôle de type-description.

6 - Si la description contient des entités de type complexe, le système vérifie leur conformité. Par exemple, une entité de type iattribut ne possède que 2 quadruplets dans son référent. De plus, ces quadruplets font intervenir certaines entités avec des rôles bien précisés.

Si la vérification franchit toutes ces étapes avec succès, alors la description est introduite dans la base.

5. Recherche d'une description dans la base

5.1. Lancement de requêtes

La recherche d'une information dans la base de connaissances se fait par l'intermédiaire de la commande ask.

La syntaxe de cette commande est la suivante :

ASK (<variable de stockage de la Description Trouvée>, <Chemin>
<Description-référence>)

La description référence peut être une description incomplète, c'est à dire que tous les quadruplets ne sont pas tous donnés. Le système reconnaît l'incomplétude des informations grâce à la présence d'une variable dans la liste des quadruplets de la description référence.

Cette façon de rechercher les informations en donnant des informations partielles sur le type de données recherchées rend l'exploitation des données plus souple.

Le système retournera dans <Chemin> la liste des descriptions utilisées pour trouver la description trouvée.

5.1.1. Une façon simple d'exprimer l'information

Soit la base de connaissances initiale initialisée par les instructions suivantes :

```
/* Jean fait une omelette avec un œuf  - description n°1*/
:-tell([[instance,acteur,*,jean1],
        [concept,act,*,faire],
        [instance,objet,*,omelette1],
        [instance, ingrédient, *, oeuf1],
        [vdv,*,*,vrai],[rt,début,*,rt1],[rt,fin,*,rt2]], X).
=> X = desc-1
/* Jean chante - description n°2*/
:-tell([[instance,acteur,*,jean1], [concept,act,*,chanter],
        [vdv,*,*,vrai],[rt,début,*,rt1],[rt,fin,*,rt2]], X).
=> X = desc-2
```

Pour demander au système ce que fait Jean, il suffit d'écrire la requête suivante

ask(DT, Chemin, [[instance, acteur, *, jean1], [concept, act, *, ACTION], X].

Parmi les réponses fournies par le système, on obtiendra au moins les deux réponses suivantes :

1ère réponse :

```
DT = [[instance,acteur,*,jean1],
      [concept,act,*,faire],[instance,objet,*,omelette1],
      [instance, ingrédient, *, oeuf1],
      [vdv,*,*,vrai],[rt,début,*,rt1],[rt,fin,*,rt2]]
Chemin = desc-1
ACTION = faire
X  =  [[instance,objet,*,omelette1],
      [instance, ingrédient, *, oeuf1]
      [vdv,*,*,vrai],[rt,début,*,rt1],[rt,fin,*,rt2]]
```

2ème réponse :

```
DT = [[instance,acteur,*,jean1], [concept,act,*,chanter],
      [vdv,*,*,vrai],[rt,début,*,rt1],[rt,fin,*,rt2]]
Chemin = desc-2
ACTION = chanter
X  = [[vdv,*,*,vrai],[rt,début,*,rt1],[rt,fin,*,rt2]]
```

On remarquera ici que les formes trouvées (par la même requête) peuvent posséder chacun un nombre de quadruplets différent.

5.1.2. Composition de plusieurs requêtes

Il est possible de composer plusieurs requêtes à la suite.

Cette possibilité permet par exemple de savoir ce que fait la personne qui est dans la maison. La traduction de cette requête donnera :

ask(DT1, Chemin1, [[instance, dénoté, *, P], [relation, prédicat, *, est_un],
concept, dénoteur,*, personne], [vdv,*,*, vrai]])
ask(DT2, Chemin2, [[instance, cible,*, P], [relation, prédicat, *, dans],
[instance, *, *, maison1], [vdv, *, *, vrai],
[rt, début, *, rt1], [rt, début, *, rt2]]),
ask(DT3, Chemin3, [[instance, acteur,*, P], [rt, début, *, rt1], [rt, début, *, rt2]], X).

La description DT3 contient l'information recherchée. De plus, le système indique dans P, le référent de la personne recherchée.

Cette décomposition en structures plus simples n'est pas nécessaire.

En effet, le même résultat est obtenu à partir de la requête ci-dessous:

```
ask(DT4, Chemin4,
   [[iensemble, acteur, *,
         [     [instance, mr, *, IPERSONNE),
               [nombre, cardinal, *, N1),
               [symbole, quantificateur, *, tout)
               [symbole, type_participation, *, distributif)
               [proposition, cr, * ,
                     [     [instance, subsumé, *, IPERSONNE],
                           [relation, prédicat,*,est_un],
                           [concept, subsumeur,*, personne],
                           [vdv , *,*, vrai]]],
               [proposition, cr, * ,
                     [     [instance, cible, *, IPERSONNE],
                           [relation, prédicat,*,dans],
                           [instance, site,*, maison1],
                           [vdv , *,*, vrai]]],
               [proposition, cr, * ,
                     [     [instance, sujet, *, N1],
                           [relation, prédicat,*,=],
                           [nombre, valeur_référence,*, 1],
                           [vdv , *,*, vrai]]]
    ]],
   [rt, début, *, rt1], [rt, début, *, rt2]], X).
```

5.2. Les inférences déclenchées par une requête

Lorsque le système reçoit une requête avec une description référence DR à rechercher, il parcourt toutes les descriptions de la base pour rechercher une description correspondante.

Chaque description balayée DB est comparée à la description référence DR.

Pour que la description DB soit retournée comme résultat de la requête, le système doit établir le fait que la description DR est dite **associable** (cette notion d'associabilité est définie dans les paragraphes qui suivront) à la description DB. Ce fait n'est établi en comparant chaque quadruplet de DR à chaque quadruplet de DB. En général, lorsque chaque quadruplet de DR est dit associable à chaque quadruplet de DB, la description DR est alors dite associable à DB.

Cette notion d'associabilité peut être rapprochée de l'unification en Prolog. En Prolog, lorsque l'unification entre deux entités (liste, atome, variable) réussit, elle aboutit à des entités de même type. Dans un système à base de DCAS, l'associabilité permet l'"association" de deux entités de type différents.

5.2.1. Notion de quadruplet et de uplet associables

Soit QR, un quadruplet de référence.

QR = (TYPE_ER, ROLE_ER, NR_ER, REFERENT_ER)

Soit QB un quadruplet d'une description balayée.

QB= (TYPE_EB, ROLE_EB, NR_EB, REFERENT_EB).

1) Notion de uplet associable

Le terme **constante** est utilisé au sens général. Une constante peut être aussi bien de type atomique (cas du référent des entités instances), que de type description (cas général des référents des entités de type composés).

Un uplet U1 est associable à un uplet U2 s'il se trouve dans l'une des conditions suivantes :

- le uplet U1 est une constante et le uplet U2 est une variable
- le uplet U1 est une variable et le uplet U2 est une variable
- le uplet U1 est une variable et le uplet U2 est une constante (ou une description)
- les uplet U1 et U2 sont deux constantes identiques

Note :

Pour un quadruplet de type complexe (par exemple, proposition), le uplet se trouvant dans le champ référent peut avoir trois formes :

- constante symbolique

Exemple : (proposition, R, NR, p1)

- variable

Exemple : (proposition, R, NR, P)

- description

Exemple : (proposition, R, NR, {(instance, acteur, *, IACT), X })

La constante symbolique p1 est développable en une DCAS.

Nous dirons dans ce cas qu'un uplet U1 situé dans la partie référent est associable à un uplet U2 dans la partie référent si leurs formes développées (c-a-d leur DCAS correspondante) sont associables. Les définitions d'associabilité entre DCAS sont données dans les paragraphes qui suivent.

2) Associabilité de quadruplet

Il existe plusieurs conditions suffisantes d'associabilités de QR à QB.

CONDITION 1 : QR et QB de même type

QR est dit associable à QB si :

- TYPE_ER est associable à TYPE_EB
- ROLE_ER est associable à ROLE_EB
- NR_ER est associable à NR_EB

Ex :

Le quadruplet (instance, X, *, i1) est associable au quadruplet (I, acteur, *, i1).

Le quadruplet (proposition, RB1,NRB1 { (concept, act, *, A), X}) est associable au quadruplet (proposition, RB2,NRB2, { (concept, act, *, manger), (instance, acteur, *, nicolas), (instance, objet, *, pomme125)}).

CONDITION 2 : QR est de type **instance**, QB de type **iensemble** avec une participation de type distributif de tout élément décrit dans cet iensemble

QR est dit associable à QB si

- ROLE_ER est associable à ROLE_EB
- NR_ER est associable à NR_EB
- (REFERENT_EB est donc constant)
- l'entité de type instance auquel se réfère QR appartient à l'iensemble donné dans QB

Cas 1 :

Lorsque REFERENT_ER est constant (il s'agit d'une instance précise), le système est capable de déduire qu'une instance appartient à un iensemble donné dans deux sous-cas possibles :

1 - l'iensemble comprend le quadruplet (symbole, quantificateur, *, tout).

Si l'entité instance vérifie toutes les conditions à respecter explicitées dans la structure du iensemble, alors cette instance appartient à ce iensemble.

Exemple :

Le quadruplet (instance, R, NR, chat12) correspondant à un chat donné est associable au quadruplet (iensemble, R2, NR2, iex) avec iex, le référent du iensemble associé à "tous les chats" dans l'expression « tous les chats sont sur la Terre. »

2 - l'iensemble comprend explicitement l'instance
(instance, membre, *, REFERENT_ER).

Exemple :

Le quadruplet (instance, R,NR,i1) est associable au quadruplet
(iensemble, R2,NR2,
{ (instance, mr,*, IMR),
(symbole, type_participation, *,distributif),
(instance, membre, , cr, *, i1),
..... })

Cas 2 :

Lorsque REFERENT_ER est variable, le système proposera pour cette variable les différents référents de toutes les instances connues de l'iensemble REFERENT_EB.

Si cet iensemble est par exemple l'ensemble de toutes les instances d'homme, le système, connaissant Gérard et Jean comme des instances d'homme, proposera ces derniers comme valeurs pour REFERENT_ER.

SI cet iensemble ne comporte pas le quadruplet (symbole, quantificateur, *, tout), seules les instances explicitées par les quadruplets de la forme (instance, membre, *, I) dans le iensemble seront proposées comme valeurs possibles pour REFERENT_ER.

CONDITION 3 : QR est de type **instance**,
QB de type **censemble** avec pour tout élément décrit dans ce censemble, une participation de type distributif

QR est dit associable à QB si
- ROLE_ER est associable à ROLE_EB
- NR_ER est associable à NR_EB
- (REFERENT_EB est donc constant)
- l'entité de type instance auquel se réfère QR appartient au censemble donné dans QB

Cas 1 :

Lorsque REFERENT_ER est constant, le système est capable de déduire qu'une instance appartient à un censemble en vérifiant que toutes les conditions explicitées dans la structure du censemble sont bien respectées.

Exemple : Prenons la phrase « toute bibliothèque est composée d'au moins 2 étagères » . La description correspondante comportera deux censembles.

Le quadruplet (instance, R, NR, ix1) est associable au censemble utilisé pour décrire "toute bibliothèque" que si ix1 est une instance du concept bibliothèque.

Cas 2 :

Lorsque REFERENT_ER est variable, le système proposera pour cette variable les différents référents de toutes les instances **connues** du censemble REFERENT_EB.

CONDITION 4 : QR est de type **iensemble**, QB de type **instance**

QR est dit associable à QB si
- ROLE_ER est associable à ROLE_EB
- NR_ER est associable à NR_EB
- REFERENT_EB est constant
- l'entité de type iensemble à laquelle se réfère QR est un singleton et l'instance désignée par QB remplit toutes les conditions explicitées dans QR.

Exemple :

Soit chat1 une instance donnée d'un chat.

Le quadruplet

```
(iensemble, R2,NR2,
              {     (instance, mr,*,  IMR),
                    (symbole, type_participation, *,distributif),
                    (nombre, cardinal, *, N),
                    (proposition, cr, *,
                          {     (instance,dénoté, *, IMR),
                                (concept,dénoteur,*, chat),
                                (relation, prédicat,*, est_un),
                                (vdv, *,*, vrai)
                          }),
                     (proposition, cr, *,
                          {     (nombre,sujet, *, N),
                                (nombre,valeur_référence,*, 1),
                                (relation, prédicat,*, =),
                                (vdv, *,*, vrai)
                          }),
              })
```

correspondant à l'expression

"un chat" dans « un chat court dans la rue », par exemple,

est associable à l'instance (instance, R,N, chat1).

Nota Bene

Lorsque REFERENT_EB est une variable, le système peut déclarer que l'iensemble ci-dessus est associable à (instance, R, N, REFERENT_EB) et REFERENT_EB ne peut prendre que des valeurs de référence d'éléments instances vérifiant les conditions à respecter de l'iensemble.

CONDITION 5 : QR et QB sont de type **iensemble** avec une participation distributive des éléments instances qu'ils décrivent et ne contiennent pas de quadruplet de la forme (symbole, quantificateur, *, tout).

QR est dit associable à QB si

- ROLE_ER est associable à ROLE_EB
- NR_ER est associable à NR_EB
- REFERENT_ER et REFERENT_EB sont constants
- l'entité de type iensemble à laquelle se réfère QR est plus général que l'entité de type iensemble désignée par QB.

CAS 1.

Dans le cas où aucun élément de REFERENT_ER n'est explicitement donné[27], pour

[27] Il n'y a donc pas de quadruplet de la forme (instance, membre, NR, IX).

que la dernière proposition énoncée plus haut soit vérifiée, il faut que les propriétés suivantes soient démontrées :

- le cardinal de REFERENT_ER est inférieur ou égal à REFERENT_EB
- le membre représentatif de REFERENT_EB vérifie toutes les conditions respectées par le membre représentatif de REFERENT_ER

Exemple :

Le référent de l'iensemble correspondant à "3 hommes" est associable au référent de l'iensemble correspondant à "4 hommes" dans l'expression « 4 hommes sont dans la maison. » .

Le référent de l'iensemble correspondant à "3 animaux" est associable au référent de l'iensemble correspondant à "4 chats" dans l'expression « 4 chats sont dans le jardin. » .

CAS 2.

Dans le cas où un (ou plusieurs) élement(s) de REFERENT_ER est (sont) explicitement donné(s), il faut que :

- ce(s) dernier(s) soit (soient) aussi explicitement donné(s) dans REFERENT_EB.
- le cardinal de REFERENT_ER est inférieur ou égal à REFERENT_EB
- le membre représentatif de REFERENT_EB vérifie toutes les conditions respectées par le membre représentatif de REFERENT_ER

Exemple :

Le référent de l'iensemble correspondant à "3 animaux dont le chat de Jacques" est associable au référent de l'iensemble correspondant à "4 chats dont celui de Jacques".

CONDITION 5B : QR et QB sont de type **iensemble** avec une participation distributive des éléments instances qu'ils décrivent et REFERENT_EB de QB contient un quadruplet de la forme (symbole, quantificateur, *, tout).

QR est dit associable à QB si :

- ROLE_ER est associable à ROLE_EB
- NR_ER est associable à NR_EB
- REFERENT_ER et REFERENT_EB sont constants
- l'entité de type iensemble à laquelle se réfère QR est plus général que l'entité de type iensemble désignée par QB.

CAS 1.

Dans le cas où aucun élément de REFERENT_ER n'est explicitement donné, il faut que le membre représentatif de REFERENT_EB vérifie toutes les conditions respectées par le

membre représentatif de REFERENT_ER

Exemple :

Le référent de l'iensemble correspondant à "3 hommes" est associable au référent de l'iensemble correspondant à "tous les hommes" dans l'expression « tous les hommes sont dans la maison. ».

Le référent de l'iensemble correspondant à "3 animaux" est associable au référent de l'iensemble correspondant à "tous les chats" dans l'expression « tous les chats sont dans le jardin. ».

CAS 2.

Dans le cas où un (ou plusieurs) élement(s) de REFERENT_ER est (sont) explicitement donné(s), il faut que :

- ce(s) dernier(s) vérifient les conditions données dans REFERENT_EB.
- le membre représentatif de REFERENT_EB vérifie toutes les conditions respectées par le membre représentatif de REFERENT_ER alors

Exemple :

Le référent de l'iensemble correspondant à "3 animaux dont le chat de Jacques" est associable au référent de l'iensemble correspondant à "tous les chats" dans « tous les chats sont dans la maison. ».

CONDITION 5C : QR et QB sont de type **iensemble** avec une participation distributive des éléments instances qu'ils décrivent et REFERENT_EB de QB contient un quadruplet de la forme (symbole, quantificateur, *, tout).

QR est dit associable à QB si

- ROLE_ER est associable à ROLE_EB
- NR_ER est associable à NR_EB
- REFERENT_ER et REFERENT_EB sont constants
- l'entité de type iensemble à laquelle se réfère QR est plus spécifique que l'entité de type iensemble désignée par QB.

Cas où aucun élément de REFERENT_ER n'est explicitement donné

Il faut montrer que le membre représentatif de REFERENT_ER vérifie toutes les conditions respectées par le membre représentatif de REFERENT_ERB

Exemple :

Le référent de l'iensemble correspondant à "3 persans" est associable au référent de l'iensemble correspondant à "tous les chats" dans l'expression « tous les chats sont dans la

maison. ».

Cas 2.

Dans le cas où un (ou plusieurs) élement(s) de REFERENT_ER est (sont) explicitement donné(s), il faut que :

- ce(s) dernier(s) vérifient les conditions données dans REFERENT_EB.
- le membre représentatif de REFERENT_ER vérifie toutes les conditions respectées par le membre représentatif de REFERENT_ER alors

Exemple :

Le référent de l'iensemble correspondant à "3 persans dont celui de Jacques" est associable au référent de l'iensemble correspondant à "tous les chats" dans l'expression « tous les chats sont dans la maison. ».

CONDITION 6: QR est de type **iensemble**, QB de type **censemble** avec une participation distributive des éléments instances qu'ils décrivent .

QR est dit associable à QB si

- ROLE_ER est associable à ROLE_EB
- NR_ER est associable à NR_EB
- REFERENT_ER et REFERENT_EB sont constants
- l'entité de type iensemble auquel se réfère QR est un iensemble décrit par le censemble auquel se réfère QB.

Tous les éléments instances décrits par leurs propriétés ou mentionnés explicitement doivent vérifier les conditions énoncées dans QB.

Exemple :

L'iensemble correspondant à "instances de tiroirs ⊕ distributif ⊕ nombre = 3" est associable au censemble correspondant à "instances de tiroirs ⊕ distributif ⊕ nombre >= 2 "

Ce même iensemble n'est pas associable au censemble correspondant à "instances de tiroirs ⊕ distributif ⊕ nombre <= 2 "

Par contre, un problème peut se poser par rapport au censemble correspondant à "instances de tiroirs ⊕ distributif ⊕ nombre >= 4 ⊕ nombre <=10" En effet, imaginons que nous représentions la phrase « Une commode a pour composantes 4 tiroirs au moins ». Sa représentation implique un censemble pour décrire les tiroirs et dans les conditions à respecter, figurent donc "nombre >= 4". Lorsque l'on veut demander si une instance de commode donnée est composant d'un ensemble de 2 tiroirs, la réponse donnée serait à première vue "non". Mais si l'on examine attentivement la question, elle pourrait être énoncée

en langage naturel par « Peut-on trouver 2 tiroirs qui sont composante d'une commode ? ». Dans ce cas, la réponse serait oui.

Par contre, à la question « Peut-on trouver 11 tiroirs composantes d'une même commode ? » la réponse est évidemment non.

5.2.2. Notion de descriptions associables

Nous avons établi l'associabilité de deux quadruplets.

Nous allons voir maintenant comment deux descriptions données, l'une que nous appellerons DR (description référence) et l'autre DB (description balayée) peuvent être associables.

Dans la partie « Lancement d'une requête » nous avons introduit l'utilisation d'une variable X, à l'intérieur de l'ensemble de quadruplets de la description référence. Cette variable nous permet de décrire dans le cadre d'une utilisation ultérieure, des quadruplets qui n'auraient pas été cités dans la description référence.

Cette variable X, en cas de recherche avec succès, aura alors été unifiée avec le reste de la description référence.

Lorsque cette variable X n'est pas utilisée dans la description référence, nous supposerons que la description référence est alors **complète** et qu'elle ne contient pas d'autres quadruplets supplémentaires. Dans le cas contraire, elle sera dite incomplète.

Nous obtenons 4 configurations d'associabilités correspondant aux combinaisons possibles de complétude/incomplétude des description référence et balayée.

Pour chaque configuration, on définira deux résultats d'association :

- le résultat de l'association sur la description référence
- le résultat de l'association sur la description balayée

Définition :

Pour une description DD se présentant sous la forme {(XD,YD,ZD,TD), X }, le quadruplet (XD,YD,ZD,TD) sera dit "**quadruplet explicite**", tandis que ceux présents dans X seront dits **potentiels**.

1) Description référence complète et description balayée complète

DB = { (XB1,YB1,ZB1,TB1),(XB2,YB2,ZB2,TB2), (XB3,YB3,ZB3,TB3)}
DR = { (XR1,YR1,ZR1,TR1),(XR2,YR12,ZR2,TR2), (XR3,YR3,ZR3,TR3)}

Dans ce contexte, la description DR est associable à la description DB si :

- tous les quadruplets QR de DR sont associables à un quadruplet explicite QB de DB
- aucun des quadruplets de QB associés ne l'est plus d'une fois. Cela implique le même nombre de quadruplets de part et d'autre.

Par la suite, nous noterons :

- DRA, la liste des quadruplets explicites de DR qui sont associables à des quadruplets de DB.
- DBA, la liste des quadruplets explicites de DB auxquels tous les éléments de DRA sont associables un à un .

Dans ce contexte, DR est donc associable à DB si DR =DRA et DB = DBA.

Exemple :

La description référence

{(instance, cible, *, chat1), (instance, site,*,ISITE),
(vdv,*,*, vrai), (relation,prédicat,*,dans)}

est associable à la description

{(iensemble, cible, *, { (instance, membre, *, chat1), ...}),
(instance, site,*,maison1),
(vdv,*,*, vrai), (relation,prédicat,*,dans)}

2) Description référence incomplète et description balayée complète

DB = { (XB1,YB1,ZB1,TB1),(XB2,YB2,ZB2,TB2), (XB3,YB3,ZB3,TB3)}
DR = { (XR1,YR1,ZR1,TR1),(XR2,YR12,ZR2,TR2), VR}

Dans ce contexte, la description DR est associable à la description DB si chaque quadruplet explicite QR de DR est associable à un quadruplet explicite QB de DB et aucun des quadruplet QB associé ne l'est plus d'une fois.

Après association de DR à DB, la variable VR est égale à la réunion de :

- l'ensemble de tous les quadruplets (de type différent de censemble) de QB auquel ne correspond aucun quadruplet
- l'ensemble de tous les quadruplets obtenus par dérivation de l'ensemble de tous les quadruplets de type censemble de QB auquel ne correspond aucun quadruplet

Dans ce contexte, DR est associable à DB si :

DR =DRA ⊕ VR et DB = DBA.⊕ DBNA

(DBNA est l'ensemble de quadruplets qui n'ont pas été associés aux quadruplets de DRA).

Après association, la variable VR sera unifiée à DBNA'.

DBNA' est obtenu par dérivation de l'ensemble DBNA.

Le résultat de l'association sur DR donnera une description DR complète.

Note :

Le nombre de quadruplets explicites de DB est supérieur ou égal au nombre de quadruplets de DR.

Exemple 1:

La description référence

{(instance, cible, *, chat1), (instance, site,*,ISITE), VR}

est associable à la description

{(iensemble, cible, *, { (instance, membre, *, chat1), ...}),
(instance, site,*,maison1),
(vdv,*,*, vrai), (relation,prédicat,*,dans)}

Le résultat de l'association sur DR donne {(instance, cible, *, chat1), (instance, site,*,ISITE), {(vdv,*,*, vrai), (relation,prédicat,*,dans)}}.

Exemple 2:

Soient :

- DB, la description balayée associée à la phrase « Tout père d'un enfant aime son enfant. » :

```
{  (censemble, acteur, *,
          {   (instance, mr, *, IPERE),
              (nombre, cardinal, *, N1),
              (symbole, quantificateur, *, tout)
              (symbole, type_participation, *, distributif)
              (proposition, cr, * ,
                    {     (instance, subsumé, *, IPERE),
                          (relation, prédicat,*,est_un),
                          (concept, subsumeur,*, personne),
                          (vdv , *,*, vrai)})
              (proposition, cr, * ,
                    {     (instance, père, *, IPERE),
                          (instance, fils,*, COREF_IENFANT),
                          (vdv , *,*, vrai)})
              (proposition, cr, * ,
                    {     (instance, sujet, *, N),
                          (relation, prédicat,*,=),
                          (nombre, valeur_référence,*, 1),
                          (vdv , *,*, vrai)
                    })}),
   (censemble, objet, *,
          {   (instance, mr, *, IENFANT),
              (instance, coref_mr, *, COREF_IENFANT),
              (nombre, cardinal, *, N2),
              (symbole, quantificateur, *, tout)
              (symbole, type_participation, *, distributif)
              (proposition, cr, * ,
                    {     (instance, subsumé, *, IENFANT),
                          (relation, prédicat,*,est_un),
                          (concept, subsumeur,*, personne),
                          (vdv , *,*, vrai)})
              (proposition, cr, * ,
                    {     (instance, sujet, *, N2),
                          (relation, prédicat,*, =),
                          (nombre, valeur_référence,*, 1),
                          (vdv , *,*, vrai)
                    })}),
   (concept,act,*,aimer),
   (vdv , *, *, vrai),
   (rt,début, *, rth1),
   (rt, fin,*,rth2),
}
```

- DR = { (instance,acteur, *, adam), (concept,act,*,aimer), (vdv,*,*, vrai), X }

- D1, une description assertée dans la base de connaissances indiquant que Adam est le père de Caïn

D1 = {(instance, père, *, adam), (instance, enfant,*, cain), (vdv,*,*, vrai)}

Le quadruplet (instance,acteur, *, adam) de DR est associable au censemble de même rôle dans DB. Le résultat de l'association sur DR donne

{ (instance,acteur, *, adam), (concept,act,*,aimer),
(instance, enfant,*, cain), (vdv,*,*, vrai),
(rt,début, *, rth1),(rt, fin,*,rth2)}

3) Description référence complète et description balayée incomplète

DB = { (XB1,YB1,ZB1,TB1),(XB2,YB2,ZB2,TB2), VB}
DR = { (XR1,YR1,ZR1,TR1),(XR2,YR12,ZR2,TR2), (XR3,YR3,ZR3,TR3)}

Dans ce contexte, la description DR est associable à la description DB si chaque quadruplet explicite QR de DR est associable à un quadruplet explicite QB de DB et aucun des quadruplet QB associé ne l'est plus d'une fois.

Après association de DR à DB, la variable VB est égale à l'ensemble de tous les quadruplets de QR auquel ne correspond aucun quadruplet dans QB.

Le résultat de l'association sur DB donnera une description DR complète.

Note :

Le nombre de quadruplets explicites de DR est supérieur ou égal au nombre de quadruplets de DR.

Exemple :

La description référence
{(instance, cible, *, chat1), (instance, site,*,ISITE),
(relation, prédicat,*,dans), (vdv,*,*,vrai)}
est associable à la description
{(iensemble, cible, *, { (instance, membre, *, chat1), ...}),
(instance, site,*,maison1),
VB}
La variable VB prendra la valeur
{(relation, prédicat,*,dans),(vdv,*,*,vrai)}
Le résultat de l'association sur DB donne
{(iensemble, cible, *,
{ (instance, membre, *, chat1), ...}),
(instance,site,*,maison1),
{(relation, prédicat,*,dans),(vdv,*,*,vrai)}
}.

4) Description référence incomplète et description balayée incomplète

DB = { (XB1,YB1,ZB1,TB1),(XB2,YB2,ZB2,TB2), (XB3,YB3,ZB3,TB3),VB}
DR = { (XR1,YR1,ZR1,TR1),(XR2,YR12,ZR2,TR2), (XR3,YR3,ZR3,TR3),VR}

Soient DRA et DBA, la liste des quadruplets associables 2 à 2.
Soient DRNA et DBNA, la liste des quadruplets non associables.

DB = DBA ⊕ DBNA ⊕ VB1 ⊕ VCOMMUN
DR = DRA ⊕ DRNA ⊕ VR1 ⊕ VCOMMUN

Dans ce contexte, la description DR est toujours associable à la description DB
Le résultat de l'association sur DR donnera une description DR incomplète de la forme DRA ⊕ DRNA ⊕ DBNA ⊕ VCOMMUN.
Le résultat de l'association sur DB donnera une description DB incomplète de la forme DBA ⊕ DBNA ⊕ DRNA ⊕ VCOMMUN.

5.2.3. Les descriptions de type règle

La notion d'associabilité, que nous venons de définir, s'appuie sur les types d'entités qui ont été définies pour la théorie des DCAS. Nous allons voir, dans les paragraphes qui suivent, que l'organisation d'autres représentations régit les raisonnements menés par le système.

Les entités cognitives du formalisme proposé incluent les entités de type règle. Celles-ci permettent au système de déduire de nouvelles connaissances. Le système utilise principalement des descriptions de ce type dans les processus d'inférences enclenchés dans la recherche d'information.

Une description de type règle est formée principalement en deux groupes de quadruplets:

- 1 groupe jouant le rôle d'antécédents. Ces éléments sont aussi nommés déclencheurs de règles.
- 1 groupe jouant le rôle de conséquents, qui sont généralement issus des effets de déclenchement de cette règle.

La forme générale d'une règle est la suivante :

Forme générale :
{(TYPE_ENTITE_COMPOSEE_1, antécédent, 1, REF_ENTITE_A_1),
(TYPE_ENTITE_COMPOSEE_2, antécédent, 2, REF_ENTITE_A_2),
...
(TYPE_ENTITE_COMPOSEE_N, antécédent, N, REF_ENTITE_A_N),

(TYPE_ENTITE_COMPOSEE_1', conclusion, 1, REF_ENTITE_C_1')
(TYPE_ENTITE_COMPOSEE_2', conclusion, 2, REF_ENTITE_C_2'),
...
(TYPE_ENTITE_COMPOSEE_N', conclusion, N', REF_ENTITE_C_N'),

(symbole, type-descripteur, *, règle),
(symbole, référent, *, REGLE_R),
(*, *, *, *)
}

Sémantique associée :

« Si les entités 'REF_ENTITE_A_1', 'REF_ENTITE_A_2', 'REF_ENTITE_A_N' (celles qui jouent le rôle d'antécédent) existent dans la base de descriptions au niveau de profondeur 1 ou bien sont toutes déductibles à partir de cette même base, alors les entités 'REF_ENTITE_C_1' ', 'REF_ENTITE_C_2' 'et 'REF_ENTITE_C_N' sont alors dites déductibles de cette même base de descriptions. »

Les raisonnements menés par le système tiennent compte des régularités (comme les rôles antécédents et conclusion) que nous venons de décrire.

Exemple :

La règle

<<Si X aime Y dans un intervalle de temps donné, et Y aime Z dans ce même intervalle de temps, et X <> Z alors X est malheureux dans ce même intervalle de temps>>

pourrait être représentée par la description ci-dessous:

```
{ (situation, antécédent, 1,
                  {(instance, acteur, *, X)
                   (concept, act, *, aimer),
                   (instance, objet, *, Y),
                    (rt, début, *, RTD),
                    (rt, début, *, RTF),
                   (vdv , *, *, vrai)
                   }),
  (situation, antécédent, 2,
                  {(instance, acteur, *, Y)
                   (concept, act, *, aimer),
                   (instance, objet, *, Z),
                    (rt, début, *, RTD),
                    (rt, début, *, RTF),
                   (vdv , *, *, vrai)
                   }),
  (situation, antécédent, 3,
                  {(instance, sujet, *, X)
                   (relation, prédicat, *, ≠),
                   (instance, valeur_référence, *, Z),
                   (vdv , *, *, vrai)
                   }),
  (situation, conclusion, *,
                  {(instance, sujet, *, X)
                   (relation, prédicat, *, malheureux),
                    (rt, début, *, RTD),
                    (rt, début, *, RTF),
                   (vdv , *, *, vrai)
                   })
  (symbole, type-descripteur, *, règle),
  (symbole, référent, *, règle_malheureux),
  (*,*,*,*)
}
```

5.2.4. Processus d'inférences

Soit une description référence donnée au système.

Cette description référence peut être de nature complète ou incomplète.

❶ Le système standardise d'abord la description référence.

Cette opération de standardisation consiste à transformer toute description en une forme dite standard.

Prenons la succession de requêtes suivantes :

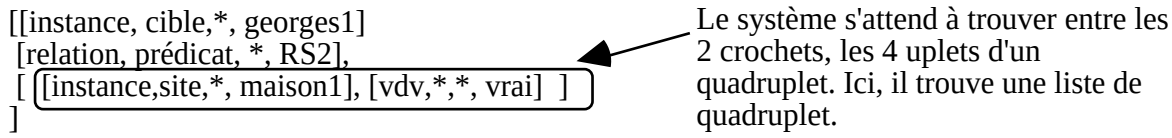
ask(DT, [[instance, R, *, jean1], [relation, prédicat, *, RS], X],

ask(DT2, [[instance, R, *, georges1], [relation, prédicat, *, RS2], X].

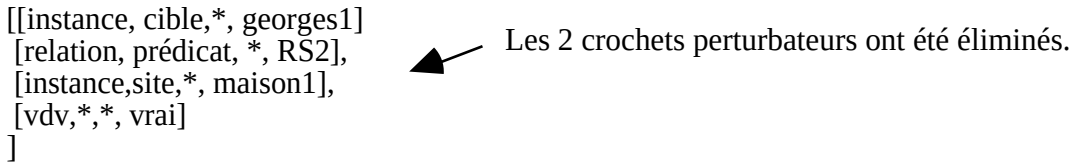
où une même variable X est utilisée pour décrire les éléments terminaux.

Lorsque la première requête est réalisée avec succès par le système, la variable X sera liée à la partie manquante de la description référence. Comme cette variable se retrouve aussi dans la seconde requête et plus précisément en tant que partie manquante de la description

référence, le système trouvera alors une description référence sous une forme non standard.
Par exemple, si X, dans l'exemple ci-dessus, était lié à la valeur "[[instance, site,*,maison1],[vdv,*,*,vrai]]", la description référence de la deuxième requête deviendrait alors



Après standardisation, la description référence deviendra :



Ainsi, si cette dernière description contient une entité de type mesure, elle est alors normalisée. Et la description référence est modifiée. Plus généralement, dans le cas où la description contient un quadruplet de type complexe, les informations normalisables le sont.

❷ Le système parcourt toutes les descriptions assertées. Il calcule l'associabilité de la description référence à chaque description balayée.
Si cette associabilité est établie, alors le système retourne comme description trouvée, le "résultat sur la description référence" de la mise en association de la description référence et la description balayée.
Ce résultat est stocké pour éviter des répétitions éventuelles de données retournées. Le stockage permettra ici d'éviter une perte de temps en calcul de résultats similaires.
Si l'instigateur de la requête exige une autre solution, le système poursuit le balayage des descriptions assertées jusqu'à épuisement de la liste.

❸ Le système parcourt ensuite toutes les descriptions règles de la base. Il calcule l'associabilité de la description référence à la conclusion de chaque description balayée.
Si la mise en relation est établie, le système lance pour chaque antécédent une requête calculée à partir du résultat de l'association de la description référence sur la conclusion de la description règle balayée.
Si l'ensemble des requêtes lancées pour chaque antécédent aboutit, le système retourne comme description trouvée, le "résultat sur la description référence" de la mise en association

de la description référence et la conclusion de la description balayée.
Ce mode de raisonnement est semblable au raisonnement en chaînage arrière. Ce type de raisonnement est ici beaucoup plus adapté à la recherche de réponses à des questions diverses. Cependant, lorsqu'une conclusion est établie, le système introduit la conclusion déduite dans la base pour en tenir compte dans les prochaines inférences. Ce comportement, que l'on peut trouver dans des chaînages dits mixtes, permet d'utiliser les nouvelles connaissances déduites pour en trouver de nouvelles.

❹ Supposons que le système arrive à cette étape sans trouver de solution.
Si les conditions suivantes sont vérifiées :

- la description de référence DR est complète
- aucune variable n'est utilisée dans aucun quadruplet
- il existe soit le quadruplet (vdv, *,*, vrai), soit le quadruplet (vdv, *, *, faux).

le système crée une description DR' issu de DR avec le quadruplet (vdv, *, *, X) remplacé par le quadruplet (vdv,*,*, Y) , Y étant le contraire de Y.
Il part à la recherche de la première information correspondant à la description référence DR' en procédant comme dans ❷ et ❸. Si la recherche aboutit, le système met alors en échec la requête.

❺ Supposons que le système arrive à cette étape sans trouver de solution.
Si les conditions suivantes sont vérifiées :

- la description de référence DR est complète
- un quadruplet de la forme (vdv, *,*, VAR) avec VAR un référent variable
- aucune autre quadruplet ne contient des variables

le système renverra alors la description DR en associant à VAR la valeur 'inconnu'.

5.2.5. Un exemple classique

Supposons la base de connaissances composée des DCAS suivantes :

```
/* Nicolas aime Sophie.*/
    [     [instance,agent,*,nicolas1],[concept,act,*,aimer],
          [instance,objet,*,sophie1],
          [rt,début,*,rt1],[rt,fin,*,rt2],
          [vdv,*,*,vrai],[*,*,*,*],
          [symbole,td,*,situation],[symbole,rd,*,s1]]
      ]).

/* Jean_Louis aime Dominique */
    [     [instance,agent,*,jean_louis1],[concept,act,*,aimer],
          [instance,objet,*,dominique1],
          [rt,début,*,rt1],[rt,fin,*,rt2],
          [vdv,*,*,vrai],[*,*,*,*],
          [symbole,td,*,situation],[symbole,rd,*,s2]]

/* Dominique aime Bob */
    [     [instance,agent,*,dominique1],[concept,act,*,aimer],
          [instance,objet,*,bob1],
          [rt,début,*,rt1],[rt,fin,*,rt2],
          [vdv,*,*,vrai],[*,*,*,*],
          [symbole,td,*,situation],[symbole,rd,*,s3]]

/* X aime Y et Y aime Z  => malheureux X'  */
[
    [situation,antécédent,1,
          [[instance,agent,*,X],[concept,act,*,aimer],
           [instance,objet,*,Y],
           [rt,début,*,RT1],[rt,fin,*,RT2],
           [vdv,*,*,vrai]]],

    [situation,antécédent,2,
          [[instance,agent,*,Y],[concept,act,*,aimer],
           [instance,objet,*,Z],
           [rt,début,*,RT1],[rt,fin,*,RT2],
           [vdv,*,*,vrai]]],

    [situation,antécédent,3,
                [[instance,argument,1,X],[relation,prédicatProlog,*,\=],
                [instance,argument,2,Z], [vdv,*,*,vrai]]],

    [situation,conclusion,*,
          [[instance,argument,1,X],
           [relation,prédicat,*,malheureux],
           [rt,début,*,RT1],[rt,fin,*,RT2],
           [vdv,*,*,vrai]]],
           [*,*,*,*]]]

    [symbole,td,*,règle],
    [symbole,rd,*,r3]]
]
```

Prenons la requête correspondant à « Qui est malheureux à un moment donné ? »

```
ask(DT, [   [relation,prédicat,*,malheureux] ,
            [rt,début,*,rt1],[rt,fin,*,rt2],
            [vdv,*,*,vrai], XA] )
```

Aucune information préalable concernant quelqu'un de malheureux n'est connue.

A l'issue de l'étape 2, le système n'a rien trouvé.

A l'étape 3, on parcourt les descriptions règle.

La description référence est associable à la conclusion de la règle r3.

La variable XA est tout d'abord liée à [[instance, argument,1,X]] avec X non encore unifié. Le système va lancer une recherche sur toutes les descriptions présentes dans la partie antécédent.

Pour le premier antécédent, il trouvera d'abord que Nicolas aime Sophie.

Il lance une recherche sur le second antécédent. Comme, il ne trouve aucune information sur la personne aimée par Sophie, le système demande un autre choix pour le premier antécédent. Il trouve en premier antécédent que Jean-Louis aime Dominique. Le second antécédent deviendra « Dominique aime X ». Le système trouve comme valeur du second antécédent que Dominique aime Bob.

Le troisième antécédent se traduirait par la phrase « Est-ce que Dominique ≠ Bob ? ».

A cette requête, le système indiquera un succès.

Jean-Louis est unifié à la variable X.

La variable XA deviendra alors :[[instance, argument,1,jean_louis1]]

Le fonctionnement que nous venons de décrire ressemble en beaucoup de points à celui d'un moteur d'un système expert. Par rapport à ces systèmes, le système des DCAS est intéressant sur plusieurs points :

- les règles s'écrivent de la même manière que les connaissances qui sont manipulées. Cette propriété est importante car cela signifie que les règles peuvent être manipulées comme les autres objets. Dans une étape ultérieure, on pourrait envisager que le système puisse faire des raisonnements sur des règles avec d'autres règles.

- La notion d'associabilité que nous avons défini nous permet d'exprimer dans les requêtes une partie des informations. Grâce à la notion de rôle comme participation de l'entité cognitive au sens d'une DCAS, la partie décrite forme un sous-ensemble d'informations sémantiques qui doivent être présentes dans les descriptions recherchées.

5.2.6. L'héritage d'informations

Deux règles "règle-hierarchie-entre-concept" et "règle-instance-concept" permettent d'inférer de nouvelles informations au niveau des liens entre concepts ou instances-concepts.

La règle "règle-hierarchie-entre-concept " définit les liens entre concepts.

<<Si un concept C1 est subsumé par un concept C2, et que ce dernier est subsumé par un concept C3, alors C1 est aussi subsumé par ce concept C3.>>

```
{ (proposition, antécédent, 1,
                  {(concept, subsumé, *, C1)
                   (relation, prédicat, *, est-un),
                   (concept, subsumeur, *, C2),
                   (vdv , *, *, vrai)
                   }),
  (proposition, antécédent, 2,
                  {(concept, subsumé, *, C2)
                   (relation, prédicat, *, est-un),
                   (concept, subsumeur, *, C3)
                   (vdv , *, *, vrai)
                   }),
  (proposition, conclusion, *,
                  {(concept, subsumé, *, C1)
                   (relation, prédicat, *, est-un),
                   (concept, subsumeur, *, C3),
                   (vdv , *, *, vrai)
                   }),
  (symbole, type-descripteur, *, règle),
  (symbole, référent, *, règle-hierarchie-entre-concept),
  (*,*,*,*)
}
```

La règle "règle-instance-concept" permet à une instance d'un concept C1 d'hériter de toutes les "propriétés" définies pour toute instance d'un concept C2 avec C2 subsumant C1.

<<Si une instance I est dénotée par un concept C1, et que le concept C1 est subsumé par un concept C2, alors l'instance I est dénotée par le concept C2. >>

```
{ (proposition, antécédent, 1,
                  {(instance, dénoté, *, I)
                   (relation, prédicat, *, est-un),
                   (concept, dénoteur, *, C1),
                   (vdv , *, *, vrai)
                   }),
  (proposition, antécédent, 2,
                  {(concept, subsumé, *, C1)
                   (relation, prédicat, *, est-un),
                   (concept, subsumeur, *, C2)
                   (vdv , *, *, vrai)
                   }),
  (proposition, conclusion, *,
                  {(concept, dénoté, *, I)
                   (relation, prédicat, *, est-un),
                   (concept, dénoteur, *, C2),
                   (vdv , *, *, vrai)
                   }),
  (symbole, type-descripteur, *, règle),
  (symbole, référent, *, règle-instance-concept),
  (*,*,*,*)
}
```

Ces 2 règles sont principalement utilisées lorsque l'on cherche à savoir si une instance I est dénotée par un concept C. Cette recherche est menée à l'intérieur même du processus d'inférences car la notion d'associabilité fait un large appel à ce type de requête. Il est intéressant de constater que le mécanisme d'inférences dans un système à base de DCAS est

basé sur des notions qui peuvent être définies explicitement par l'intermédiaire de structures fournies au système.

Grâce à ces deux règles, les informations :

- le concept chat subsume le concept siamois

```
{ (concept, subsumé, *, siamois) ,(relation, *, *, est-un),
  (concept, subsumeur, *, chat) , (vdv, *, *, vrai)
}
```

- le concept animal subsume le concept chat

```
{ (concept, subsumé, *, chat) , (relation, *, *, est-un),
  (concept, subsumeur, *, animal) , (vdv, *, *, vrai)

}
```

- une instance de siamois mange une instance de souris.

```
{ (concept, dénoteur, *, siamois) ,
  (relation, *, *, est-un),
  (instance, dénoté, *, siamois15) , (vdv, *, *, vrai)

}
{ (concept, dénoteur, *, souris) ,(relation, *, *, est-un),
  (instance, dénoté, *, souris2) , (vdv, *, *, vrai)

}
{ (concept, act, *,       manger),
  (instance, acteur, *, siamois15),
  (instance, objet, *, souris2),
  (vdv , *, *,            vrai),
  (rt, début, *, rt34),
  (rt, fin, *, rt35)
}
```

permettent de répondre à la question « Y-a-t-il un animal qui mange ? » que l'on peut traduire par la composition de deux recherches de descriptions partielles DP1 et DP2.

```
DP1 = { (concept, act, *,       manger),
        (instance, acteur, *, X),
        (vdv , *, *,            vrai),
        (rt, début, *, rt34),
        (rt, fin, *, rt35),
        X
      }


DP2 = { (concept, dénoteur, *, animal) ,
        (relation, *, *, est-un),
        (instance, dénoté, *, IANIMAL) ,
        (vdv, *, *, vrai)
      }
```

On notera que l'on aurait pu se contenter d'une unique description partielle. Au lieu d'avoir un quadruplet de type instance dans DP1, nous aurions pu utiliser le quadruplet de type iensemble suivant :

```
(iensemble, acteur,*,
                {     (instance, mr,*,  IMR),
                      (symbole, type_participation, *,distributif),
                      (nombre, cardinal, *, N),
                      (proposition, cr, *,
                            {     (instance,dénoté, *, IMR),
                                  (concept,dénoteur,*, animal),
                                  (relation, prédicat,*, est_un),
                                  (vdv, *,*, vrai)
                            }),
                       (proposition, cr, *,
                            {     (nombre,sujet, *, N),
                                  (nombre,valeur_référence,*, 1),
                                  (relation, prédicat,*, =),
                                  (vdv, *,*, vrai)
                            }),
                })
```

Le système trouvera la condition n°4 d'associabilité remplie en recherchant à un instant donné si l'instance siamois15 est dénotée par le concept "animal".

5.2.7. Optimisation des inférences

L'associabilité d'un quadruplet à un autre engendre fréquemment la vérification d'un certain nombre de conditions (comme par exemple, la dénotation d'une instance IX par un concept CX). Ces appels sont extrêmement fréquents et conditionnent la rapidité du système.

C'est pour accélérer le système que nous proposons dans un système à base de DCAS de compiler les descriptions de la forme

```
{(concept, subsumé, *, oiseau) , (relation, prédicat, *, est-un),
 (concept, subsumeur, *, animal) , (vdv, *, *, vrai)}
```

Une règle interne permettra de déduire les relations de dénotations entre un concept et une instance.

D'autres déductions peuvent aussi faire l'objet d'améliorations par l'intégration de règles internes et par la compilation de certaines formes de connaissances :

```
{(concept, subsumé, *, C1) , (relation, prédicat, *, est-un),
 (concept, subsumeur, *, C2) , (vdv, *, *, faux)}

{(concept, *, *, C1) , (relation, prédicat, *, disjoint),
 (concept, *, *, C2) , (vdv, *, *, vrai)}

{(concept, *, *, C1) , (relation, prédicat, *, disjoint),
 (concept, *, *, C2) , (vdv, *, *, faux)}

{(concept, dénoteur, *, C) , (relation, *, *, est-un),
 (instance, dénoté, *, I) , (vdv, *, *, vrai)}

{(concept, dénoteur, *, C) , (relation, *, *, est-un),
 (instance, dénoté, *, I) , (vdv, *, *, faux)}
```

Chapitre 4 : Application du formalisme dans le cadre de la problématique SYSQUEST

Nous montrons dans ce chapitre comment ce formalisme est utilisé pour résoudre la problématique SYSQUEST. Tout d'abord, nous décrirons les DCAS qui sont utilisées pour représenter les différents aspects de la localisation spatiale retenus dans la première partie de cette thèse. Ensuite, nous décrirons les règles permettant de déduire de nouvelles informations relatives aux localisations spatiales.

Dans un dernier temps, nous montrerons l'utilisation de ces représentations, dans le cadre de AMICAL, pour l'aide au choix de questions de compréhensions posées.

1. Représentations

Pour représenter une connaissance spatiale, nous employons les trois rôles:

prédicat, cible, site.

Le rôle prédicat est utilisé pour indiquer le type de relation spatiale mise en jeu.

Le rôle cible permet de reconnaître dans une relation spatiale l'objet qui est la cible de la relation spatiale.

Le rôle site donne l'entité cognitive qui sert d'objet de référence dans la relation spatiale.

Si la relation spatiale met en jeu un site unique (comme par exemple, avec les relations "dans", "sur", "sous", "assis-sur", etc...) , il n'existe alors qu'un quadruplet dans la DCAS ayant ce rôle.

Par contre, si la relation admet deux sites (comme c'est le cas de la relation spatiale "entre"), il existe alors deux quadruplets ayant le rôle site. La distinction se fait en utilisant le numéro de rôle.

1.1. Les représentations associées aux relations spatiales entre objets “concrets”

La DCAS correspondante à la phrase « Nicolas est dans la maison » est la suivante :

```
{   (instance, cible,*,nicolas1),
     (relation, prédicat, *, dans),
     (instance, site, *, maison12),
     (vdv, *, *, vrai)
     (rt, début, *, rt5),
     (rt, fin, *, rt6)
}
```

Elle met en œuvre :

- une instance du concept “maison” jouant le rôle de site
- une instance du concept “personne” jouant le rôle cible.
- la relation “dans”
- la valeur de vérité “vrai”
- deux repères temporels permettant de situer cet événement dans le temps.

Lorsque le référent de l’instance cible ou site n’est pas connu avec exactitude, on trouve un quadruplet de type iensemble.

Par exemple, à la phrase, « Un chat est dans la maison », nous verrons associer la DCAS

```
{   (iensemble, *,*,
     {  (instance, mr, *, IMR),
         (nombre, cardinal, *, N),
         (symbole,type_participation,*,distributif),
         (proposition, cr,
          {(instance,dénoté,*,IMR),(concept,dénoteur,*,chat),
            (relation,prédicat,*,est_un),(vdv,*,*,vrai)
          }),
          {(nombre,sujet,*,IMR),(nombre, valeur_référence,*,1),
            (relation,prédicat,*,=),(vdv,*,*,vrai)
          }),
     }),
     (relation, prédicat, *, dans),
     (instance, site, *, maison12),
     (vdv, *, *, vrai)
     (rt, début, *, rt5),
     (rt, fin, *, rt6)
}
```

Les quadruplets de type iensemble permettent aussi de décrire des connaissances comme « Nicolas, Sophie et John sont dans la maison ». Un iensemble sera associé à “Nicolas, Sophie et John”. Cet iensemble sera de la forme :

```
{  (instance, mr, *, IMR),
   (nombre, cardinal, *, N),
   (instance, membre, 1, nicolas1),
   (instance, membre, 2, sophie1),
   (instance, membre, 3, john1),
   (symbole,type_participation,*,distributif),
   (proposition, cr,
     {(instance,dénoté,*,IMR),(concept,dénoteur,*,personne),
      (relation,prédicat,*,est_un),(vdv,*,*,vrai)
     }),
     {(nombre,sujet,*,IMR),(nombre, valeur_référence,*,3),
      (relation,prédicat,*,=),(vdv,*,*,vrai)
     }
```

La recherche d'une relation spatiale se fera à partir de descriptions références.

Pour savoir où se trouve Nicolas, il suffit de donner la forme suivante :

```
{ (instance, cible, *, nicolas1),
    (relation, prédicat,*,RS),
    (vdv,*,*,vrai),
    X }
```

Ainsi, la requête

```
?- ask(DT, Chemin,      [     [instance, cible, *, nicolas1],
                              [relation, prédicat,*,RS],
                              [vdv,*,*,vrai],  X])
```

associera à DT toutes les connaissances où Nicolas jouele rôle de cible de la relation spatiale. Si l'on recherche l'élément dans lequel Nicolas est l'objet de référence de la localisation spatiale, il suffit d'utiliser :

```
{ (instance, site, NR, nicolas),
  (relation, prédicat,*,RS),
  (TE, cible,*, REF_E),
  (vdv,*,,*, vrai), X}
```

Si l'on recherche la présence d'un homme dans la maison, il suffit de donner la description référence ci-dessous :

```
{   (iensemble, cible,*,
     {  (instance, mr, *, IMR),
         (nombre, cardinal, *, N),
         (symbole,type_participation,*,distributif),
         (proposition, cr,
          {(instance,dénoté,*,IMR),(concept,dénoteur,*,homme),
            (relation,prédicat,*,est_un),(vdv,*,*,vrai)
          }),
          {(nombre,sujet,*,IMR),(nombre, valeur_référence,*,1),
            (relation,prédicat,*,=),(vdv,*,*,vrai)
          }),
     }),
      (relation, prédicat, *, dans),
      (instance, site, *, maison12),
      (vdv, *, *, vrai)
      (rt, début, *, rt5),
      (rt, fin, *, rt6)
}
```

A l'étape ❷ du processus d'inférences, la condition n°4 d'associabilité de quadruplet sera vérifiée pour le quadruplet iensemble correspondant à "un homme" et le quadruplet (instance, cible, *, nicolas1).

Note 1 :

La requête

```
?- ask(DT, Chemin, [ [instance, cible, *, nicolas1], [relation, prédicat,*,RS],
                                    [vdv,*,*,vrai],  X])
```

indiquera dans DT toutes les localisations où intervient Nicolas, et quel que soit le nombre de sites utilisés par la relation spatiale.

Note 2 :

Il nous semble intéressant de comparer cette façon d'exprimer la requête à celle que l'on serait amené à faire si des représentations du type Prolog étaient choisies.

Supposons que la base de connaissances initiale contienne les faits suivants :

```
dans(rt1, rt2, jean1, maison1). /* Jean est dans la maison */
entre(rt1, rt2, jean1, chaise1, lit1). /* jean est entre la chaise et le lit */
```

Le calcul de la localisation de jean1 se fera par l'intermédiaire des clauses règles ouEst :

```
ouEst(X,Reponse) :-
      def(dans), dans(rt1,rt2, X, Y), Reponse = [dans, Y].
ouEst(X,Reponse) :-
      def(sur), sur(rt1,rt2, X, Y), Reponse = [sur, Y].
  .....
ouEst(X,Reponse) :-
      def(entre), entre(rt1,rt2, X, Y,Z), Reponse = [entre, Y,Z].
```

La requête « ?- ouEst(jean1, X) » lance la recherche de la localisation de jean1.

Pour rendre cette recherche possible, la requête a du être programmée. On remarquera qu'ici, il a été nécessaire d'énumérer tous les prédicats associés aux prépositions spatiales.

Note 3 :

La requête

```
?- ask(DT, Chemin, [ [instance, site, NRSITE, nicolas1],
                                    [relation, prédicat,*,RS],
                                    [vdv,*,*,vrai],  X])
```

indiquera dans DT toutes les localisations où intervient Nicolas en tant que site et quel

que soit le nombre de sites utilisés par la relation spatiale. Si l'on compare cette requête aux trois types systèmes de représentations présentés au chapitre 2, on constatera que cette formulation est simple dans un système à base de DCAS. Nous sommes en accord avec la contrainte n°5 de type génie logiciel.

1.2. Localisations spatiales quantifiées

Pour décrire une connaissance générale telle que « Toute chambre d'une maison se trouve dans cette même maison », nous emploierons des CENSEMBLEs.

A chaque élément d'information quantifiable de la connaissance générale à intégrer, nous associerons généralement un CENSEMBLE.

Par exemple, pour la phrase « Toute chambre de maison se trouve dans cette même maison. », nous utiliserons deux censembles. Le premier est relatif à "toute chambre de maison", le second à "même maison". Ainsi, nous obtenons la description suivante **D1**:

```
{       (censemble, cible, *,
          {   (instance, mr, *, C),
              (nombre, cardinal, *, N1),
              (symbole, quantificateur, *, tout)
              (symbole, type_participation, *, distributif)
              (proposition, cr, * ,
                    {     (instance, subsumé, *, C),
                          (relation, prédicat,*,est_un),
                          (concept, subsumeur,*, chambre),
                          (vdv , *,*, vrai)}),
              (proposition, cr, * ,
                    {     (instance, sujet, *, N),
                          (relation, prédicat,*,=),
                          (nombre, valeur_référence,*, 1),
                          (vdv , *,*, vrai)}),
              (proposition, cr, * ,
                    {     (instance, composé, *, C),
                          (instance, composant,*, COREF_M),
                          (vdv , *,*, vrai)
                    }),
              (proposition, cr, * ,
                    {     (instance, subsumé, *, COREF_M),
                          (relation, prédicat,*,est_un),
                          (concept, subsumeur,*, maison),
                          (vdv , *,*, vrai)})}),
        (relation, prédicat, *, dans),
        (censemble, site, *,
          {   (instance, mr, *, M),
              (instance, coref_mr, *, COREF_M)
              (nombre, cardinal, *, N2),
              (symbole, quantificateur, *, tout)
              (symbole, type_participation, *, distributif)
              (proposition, cr, * ,
                    {     (instance, subsumé, *, M),
                          (relation, prédicat,*,est_un),
                          (concept, subsumeur,*, maison),
                          (vdv , *,*, vrai)}),
              (proposition, cr, * ,
                    {     (instance, sujet, *, N2),
                          (relation, prédicat,*, =),
                          (nombre, valeur_référence,*, 1),
                          (vdv , *,*, vrai)}),
        (vdv , *, *, vrai)
        }}
```

Supposons que les connaissances suivantes appartiennent au système :

```
{(instance, subsumé, *, chambre1),
  (relation, prédicat,*,est_un),
  (concept, subsumeur,*, chambre),
  (vdv , *,*, vrai) }

{(instance, subsumé, *, maison1),
  (relation, prédicat,*,est_un),
  (concept, subsumeur,*, maison),
  (vdv , *,*, vrai) }

{(instance, composé, *, chambre1),
  (concept, composant,*, maison1),
  (vdv , *,*, vrai) }
```

Lorsque la description référence {(instance, cible, *, chambre1), X } est donnée, le système déduira à partir de :

- la description D1 ,
- la condition 3 (cas 1) d'associabilité d'une instance à un CENSEMBLE applicable au quadruplet instance chambre1.

la valeur de X suivante :

```
{ (relation, prédicat, *, dans),
  (instance, site, *, maison1)
  (vdv , *, *, vrai)
}
```

1.3. L'aspect temporel dans une relation spatiale

Le couple de quadruplets sous la forme

```
{  (rt, début, *, RT_DEB),
   (rt, fin, *, RT_FIN)
}
```

permet d'associer à chaque description des informations temporelles.

1.4. Localisation spatiale d'événements

Les informations sur les événements survenus dans le monde du discours se traduisent en termes de descriptions de type situation. Un événement peut survenir à des emplacements qui peuvent être précisés. Par exemple, dans « Jean fabrique une chaise avec un marteau dans la maison. », l'action se situe dans la maison.

Pour représenter des phrases de ce type, nous créerons des descriptions dans lesquelles nous retrouverons les éléments suivants :

- une entité cognitive de type situation qui jouera le rôle de cible
- une entité relation pour la relation spatiale
- une ou plusieurs entités jouant le rôle de site
- deux repères temporels

Nous obtenons à partir de « Jean fabrique une chaise avec un marteau dans la maison. » la description suivante :

```
{ (situation, cible, *,
    {   (concept, act, *,       fabriquer),
        (instance, acteur, *, jean1),
        (instance, objet, *, chaise1),
        (instance, instrument, *, marteau1),
        (vdv , *, *,                  vrai),
        (rt, début, *, rt32),
        (rt, fin, *, rt33)
    }
  (relation, prédicat, *, dans),
  (instance, site, *, maison25)
  (rt, début, *, rt32),
  (rt, fin, *, rt33)
}
```

On notera que la description ci-dessus est une description de type situation. Les repères temporels utilisés au niveau de profondeur 1 sont les mêmes que ceux rencontrés dans le référent de la situation cible.

La phrase « Michel croit que Jean fabrique une chaise avec un marteau dans la maison. » pose quelques difficultés de représentation.

La première possibilité de traduction en terme de DCAS est :

```
{ (instance, acteur, *, michel),
  (concept, act, *, croire),
  (proposition, objet, *,
   {   (situation, cible, *,
              {(concept, act, *, fabriquer),
               (instance, acteur, *, jean1),
               (instance, objet, *, chaise1),
               (instance, instrument, *, marteau1),
               (vdv , *, *,  vrai),
               (rt, début, *, rt32),
               (rt, fin, *, rt33)
              }
        (relation, prédicat, *, dans),
        (instance, site, *, maison25)})
  (vdv , *, *, vrai),
  (rt, début, *, rt32),
  (rt, fin, *, rt33)
}
```

Nous restons en accord avec Wilensky, dans le sens où l'objet de ce que croit Michel est bien une proposition. Cette proposition décrit la relation spatiale entre une entité jouant le rôle de site et une situation donnée. Cette dernière (dans la description de niveau de profondeur 3) n'est pas obligatoirement associée à un fait réel. Cette première traduction possible comporte l'inconvénient de focaliser l'objet de la croyance de Michel vers la localisation de l'action de Jean. La seconde possibilité de traduction consiste à proposer deux DCAS. La première DCAS correspond à la première traduction donnée. La seconde DCAS est celle donnée ci-dessous :

```
{  (instance, acteur, *, michel),
   (concept, act, *, croire),
   (proposition, objet, *,
               {(concept, act, *, fabriquer),
                (instance, acteur, *, jean1),
                (instance, objet, *, chaise1),
                (instance, instrument, *, marteau1),
                (vdv , *, *,  vrai),
                (rt, début, *, rt32),
                (rt, fin, *, rt33)
               }),
   (vdv , *, *, vrai),
   (rt, début, *, rt32),
   (rt, fin, *, rt33)
}
```

Une troisième possibilité consiste à réunir ces deux descriptions en une DCAS unique :

```
{  (instance, acteur, *, michel),
   (concept, act, *, croire),
   (proposition, objet, *,
    {   (situation, cible, *,
               {(concept, act, *, fabriquer),
                (instance, acteur, *, jean1),
                (instance, objet, *, chaise1),
                (instance, instrument, *, marteau1),
                (vdv , *, *,  vrai),
                (rt, début, *, rt32),
                (rt, fin, *, rt33)
               }
         (relation, prédicat, *, dans),
         (instance, site, *, maison25)}),
   (proposition, objet, *,
               {(concept, act, *, fabriquer),
                (instance, acteur, *, jean1),
                (instance, objet, *, chaise1),
                (instance, instrument, *, marteau1),
                (vdv , *, *,  vrai),
                (rt, début, *, rt32),
                (rt, fin, *, rt33)
               }),

   (vdv , *, *, vrai),
   (rt, début, *, rt32),
   (rt, fin, *, rt33)
}
```

Pour indiquer laquelle des deux propositions objets est le focus de la phrase, on pourrait envisager un numéro de rôle. Il est clair que ce procédé est beaucoup moins intéressant par rapport à celui présenté dans les graphes conceptuels où c'est dans le choix du concept tête que l'on indique la focalisation.

Il serait intéressant de développer dans le cadre d'extensions ultérieures cette notion de focus dans les systèmes à base de DCAS.

1.5. Localisations spatiales et négation

Un quadruplet de type vdv permet d'associer une valeur de vérité à toute information. Lorsque la localisation spatiale est effective, le quadruplet (vdv, *, *, vrai) est utilisé. Si l'information est niée, le quadruplet (vdv, *, *, faux) est alors présent.

La DCAS correspondante à la phrase « Nicolas n'est pas dans la maison » est la suivante :

```
{   (instance, cible,*,nicolas1),
     (relation, prédicat, *, dans),
     (instance, site, *, maison12),
     (vdv, *, *, faux)
     (rt, début, *, rt5),
     (rt, fin, *, rt6)
}
```

A la phrase « Jean ne casse pas la chaise dans la maison. », nous aurons deux DCAS possibles (figure 4.1). La première décrit une situation « Nicolas ne casse pas la chaise » localisée dans la maison. La seconde décrit une situation « Nicolas casse la chaise » non localisée dans la maison. Suivant le contexte, l'une des deux DCAS sera choisie.

```
{ (situation, cible, *,
   {(concept, act, *, casser),
    (instance, acteur, *, jean1),
    (instance, objet, *, chaise1),
    (vdv , *, *, faux),
    (rt, début, *, rt32),
    (rt, fin, *, rt33)
   }
  (relation, prédicat, *, dans),
  (instance, site, *, maison25),
  (vdv,*,*,vrai),
  (rt, début, *, rt32),
  (rt, fin, *, rt33)
}
```

DCAS1

```
{ (situation, cible, *,
   {(concept, act, *, casser),
    (instance, acteur, *, jean1),
    (instance, objet, *, chaise1),
    (vdv , *, *, vrai),
    (rt, début, *, rt32),
    (rt, fin, *, rt33)
   }
  (relation, prédicat, *, dans),
  (instance, site, *, maison25),
  (vdv,*,*,faux),
  (rt, début, *, rt32),
  (rt, fin, *, rt33)
}
```

DCAS2

figure 4.1

2. Inférences

Pour déduire de nouvelles localisations spatiales, nous introduirons des connaissances de type règle. Nous donnons ici quelques exemples de règles utilisées.

Exemple 1 :

Si une action se passe dans un intervalle de temps donné par le couple (RT1,RT2) et que cette dernière action est localisée à un endroit X, alors l'auteur de l'action se trouve aussi à cet endroit X.

```
{ (situation, antécédent, 1,
                  {(situation, cible, *,
                     { (instance, acteur, *, ACTEUR),
                       (concept, act,*, ACT),
                       (vdv, *, *, vrai),
                       (rt, début, *, RT1),
                       (rt, fin, *, RT2),
                        RESTE_SITUATION
                     }
                   )
                   (relation, prédicat, *, RS),
                   (vdv , *, *, vrai),
                   (rt, début, *, RT1),
                   (rt, fin, *, RT2),
                    RESTE_LOCALISATION_SPATIALE
                   }),

  (situation, conclusion, *,
                  {(instance, cible, *, ACTEUR)
                   (relation, prédicat, *, RS),
                   (rt, début, *, RT1),
                   (rt, fin, *, RT2),
                   (vdv , *, *, vrai),
                    RESTE_LOCALISATION_SPATIALE
                  }),
  (symbole, type-descripteur, *, règle),
  (symbole, référent, *, règle-acteur-situation),
  (*,*,*,*)
}
```

La partie conclusion est associable à toute requête de localisation spatiale (sans contrainte sur le nombre de sites) grâce à la variable RESTE_LOCALISATION_SPATIALE qui contiendra tous les sites de la relation spatiale. La variable RESTE_SITUATION permet de recueillir tous les quadruplets "restant" de la situation cible.

Cette règle peut bien sûr être affinée pour certains types d'actions. Dans ce cas, il suffit alors de donner une autre condition dans la règle, c'est à dire définir un autre quadruplet ayant un rôle "antécédent". (On pourrait par exemple affiner le concept jouant le rôle "act" en le situant par rapport à une classe d'actions.)

Même si la structure semble lourde au premier abord, nous ne pouvons que constater qu'il est simple de décrire de telles connaissances grâce à cette possibilité de décrire des descriptions partielles.

Exemple 2 :

Si quelqu'un est "assis-sur" quelque chose, on peut dire que cette personne possède la

position assis.

Exemple 3 :
Si Z est à côté de Y
et X est sur Z
et Z est de catégorie dimensionnelle inférieure à Y
alors X est à côté de Z.

L'exploitation de la règle 3 n'est concevable que si des informations dimensionnelles sont fournies au système. On pourra donner , par exemple, l'échelle suivante :

univers
système_planétaire, galaxie
planète
continent
pays
région
montagne
immeuble,tour, monument
maison, chalet, appartement
chambre,pièce, cuisine
chat, chien, ordinateur, porte, cage, tête-personne, souris, oiseau, livre, cahier, boite_a_chaussure, tapis, canapé, bureau, lit, personne
puce, page

Il faudra indiquer au système que les instances de immeuble et de tour sont des objets appartenant à la même catégorie dimensionnelle.

3. Utilisation de ces représentations dans le cadre de Amical

3.1. Description systémique de SYSQUEST

Nous commencerons la description de ce système par l'analyse des flux d'informations (figure 4.2).

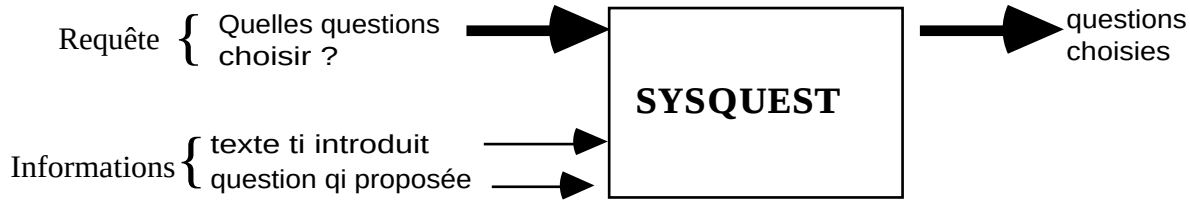


figure 4.2 : Flux d'informations au niveau de SYSQUEST

Deux types de flux existent. Le premier type est de nature informative. Des informations sont envoyées au système pour lui permettre de mettre à jour ses connaissances. Le second type est de nature requête.

Les informations

Ces informations que reçoit le système sont classées en deux catégories :

- celles relatives aux textes lus,
- celles relatives aux questions posées à l'apprenant, ainsi que la réponse attendue et la réponse effectivement donnée.

Le premier groupe d'informations provoque la mise à jour des connaissances de SYSQUEST relatives à l'histoire.

Exemple : « Le texte Ti de l'histoire Hj vient d'être lu. »

Le second groupe permet de faire connaître au système les questions qui ont été posées à l'enfant.

Exemple : « La question Qi de l'histoire Hj a été posée. La réponse attendue était RA. La réponse donnée est RD. La liste de textes qui permet de répondre à cette question est L. »

Les requêtes.

Le système SYSQUEST répond à des demandes de choix de question.

Il n'y a qu'un seul type de requête. Traduite en langage naturel, elle s'écrit :

« Donnez une liste de questions avec la réponse attendue pour chacune des questions. »

A partir de la connaissance des textes introduits, SYSQUEST va élaborer une liste de questions qui pourront être reprises totalement ou partiellement par le système formulant la requête. Le choix des questions de compréhension se fera en tenant compte des conditions de restrictions formulées dans la première partie.

3.2. Implémentation du système SYSQUEST

Le support de représentation à base de DCAS est implémenté en Prolog sous MacOS. Des détails sur son implémentation sont donnés en Annexe 4. Certains cas d'associabilités entre quadruplets (notamment celui de l'associabilité iensemble-censemble) n'ont pas encore été implémentés.

Le système SYSQUEST est aussi implémenté en Prolog.

SIMULREQ est une application de type HYPERCARD.

Les deux applications SYSQUEST et SIMULREQ communiquent via l'intermédiaire des structures appelées "AppleEvents" propre au système Macintosh.

Nous utiliserons une base de DCAS pour représenter l'histoire qui est lue. Ce système pourra accéder, pour toute histoire donnée, aux structures DCAS associées à chacune des phrases composantes. Dans l'implémentation proposée, ces structures DCAS associées aux phrases ne sont pas issues d'un analyseur de textes. Les DCAS sont introduites manuellement.

Pour chaque question qui pourrait être posée tout au long de cette histoire, nous aurons aussi associé un ensemble de DCAS décrivant la question. SYSQUEST utilisera ces descriptions pour effectuer des recherches dans la base de connaissances de l'histoire.

3.3. Base de ressources de SYSQUEST

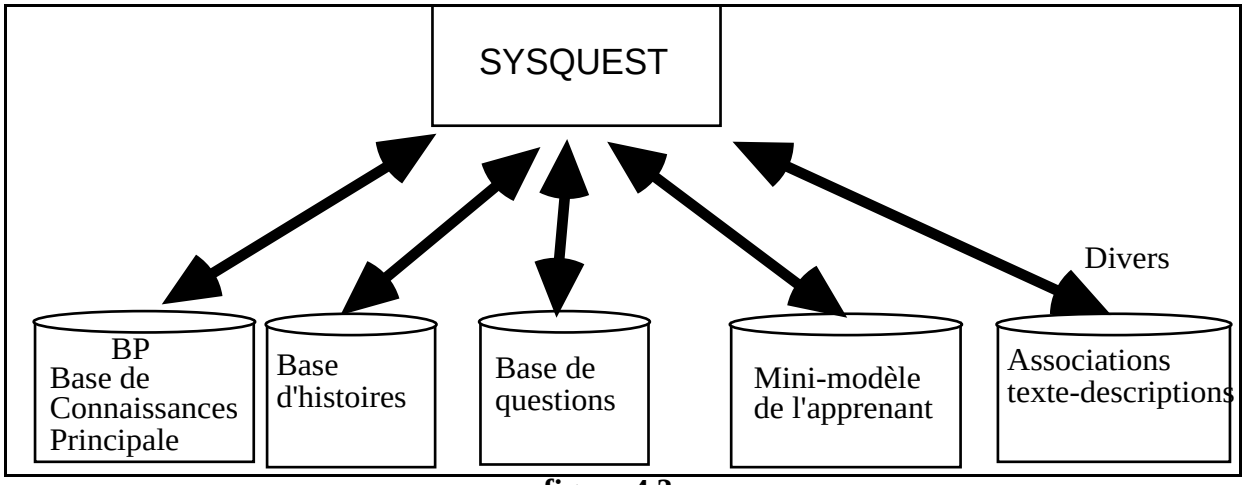


figure 4.3

SYSQUEST utilise 5 bases distinctes de connaissances (figure 4.3) :

- une base de connaissances principale (BP)
- une base d'histoires (BH)
- une base de questions (BQ)
- un modèle restreint de l'apprenant (BA)
- une base de connaissances diverses (BD) contenant des ensembles d'associations texte-descriptions.

3.3.1. Base de connaissances principales (BP)

La base de connaissances principale BP est une base de connaissance modélisée à partir de DCAS. Ces dernières représentent des connaissances générales du monde(en termes de propositions, situations et de règles). Elle contient aussi les connaissances apportées par les textes qui ont été effectivement présentés à l'enfant.

3.3.2. Base d'histoires (BH)

La base d'histoires BH contient un ensemble d'informations sur toutes les histoires qui peuvent être lues. Elle fournit au système les structures de connaissances nécessaires au travail sémantique qui peut être réalisé pour toute histoire, dans le cadre de Amical.

Ce sont :

- des structures propres à l'histoire, comme le début et la fin d'une histoire.
- des structures de connaissances propres à chaque texte d'une histoire.

Le début et de la fin de l'histoire (figure 4.4) permet de connaître les bornes temporelles dans laquelle l'histoire se déroule.

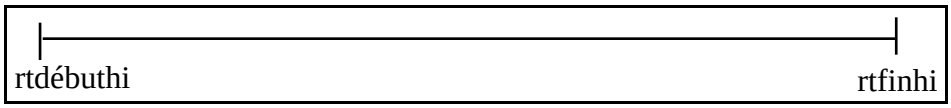


figure 4.4

Cette connaissance dans BH se traduira sous la forme :

histoire(hi, 'Histoire de ...',
[[[rt,début,,<rtdébuth1>],[rt,fin,*,<rtfinh1>]]*
]).

Nous limitons chaque texte ti de toute histoire à une seule phrase.

A chaque texte, seront associées les informations suivantes :

- un ensemble de DCAS dit principal. (EDP)

Cet ensemble contient les DCAS principalement transmises par la phrase. Par exemple, dans la phrase « Nicolas est dans la maison », la DCAS relative à la localisation spatiale de Nicolas appartient à cet ensemble d'éléments.

- un ensemble de DCAS dit secondaire. (EDS)

Cet ensemble contient les DCAS donnant des informations jugées secondaires sur

les éléments mis en œuvre dans EDP. Par exemple, dans la phrase précédemment citée, la description indiquant que Nicolas est une personne, appartient à EDS.

- un couple de deux référents de repères temporels.

Ce couple permet de situer l'observateur virtuel (que nous appellerons OV par la suite) placé par le narrateur à l'intérieur de cette même histoire. Il nous permettra de savoir si l'événement relaté est un événement présent, passé ou futur, par rapport au narrateur. Nous supposerons que les repères temporels associés à cet observateur au cours de la narration de l'histoire sont consécutifs dans le temps et balaient l'histoire de rtdébuthi à rtfinhi.

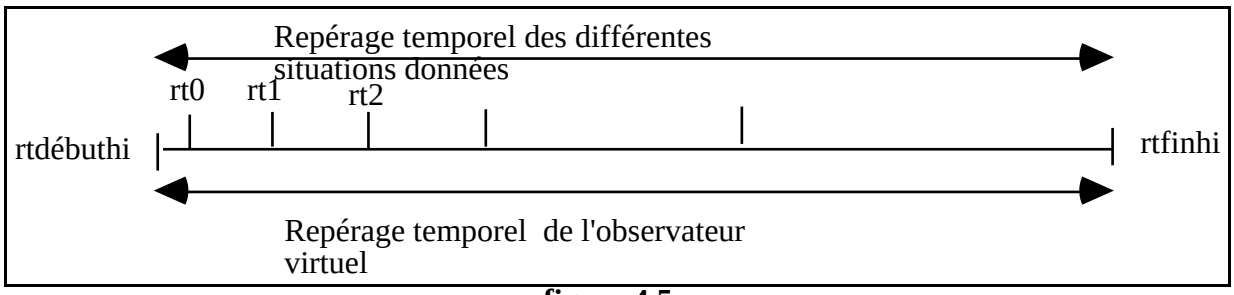


figure 4.5

Prenons, par exemple, les informations associées au texte t5 de l'histoire h1. Elles se présentent sous la forme donnée ci-dessous. La figure 4.6 illustre la position des repères temporels utilisés pour représenter ce texte.

```
texte(h1, t5, 'Dans la maison de Nicolas, il y a un oiseau.',

% EDP
[[[instance,cible,*,oiseau],[relation,*,*,dans],
  [instance,site,*,maison],
   [rt,début,*,rt1],[rt,fin,*,rt2],
    [vdv,*,*,vrai],[*,*,*,*]],

% EDS
 [[[instance,dénoté,*,maison],[relation,*,*,est-un],
   [concept,dénoteur,*,maison],
   [vdv,*,*,vrai],[*,*,*,*]],
 [[instance,dénoté,*,oiseau],[relation,*,*,est-un],
  [concept,dénoteur,*,oiseau],
  [vdv,*,*,vrai],[*,*,*,*]],
 [[instance,possédé,*,maison],[instance,possesseur,*,nicolas],
  [rt, début, *, rtdébuth1],[rt, fin, *, rtfinth1],
  [vdv,*,*,vrai],[*,*,*,*] ],
[[rt,objet,*,rt1],[relation,prédicat,*,>],
 [rt,valeur_référence,*,rt0],
 [vdv,*,*,vrai],[*,*,*,*]],
[[rt,objet,*,rt1],[relation,prédicat,*,proche],
 [rt,valeur_référence,*,rt0],[vdv,*,*,vrai],[*,*,*,*]]],

rt1,rt2
).
```

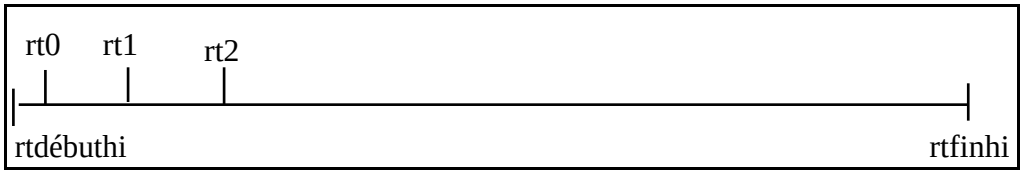


figure 4.6: Repères utilisés dans le texte t5 de h1

Si au lieu de t5, nous avions la phrase t5’ « Dans la maison de Nicolas, il y avait un oiseau. », nous aurions alors :

```
texte(h1, t5', 'Dans la maison de Nicolas, il y avait un oiseau.',

% EDP
[[[instance,cible,*,oiseau],[relation,*,*,dans],
  [instance,site,*,maison],
   [rt,début,*,rt'1],[rt,fin,*,rt'2],
    [vdv,*,*,vrai],[*,*,*,*]],

% EDS
 [[[instance,dénoté,*,maison],[relation,*,*,est-un],
   [concept,dénoteur,*,maison],
   [vdv,*,*,vrai],[*,*,*,*]],
 [[instance,dénoté,*,oiseau],[relation,*,*,est-un],
  [concept,dénoteur,*,oiseau],
  [vdv,*,*,vrai],[*,*,*,*]],
 [[instance,possédé,*,maison],[instance,possesseur,*,nicolas],
  [rt, début, *, rtdébuth1],[rt, fin, *, rtfinth1],
  [vdv,*,*,vrai],[*,*,*,*] ],

[[rt,objet,*,rt'1],[relation,prédicat,*,<],
 [rt,valeur_référence,*,rt1],
 [vdv,*,*,vrai],[*,*,*,*]],
[[rt,objet,*,rt'1],[relation,prédicat,*,proche],
 [rt,valeur_référence,*,rt0],[vdv,*,*,vrai],[*,*,*,*]]

[[rt,objet,*,rt1],[relation,prédicat,*,>],
 [rt,valeur_référence,*,rt0],
 [vdv,*,*,vrai],[*,*,*,*]],
[[rt,objet,*,rt1],[relation,prédicat,*,proche],
 [rt,valeur_référence,*,rt0],[vdv,*,*,vrai],[*,*,*,*]]],

rt1,rt2
).
```

Ici, comme le montre la figure ci-dessous, nous situons l'observateur virtuel après l'intervalle de temps (rt'1,rt'2) durant lequel l'oiseau est dans la maison.

Figure 4.7 : Repères utilisés dans le texte t5' de h1

Lorsqu'une situation ou proposition utilise des repères temporels (RT), ces derniers sont toujours précisés par rapport à d'autres repères supposés connus.

Pour éviter d'alourdir le système avec des informations redondantes sur les relations (de succession, par exemple) existant entre les RTs utilisés dans EDP et tous les autres RTs qui seront utilisés dans l'histoire, nous avons choisi l'hypothèse de travail suivante :

Hypothèse : Lorsque nous travaillons sur les RTs d'un texte t_i, nous connaissons toutes les informations relatives au repères temporels utilisés dans les textes t_{i-1}.

Pour chacune des phrases, nous connaissons donc l'histoire à laquelle elle appartient ainsi que sa description sémantique sous forme d'ensemble de DCAS. Plusieurs DCAS peuvent être associées à une phrase unique.

Note :

Les descriptions associées aux textes sont dans un premier temps introduites par un opérateur humain. Ce dernier doit être capable de construire les DCAS associées à un texte. Dans le cadre des extensions possibles, on peut envisager la création d'une interface texte/graphique lui fournissant toute l'aide nécessaire à l'écriture de ces DCAS. Cette interface pourrait intégrer des outils basiques de génération de DCAS à partir de phrases en langage naturel.

3.3.3. Base de questions (BQ)

La base de questions BQ contient toutes les questions qui peuvent être posées à l'enfant. Elle contient des références vers les ressources sonores associées aux questions ainsi que l'ensemble de descriptions sémantiques associées à chaque question. Ces structures permettront de calculer la réponse à cette question.

Ces informations sont stockées sous la forme suivante :

question(<référence histoire>, <référence question>, <texte de la question>,
<information sur la localisation de la ressource sonore>,
<contexte temporel 1>, <contexte temporel 2>,
<liste des éléments inconnus dans la question>,
<ensemble des descriptions principales dans la question>,
<ensemble des descriptions secondaires dans la question>,)

La référence de l'histoire permet de savoir à quelle histoire particulière correspond la question. Cela permet d'éviter à SYSQUEST de parcourir toutes les questions possibles de toutes les histoires à chaque requête.

La référence de la question doit être utilisée conjointement avec le référent de l'histoire pour identifier complètement la question. En effet, deux mêmes référents peuvent être utilisés par deux questions différentes. L'information qui doit permettre de choisir l'une plutôt que l'autre est la référence de l'histoire utilisée.

Le texte de la question est utilisé en vue de son affichage éventuel.

Le système a besoin de savoir le lieu de la ressource sonore associée à la question.

Les contextes temporels permettent de donner les repères temporels associés à l'observateur virtuel placé par le narrateur au moment où la question est posée. Ce sont en général deux variables qui sont données ici.

La liste des éléments inconnus de la question est utilisée par le système pour connaître le type de la question. Si par exemple, cette liste était réduite à NIL, la question posée est de type oui/non/jpps. Si cette liste est non vide, l'élément donné est une inconnue dans le problème qui est soit explicitement nommée, soit implicitement utilisée dans la question.

Par exemple, pour la question « Qui est dans la maison », l'élément inconnu est l'instance qui jouera le rôle de cible dans une DCAS où se trouveraient une vdv 'vrai', une instance 'site' du concept 'maison' et une relation (jouant le rôle de prédicat) 'dans'.

A la question « Où est Nicolas ? », il y a deux inconnues : une inconnue jouant le rôle de site et une autre de type relation jouant le rôle de prédicat.

Exemple : A la question 'Le chat est-il dans la maison ?', correspond la connaissance suivante :

```
question(h1, q2, 'Le chat est-il dans la maison ?',

     /* localisation de la ressource sonore associée */
     [ [path/"DD Amical:Base de ressources:H1",
        fichier/questions,type/ressource,
         typeRessource/'snd',idNom/q2]],
     /* inconnues */
     [],

    /* contexte temporel */
     RT1,RT2,
     /* principal */
     [[[instance,cible,*,chat],[relation,*,*,dans],
       [instance,site,*,maison],
       [rt,début,*,RT1],[rt,fin,*,RT2],
       [vdv,*,*,vrai],[*,*,*,*]]
     ],

     /* secondaire */
   [[[instance,dénoté,*,maison],[relation,*,*,est-un],
     [concept,dénoteur,*,maison],[vdv,*,*,vrai], [*,*,*,*]],
    [[instance,dénoté,*,chat],[relation,*,*,est-un],
     [concept,dénoteur,*,chat],[vdv,*,*,vrai], [*,*,*,*]],
    [[instance,possédé,*,maison],[instance,possesseur,*,nicolas],
     [rt,début,*,rtdébuth1],[rt,fin,*,rtfinth1],
   [vdv,*,*,vrai],[*,*,*,*]]
    ]).
```

A la question 'Le chat sera-t-il dans la maison ?', nous associerons la donnée suivante :

```
question(h1, q2futur,  'Le chat sera-t-il dans la maison ?',
     /* localisation de la ressource sonore associée */
     [ [path/"DD Amical:Base de ressources:H1",
        fichier/questions,
        type/ressource,typeRessource/'snd ',idNom/q2f]],

     /* inconnues */
     [],
    /* contexte temporel */
     RT1p,RT2p,

     /* principal */
     [[[instance,cible,*,chat],[relation,*,*,dans],
       [instance,site,*,maison],
       [rt,début,*,RT1],[rt,fin,*,RT2],
       [vdv,*,*,vrai],[*,*,*,*]]
     ],

     /* secondaire */
    [[[instance,dénoté,*,maison],[relation,*,*,est-un],
      [concept,dénoteur,*,maison], [vdv,*,*,vrai], [*,*,*,*]],
     [[instance,dénoté,*,chat],[relation,*,*,est-un],
      [concept,dénoteur,*,chat],[vdv,*,*,vrai], [*,*,*,*]],
     [[rt,sujet,*,RT1],[relation,prédicat,*,>],
      [rt,valeur_référence,*,RT2p],[vdv,*,*,vrai], [*,*,*,*]],
     [[instance,possédé,*,maison],[instance,possesseur,*,nicolas],
      [rt,début,*,rtdébuth1],[rt,fin,*,rtfinth1],
      [vdv,*,*,vrai],[*,*,*,*]]
     ]).
```

Note :

Les DCAS associées aux questions seront aussi introduites par un opérateur humain.

La perspective de fournir une interface graphique/texte pour la génération de ces DCAS peut aussi être envisagée.

3.3.4. Base concernant l'apprenant (BA)

Le modèle restreint de l'apprenant BA contient principalement des références liées aux phrases-textes déjà lus et les questions-réponses déjà donnés par l'apprenant. Elles sont utilisées par SYSQUEST lors de l'établissement de la liste à délivrer.

3.3.5. Base de connaissances diverses (BD)

SYSQUEST doit connaître la provenance de certaines descriptions à l'intérieur de la base BP. Certaines descriptions peuvent provenir de connaissances diverses générales du monde. D'autres proviennent plus directement des textes qui ont été présentés à l'enfant. Ce sont ces dernières connaissances qui nous intéressent car les questions qui seront choisies seront celles qui demanderont de la part de l'enfant l'utilisation des connaissances véhiculées par les descriptions associées au texte.

3.4. Traitement des flux par SYSQUEST

3.4.1. Réception d'une information concernant l'introduction d'un texte

SYSQUEST reçoit la référence du texte présenté ti (ex : t1) , ainsi que la référence de l'histoire hi (ex : h1) à laquelle ce texte appartient.

Il y a cinq étapes dans le traitement de cette information.

❶ Le système vérifie tout d'abord que cette histoire hi est une histoire dont il connaît les instants débuts et fins. Il cherchera dans ses connaissances le terme suivant :

debutEtFinHistoire(hi,RT_DEBUT,RT_FIN)

S'il ne trouve pas ce terme, il cherchera dans la base BH une connaissance de la forme :

histoire(hi, 'Histoire de ...',
[[[rt,début,,<rtdébuth1>],[rt,fin,*,<rtfinh1>]]*
]).

et crée ensuite le terme "**debutEtFinHistoire(hi, <rtdébuth1>, <rtfinh1>)**".

❷ Le système vérifie que les informations sémantiques liées au texte ti n'ont pas déjà été introduites. Cette situation apparaît dans le cas où il y a relecture de ce texte.

Pour cela, il cherchera dans BA, la présence du terme suivant "*descriptionsAssociéesIntroduitesDansBP(ti,hi)*".

Si cette information n'existe pas dans la base, le processus se poursuit en ❸ après avoir introduit "**descriptionsAssociéesIntroduitesDansBP(ti,hi)**" dans BA.

Si cette information existe, le système n'a pas besoin d'introduire les descriptions associées au texte indiqué. Il peut éventuellement incrémenter le nombre de fois que ce texte a été lue si cette information est nécessaire. Le processus d'introduction est terminé.

❸ A partir de ti et hi, SYSQUEST recherchera dans BH les descriptions DCAS associées au texte ti de l'histoire hi.

Il trouvera dans la base une connaissance sous la forme suivante :

texte(<hi>, <ti>, <EDP>, <EDS>, <INFO_SUR_VARIABLE>,
<RT_DEBUT_OV > , <RT_FIN_OV >).

Toutes les descriptions de EDP doivent être introduites dans le système de représentation. Lorsque une description est effectivement incorporée dans la base de connaissances, un numéro unique d'identification de la description est retournée par la procédure d'introduction. SYSQUEST possède alors la liste des numéros de référence des descriptions effectivement introduites (LNRDI) pour le texte ti.

Cette information sera stockée dans la base BD sous la forme suivante :

listeDescriptionsAssociees(hi,ti, <LNRDI>).

❹ Toutes les descriptions de EDS qui peuvent être introduites le sont. Ici, le système ne garde pas la liste des numéros de toutes les descriptions effectivement introduites.

❺ SYSQUEST mémorise le couple de repères temporels (<RT_DEBUT_OV> , <RT_FIN_OV >) trouvé dans le dernier texte ti introduit.

Pour cela, il crée dans BD l'information

repèresObservateur(<RT_DEBUT_OV> , <RT_FIN_OV >).

et supprime les anciennes informations de cette forme.

3.4.2. Réception d'une information concernant une question de compréhension posée

SYSQUEST reçoit :

- la référence de la question présentée (qi)
- la référence de l'histoire à laquelle cette question appartient (hi)
- la réponse attendue (ra)
- la réponse donnée (rd).
- la liste des texte utilisés pour trouver la réponse (ltur)

Le système procède à l'introduction dans la base de connaissances de l'élève (BA) de l'information suivante :

interaction(hi,qi,ra,rd,ltur).

Cette information est utilisée dans le traitement d'une requête.

3.4.3. Traitement d'une requête

Si l'on formulait en langage naturel la requête reçue par SYSQUEST, elle s'énoncerait comme suit :

<< A partir de la situation actuelle de l'apprenant, indiques-moi quelles questions de compréhension choisir pour l'histoire hi?>>

Le système devrait donner une liste de questions avec bien sûr les réponses attendues.
Un type de réponse donnée serait :

<< Pour l'histoire Hi, la liste de couples (<question de compréhension à pôser>, <réponse attendue>, <liste des textes utilisés pour prépondre à la question>) est .... >>

Pour répondre à cette requête, deux étapes sont nécessaires.

❶ Le système sélectionne toutes les questions qi de hi qui auront un réponse de type oui/non/jpps. Pour cela, il ne sélectionne que les questions dont le champ INCONNUES est réduit à un ensemble vide.

❷ Pour chaque question qi présente dans cette liste, le système devra fournir la réponse attendue oui/non/jpps ainsi que le chemin de descriptions utilisée pour résoudre cette question.

<u>Détail de la résolution d'une question.</u>
Le système possède l'information

"repèresObservateur(RT_DEBUT_OV , RT_FIN_OV)".

Pour chaque question de BQ de la forme

question(REF_HIST, REF_QUEST, TEXTE_QUEST,
LOC_RESSOURCE,
CTXT_TEMP1, CTXT_TEMP2,
LIST_ELEM_INCONNUS_DS_QUEST,
EDP,
EDS)

le système unifiera tout d'abord les variables CTXT_TEMP1 et CTXT_TEMP2 à RT_DEBUT_OV et RT_FIN_OV.

Le système crée une nouvelle liste EDP_EDS en concaténant EDP et EDS.

SYSQUEST prend ensuite la première DCAS présente dans EDP_EDS et lance la recherche de cette DCAS en l'utilisant comme description référence. Lorsqu'il réussit à trouver l'information, le système prend la seconde DCAS de EDP_EDS et lance une nouvelle recherche avec une nouvelle description référence. Si cette dernière recherche aboutit avec succès, il passe à la DCAS suivante dans EDP_EDS jusqu'à épuisement de cette liste. Si la recherche n'aboutit pas pour une DCAS de EDP, le système demande un autre résultat pour la DCAS précédente. Lorsque toutes les DCAS de EDP_EDS ont été utilisées comme description référence avec succès, le système peut alors déclarer que la réponse à la question REF_QUEST de l'histoire REF_HIST est **vrai**. S'il y a échec dans la tentative de trouver des informations vérifiant toutes les DCAS de EDP_EDS, le système crée d'abord une seconde liste EDP_EDS_2 en affectant à ce dernier la liste EDP_EDS, puis change dans la première description de EDP_EDS_2 le quadruplet (vdv,*,*, X) en un quadruplet (vdv,*,*, Y) avec X opposé à Y. Le système relance le processus de recherche sur cette nouvelle donnée. Si une réponse est trouvée, alors le résultat de la question est **faux**. Si cette dernière recherche n'aboutit pas, alors le résultat de la question est considéré comme **je-ne-peux-pas-savoir**.

Lors de la recherche, lorsque la réponse déduite à la question est vrai ou faux, le système a obtenu en retour l'ensemble des numéros des descriptions qui ont permis de répondre aux multiples interrogations de la base.

A l'issue de cette étape, nous obtenons une liste L1 de triplets (<question>, <réponse attendue>, <chemin en numéros des descriptions>)

❸ Une autre liste L2 est constituée à partir de la liste L1 en transformant chacun des triplets de L1 en un triplet (<question>, <réponse attendue>, <liste des textes ti nécessaires à la résolution>)

❹ A partir de cet instant, une série de filtrages est enclenchée par le système.

Filtrage 1 : (Elimination des questions ne faisant pas appel aux connaissances sur les textes).

Tous les triplets de L2 dont le 3ème uplet est égal à une liste vide sont éliminés. Ces questions ne font vraisemblablement pas appel à des connaissances issues des textes présentés.

Filtrage 2 :(Elimination des questions déjà posées et qui ont reçu une réponse correcte)

Si le nombre de questions reste supérieur à 4, les triplets (QI, RA1, LTEXTES) vérifiant

les conditions ci-dessous sont supprimés de la liste.

- il existe dans BA l'information "intéraction(HI,QI,RA2,RD)".
- RA2 = RA1
- RA = RD (l'élève a répondu correctement à la question)

Filtrage 3 : (Elimination des questions faisant appel à des textes moins récents)

Soient :

- LTL1, la liste de tous les textes qui ont été lus
- LTU2 la liste des textes utilisés dans toutes les questions qui ont été effectivement posées. Elle se calcule à partir des informations présentes dans BA. C'est l'union de toutes les valeurs possibles de "ltur" des informations "intéraction(hi,qi,ra,rd,ltur)" trouvées dans BA.
- LTLPR, la liste de tous les textes les plus récents.

Ici, LTLPR se calcule en prenant le complémentaire de LTU2 dans LTL1.

Si LTLPR est vide, cela signifie que l'on redemande un autre choix de questions et qu'aucun texte nouveau n'a été présenté.

Tant que le nombre de triplets de L2 reste supérieur à 4, SYSQUEST enlève les questions qui n'exploitent aucun texte de LTLPR.

Conclusion

A travers la recherche de solutions pour une problématique de représentation de connaissances à l'intérieur d'un environnement informatique d'aide à l'apprentissage, nous avons pu voir que la plupart des structures proposées par quelques systèmes de représentation ne pouvaient convenir.

Cela nous a conduit à élaborer un système de représentation, qui tout en tenant compte des points étudiés, reste relativement général pour décrire d'autres types de connaissances. Même si l'application de ce formalisme a été restreinte à une problématique bien précise, il semble que son utilisation puisse être élargie dans Amical. On pourra envisager, l'utilisation de la représentation sémantique de l'histoire, pour l'analyse de réponses d'un apprenant. D'autres applications sont possibles : cela va de la modélisation de la connaissance (pas seulement de l'histoire) de l'apprenant à la représentation des connaissances du système sur les ressources multimédia possédées.

Il semble que l'un des principaux atouts de ce système résulte du fait que l'utilisateur d'une base de DCAS donnée peut utiliser des formes partielles à partir desquelles le système fera des recherches et inférera de nouvelles informations. Pour rechercher une information sur quelqu'un ou quelque chose, il suffit de connaître uniquement l'élément cognitif qui lui a été associé (c-à-d le type et le référent) et de fournir au système une DCAS incomplète contenant un quadruplet possédant ce même type et même référent. Pour filtrer les informations, l'utilisateur peut préciser la contribution de cet élément au sens présent dans l'information recherchée ainsi que les quadruplets connus appartenant à cette même DCAS .

Plusieurs directions de travail peuvent être envisagées.

Suivant la direction liée à la problématique SYSQUEST, il serait intéressant de fournir à la personne introduisant les représentations sémantiques associées aux textes et questions, une aide à leur création/mise à jour.

Suivant l'angle d'implémentation, il serait intéressant de réaliser des optimisations de procédure et de recherche, surtout lorsque peu d'informations sont précisées dans la DCAS référence.

Du point de vue utilisateur de ce formalisme dans n'importe quel problématique donnée, il serait intéressant d'étudier sous l'aspect traduction/génération des DCAS vers le

langage naturel. La recherche de régularités dans les DCAS permettrait d'enrichir les bases de DCAS.

Le formalisme présenté peut être prolongé suivant plusieurs orientations :

1) les possibilités d'extensions vers un raisonnement par défaut ([Reiter, 1978b]).
Il faudra voir quelles sont les répercussions sur le mécanisme d'inférence du système.

2) l'utilisation de DCAS pour avoir un méta-contrôle du raisonnement.
Pourrait-on concevoir des DCAS de type méta-règle qui fourniraient un moyen de rendre plus souple le contrôle du raisonnement ?

3) Définition de nouvelles entités cognitives simples ou complexes.
Des quadruplets complexes de type scénario pourraient être étudiés.
Actuellement, le système dispose de deux moyens de quantifier de manière universelle ses données. Il s'agit des censembles et des DCAS contenant les concepts subsumeur/subsumé. On peut envisager les quantifications au niveau du temps. Y-aurait-il des CRepereTemporel ?

Bibliographie

[Allen, 1981] James F. Allen
"An Interval-Based Representation of Temporal Knowledge",
Proc. 7th IJCAI, Vancouver, PP. 221-226.

[Allen, 1983] James F. Allen
"Maintaining Knowledge About Temporal Intervals",
Com. ACM, vol. 26, n°11, pp. 832-843.

[Allen, 1984] James F. Allen
"Towards a General Theory of Action and Time",
Artificial Intelligence, vol. 23, N°2, pp. 123-154.

[Barwise & Perry, 1983] John Barwise & John Perry
Situations and Attitudes, MIT Press, Cambridge, MA.

[Bennett et al., 1978] Bennett J., Creary L., Engelmore R., and Melosh R.
"SACON : A knowledge-based consultant for structural analysis",
STAN-CS-78-699 and HPP Memo 78-23, Stanford University

[Biebow & al., 1987] Biebow B. , Canet B. , Castaing J. ,Coupey P. , Szulman S. ,
Teulat D. & Zegel F. (LIPN)
"Enrichissement de Réseau Sémantique à partir de connaissances exprimées en Langage Naturel",
6 ème Congrès Exposition Reconnaissance des Formes et Intelligence Artificielle, Dunod Informatique, Tome 2, 16-20 Nov. 1987, ANTIBES, p. 637-751.

[Bobrow & Winograd, 1977] Bobrow D.G. & Winograd T.
"An overview of KRL, a knowledge representation language",
Cognitive Science, 1977, vol. 1, n°1, p. 3-46.

[Bonnet, 1984] Alain Bonnet
L'intelligence artificielle . Promesses et Réalités,
Inter Editions, 1984, Paris.

[Boons, 1987] J.-P. Boons
"La notion sémantique de déplacement dans une classification syntaxique des verbes locatifs", *Langue Française*, n°76, Décembre 1987, p.5-40

[Borillo & Vieu, 1989] Mario Borillo, Laure Vieu
"Eléments pour la formalisation du raisonnment spatial",
RFIA 1989, p 695-709.

[Borillo, 1990] Andrée Borillo
"A propos de la localisation spatiale",
Langue Française, n°86, Mai 1990, p.75-84.

[Brachman, 1977] Brachman R.J
"What's in a concept : Structural Foundations for Semantic Networks" ,
International Journal of Man-Machine Studies, 1977, 9,127-152.

[Brachman, 1979] Brachman R.J
"On the Epistemological Status of Semantic Networks",
Associative Networks : Representation and use of knowledge by computers,
Findler, New York, Academic Press, 1979

[Brachman, 1983] Brachman R.J
"What ISA Is and Isn't : An analysis of taxonomic links in Semantic Networks",
IEEE Computer, vol. 16, n° 10, p. 30-36.

[Brachman, Fikes & Levesque,1983] Brachman R.J., Fikes R.E. & LevesqueH.J
"KRYPTON : A functional approach to knowledge representation",
IEEE Computer, Octobre 1983, vol. 16, n° 10, p. 67-73.
aussi dans
AI Technical Report n° 16, Mai 1983.

[Brachman, Gilbert & Levesque,1985] Brachman R.J. , Gilbert V.P. & Levesque H.J.
"An essential Hybrid Reasoning System : Knowledge and Symbol Level Accounts of KRYPTON", *IJCAI 85*.

[Brachman & Schmolze, 1985] Brachman R.J. & SchmolzeJ.G.
"An overview of the KL-ONE knowledge representation system",
Cognitive Science, vol. 9, n°2, p. 171-216.

[Brachman & al., 1991] Ronald J. Brachman, Deborah L. McGuinness,
Peter F. Patel-Schneider, Lori Alperin Resnick, Alexander Borgida
"Living with Classic : When and How to Use a KL-ONE-Like Language.",
Principles of Semantic Networks : Explorations in the representation of knowledge,
Ed. John F. Sowa, Morgan Kaufmann, 1990-1, p401-456.

[Bras, 1990] Myriam Bras
Calcul des structures temporelles du discours
Thèse de l'Université de Toulouse, 1990.

[Brooks, 1991a] Rodney A. Brooks
"Intelligence without Representation",
Artificial Intelligence, 47, 1991, p. 139-160.

[Catach, 1988] Nina Catach,
"Fonctionnement linguistique et apprentissage de la lecture",
Langue Française, Décembre 1988, N°80, pp. 6-19.

[Cercone & McCalla, 1987] Nick Cercone & Gordon McCalla,
"What is Knowledge Representation",
The knowledge Frontier : Essays in the Representation of Knowledge,
ed. Nick Cercone, Gordon McCalla, Springer Verlag, 1987
basé sur l'article,
Approaches to Knowledge Representation, G. McCalla and N. Cercone, apparu dans *COMPUTER*, Volume 16, Number 10, October, 1983.

[Chambreuil , 1990] Chambreuil A. , Chambreuil M.
"Advanced Information Technologies : Prospect for Learning to Read."
EURIT'90, Herning, Denmark.

[Chambreuil, 1991] Chambreuil A., Chambreuil M., Chanier T., Lotin P. & Néhémie P. "Cognitive Science, Artificial Intelligence, New Technologies: How to Cooperate for a Computer-Assisted Learning to Read System",
The International Conference on the Learning Sciences, Evanston, pp 74-82.

[Chambreuil & Cherkaoui, 1993] Chambreuil A., Cherkaoui C.
"Résolution de problèmes d'enseignement dans un environnement d'aide à l'apprentissage de la lecture."

Actes du colloque SCIAL'93, Clermont-Ferrand, France, pp 43-53

[Chambreuil & al., 1994] Chambreuil A., Chambreuil M., Cherkaoui C.
Individualization within a multi-agent computer-assisted learning-to-read environment. *Journal of Artificial Intelligence in Education* (à paraître)

[Cherkaoui, 1991] Cherkaoui C.
Etude et développement d'un dictionnaire didactique dans un environnement d'aide à l'apprentissage de la lecture,
.Rapport de DEA de Linguistique et Informatique, Université Clermont 2.

[Cherkaoui & Néhémie, 1993] Cherkaoui C., Néhémie P.
"Planification dynamique dans environnement Multi-agents d'aide à l'apprentissage de la lecture",
Actes des 2ème Journées de Volcans-IA. Clermont-Ferrand.

[Clancey, 1979] William J. Clancey
"Tutoring rules for guiding a case method dialogue",
The International Journal of Man-Machine Studies, 11:25-49
aussi dans
Intelligent Tutoring Systems, Ed. Sleeman and Brown, New York,
Academic Press 1982

[Clancey, 1985] William J. Clancey
"Heuristic Classification"
Artificial Intelligence, 27:289-350

[Clark, 1978] K.L. Clark
"Negation as Failure",
Logic ansd Data Bases, H. Gallaire and J. Minker, Eds.,Plenum Press, N.Y.,
1978, 293-322.

[Clocksin & Mellish, 1981] Clocksin W.F. & Mellish C.S.
Programming in Prolog,
Springer-Verlag,Berlin Heidelberg,New York.

[Condillac, 1986] Condillac M.
Prolog : Fondements et applications,
Dunod Informatique, Paris.

[Cummings & Self, 1985] Geoff Cummings, John Self
"Collaborative Intelligent Educational Systems",
Artificial Intelligence and Education, p73-85, Amsterdam, 1985

[Clancey, 1987] William J. Clancey
Knowledge-Based Tutoring : The GUIDON Program,
The MIT Press, 379 p., 1987

[CRIN, 1992]
Troisième rapport d'activités sur les systèmes multi-agents,
CRIN/INRIA-LORRAINE - Equipe RFIA, Mai 1992

[Danlos, 1984] Danlos,
Génération automatique de textes en langues naturelles,
Thèse de doctorat ès sciences, Paris VII

[Devanbu & Litman, 1991] Premkumar T. Devanbu, Diane J. Litman
"Plan-Based Terminological Reasoning",
KR91 "Principles of Knowledge Representation and Reasoning", Proceedings of the Second International Conference, Cambridge, Massachusetts,
p128-138, Avril 1991

[Erman & al., 1980] Erman L.D., Hayes-Roth F., Lesser V. R., Reddy D.R.
"The Hearsay-II Speech Understanding System : Integrating kowledge to resolve uncertainty",
Computing Surveys, vol.12, n°2

[Fargues, 1989]
"Graphes conceptuels et compréhension du langage naturel",
Semantica 1989, Paris.

[Fillmore, 1968] Charles Fillmore
"The case for case",
Universals in Linguistic theory, Bach & Harms, Chicago, Holt, Rinehart and WInston, pp. 1-90.

[Frost, 1986] Frost R.A..
Intoduction to Knowledge Base Systems, Collins, London.

[Genesereth & Nilsson, 1987] Michael R. Genesereth & Nils J. Nilsson,
Logical Foundations of Artificial Intelligence,
ed. Michael B. Morgan, Morgan Kaufmann Publishers.

[Goldman, 1974] Goldman, N.
Computer Generation of natural language from a deep conceptual base,
Ph. D. Thesis, Computer Science Department, Standford University,
Standford, California,

[Haton & al, 1978] Haton J.P., Messenet G., Pierrel J.M., Sanchez C.
"La chaîne de compréhension parlée du système MYRTILLE-II",
Congrès AFCET-TTI, Paris

[Haton J.P& al., 1991]Haton, Jean Paul., Bouzid Nadjet, Charpillet François,
Haton Marie-Christine, Lâasri Brigitte, Lâasri Hassan,
Marquis Pierre, Mondot Thierry, Napoli Amadeo
Le raisonnement en intelligence artificielle : Modèles, techniques et architectures pour les systèmes à base de connaissances,
InterEditions, Paris.

[Hendrix, 1979] Gary G. Hendrix
"Encoding Knowledge in Partioned Networks",
Associative Networks : Representation and use of knowledge by computers,
Findler, New York, Academic Press.

[Jouvet, 1987] Denis Jouvet
"Reconnaissance de mots connectés indépendamment du locuteur par des méthodes statistiques",
RFIA 1987, Tome 1, Antibes, p 65-72

[Kobsa, 1991] Alfred Kobsa 1991,
"Utilizing Knowledge : The components of the SB-ONE Knowledge Representation Workbench.",
Principles of Semantic Networks : Explorations in the representation of knowledge,
Ed. John F. Sowa, Morgan Kaufmann, 1990-1, p457-486.

[Kunz et al., 1979] Kunz, J. C., Fallat, R. J., McClung, D. H., Osborn, J. J., Votteri, B. A., Nii, H. P., Aikins, J., Fagan, L., and Feigenbaum, E.
"A physiological rule-based system for interpreting pulmonary fonction test results. ",
Proceedings of the Computers in Critical Care and Pulmonary Medecine,
pages 375-379, IEEE Press

[Laasri & Maitre, 1989] Hassan Laasri & Brigitte Maitre
Coopération dans un univers multi-agents basée sur le modèle du blackboard : etudes et réalisations,
Thèse de l'Université de Nancy I, Février 1989

[Lafont , 1989] Christine Lafont
Systèmes tutoriels intelligents et Apprentissage de la lecture. Architectures tableaux noirs. Planification,
Rapport de Stage de DEA, Laboratoire d'Informatique,
Universté Blaise Pascal, Clermont 2.

[Lamiroy, 1987] J.-B. Lamiroy
"Les verbes de mouvement emplois figurés et extensions métaphoriques",
Langue Française, n°76, Décembre 1987, p.41-58.

[Langacker, 1986a] Langacker, Ronald W.
"An introduction in Cognitive Grammar
Cognitive Science , Vol 10, N°1, p. 1-40.

[Langacker, 1986b] Langacker, Ronald W.
Foundations of cognitive Grammar, vol. 1, Stanford University Press, Stanford

[Laurière, 1986] Laurière J.L.
Intelligence Artificielle : Résolution de problèmes par l'Homme et la Machine,
Editions Eyrolles, Paris.

[Levesque, 1984] Levesque H.J.
"Foundations of a Functionnal Approach to Knowledge Representation",
Artificial Intelligence, Juillet 1984, 23(2), p. 155-212.

[Mac Gregor, 1991] Robert Mac Gregor
"The evolving technology of classification-based knowledge representation systems.",
Principles of Semantic Networks : Explorations in the representation of knowledge,
Ed. John F. Sowa, Morgan Kaufmann, 1990-1, p. 385-400.

[Mahmoud & Chambreuil, 1995] Mahmoud Ali & Chambreuil Michel
"Architecture multi-agents pour la conception des Interfaces Apprenant-Machine",
Environnements Interactifs d'Apprentissage avec ordinateur, Tome 2,
Eyrolles, Paris, 1995, p. 291-302

[Maida & Shapiro, 1982] Anthony S. Maida and Stuart C. Shapiro,
"Intensional Concepts in Propositional Semantic Networks,"
Cognitive Science 6(4), p. 291-330
aussi dans
Readings in Knowledge Representation, Morgan Kaufmann, 1985, p169

[McDermott, 1982] D. McDermott
"Temporal Logic for Reasoning about Processes and Plans",
Cognitive Science, vol. 6, pp. 101-155.

[McDermott, 1985] McDermott, D.
"Reasoning about Plans",
Formal Theories of the Commonsense World,
Hobbs and Moore (eds), Ablex, Norwood, pp. 269-319.

[Moser, 1983] M.G. Moser
"An overview of NIKL, the New Implementation of KL-One",
BBN Annual Report, BBN Rep. N°5421, pp 27-39

[Néhémie, 1992] Néhémie P.
"A systemic approach for student modelling in a multi-agent aided learning environment. ",
Intelligent Tutoring Systems: ITS 92., Frasson C., Gauthier G. and McCalla G.I., (eds), Lecture Notes in Computer Science, Springer-Verlag, Heidelberg, pp. 475-482.

[Nicaud, 1987] J.F. Nicaud
Aplusix : un système expert de résolution pédagogique d'exercices d'algèbre,
Thèse de l'Université Paris XI, Orsay, Décembre 1987.

[Nicaud, 1988] J.F. Nicaud
"Aplusix : un environnement d'apprentissage de raisonnement algébrique"
Applica 88 (Intelligence Artificielle et formation", Lille 1988.

[Patel-Schneider & al., 1984] Peter F. Patel-Schneider, R.J. Brachman, H.J. Levesque
"Argon : Knowledge Representation meets Information Retrieval",
Proc. First Conference on Artificial Intelligence Applications, Denver, Décembre 1984, pp 280-286

[Pujet, 1989] Jean-François Puget
"Représentation explicite de la négation d'un concept",
RFIA 1989, Tome 2, Paris, p 869-883

[Rastier & al., 1987] Rastier F. & al.
"Sémantique et Intelligence Artificielle",
LANGAGES, Septembre 1987, Larousse, France.

[Reiter, 1978a] R. Reiter
"Closed World Assumption",
Logic ansd Data Bases, H. Gallaire and J. Minker, Eds., Plenum Press, N.Y., 55-76.

[Reiter, 1978b] R. Reiter

Readings in Knowledge Representation,
R.J. Brachman and H.J. Levesque (eds.), Los Altos, CA: Morgan Kaufmann Publishers, Inc., 1985, pp. 401-410

[Rich, 1983] Rich E. ,
Intelligence Artificielle,
Masson, Paris, 1987, trad. de "*Artificial Intelligence*", McGraw-Hill, 1983.

[Sabah, 1988] Sabah G.
L'intelligence artificielle et le langage : représentation des connaissances (volume 1),
HERMES, Paris.

[Sabah, 1988] Sabah G.
L'intelligence artificielle et le langage : Processus de comprehension (volume 2),
HERMES, Paris.

[Schank, 1975] Roger C. Schank
Conceptual Information Processing,
Fundamental Studies in Computer Science, Volume 3,
north-holland/american elsevier.

[Schank & Riesbeck, 1981] Roger C. Schank & Christopher K. Riesbeck
Inside Computer Understanding,
Lawrence Erlbaum Associates, Hillsdale, New Jersey.

[Schmolze & Lipkis, 1983] Schmolze J.G. & Lipkis,
"Classification in the KL-ONE Knowledge Representation System",
IJCAI-83, , pp. 330-332.

[Schubert, L., Goebel, R., Cercone, N. , 1979] Lenhart K. Schubert, Randolph G. Goebel, and Nicholas J. Cercone
"The Structure and Organisation of a Semantic Net for Comprehension and Inference",
Associative Networks: Representation and and Use of Knowledge by Computers,
N.V. Findler (ed.), Academic Press.

[Setre, 1989] Isabelle Setre
Modélisation de l'apprenant et Croyances dans un système d'aide à l'apprentissage de la lecture,
Rapport de Stage de DEA, Laboratoire d'Informatique,
Universté Blaise Pascal, Clermont II.

[Shortliffe, 1974] E. H. Shortliffe
MYCIN : A rule-based computer program for advising physicians regarding antimicrobial therapy selection.
Ph.D. dissertation in Medical Information Sciences, Standford University.
aussi dans
Computer-based Medical Consultations : MYCIN
New York : American Elsevier, 1976.

[Sowa, 1984] John F. Sowa
Conceptual Structures. Innformation Processing in Mind and Machine,
éditions Addison Wesley, REading, MA.

[Tarby, 1989] Jean-Claude Tarby
Dictionnaire didactique", Rapport de Stage de DEA, Laboratoire d'Informatique,
Universté Blaise Pascal, Clermont 2.

[Vandeloise, 1986] Claude Vandeloise
L'espace en Français : Sémantique des prépositions spatiales.
dans la collection "Travaux linguistiques" dirigée par Nicolas Ruwet,
Editions du Seuil, Octobre 1986.

[van Melle , 1979] van Melle W.
"A Domain-Independant Production Rule System for Consultation Programs",
Proceedings of the Sixth International Joint Conference on Artificial Intelligence, Los Angeles, California : William Kaufmann, Inc., p.923-925

[Vieu, 1991] Laure Vieu
Sémantique des relations spatiales et inférences spatio-temporelles : une contribution à l'étude des structures formelles de l'espace en langage naturel,
Thèse de l'Université Paul Sabatier

[Vilain, 1985] Vilain M.
"The Restricted Language Architecture of a Hybrid Representation System",
Proc. of IJCAI-85, p. 547-551.

[Von Luck, Nebel, Petalson & Schmiedel, 1986] Von Luck K., Nebel B., Petalson C., Schmiedel A.
"The anatomy of the BACK-SYSTEM",
Rapport interne Projet KIT-BACK,
TU-Berlin, Novembre 1986.

[Voyer, 1987] Robert Voyer
Moteurs de systèmes experts,
Editions Eyrolles.

[Wenger, 1987] E. Wenger
Artificial Intelligence and Tutoring Systems.,
Los Altos, Morgan Kaufmann.

[Wilensky, 1991a] Robert Wilensky
"The Ontology and Representation of Situations",
KR91 "Principles of Knowledge Representation and Reasoning", Proceedings of the Second International Conference, Cambridge, Massachusetts, p.558-569, Avril 1991.

[Wilensky, 1991b] Robert Wilensky
"Sentences, situations, and propositions.",
Principles of Semantic Networks : Explorations in the representation of knowledge,
Ed. John F. Sowa, Morgan Kaufmann, 1990-1, p. 191-227.

[Woods, 1975] W.A.Woods
"What 's in a link : Foundations for Semantic Networks",
Representations and Understanding : Studies in Cognitive Science,
Bobrow D.G. et Collins A.M., Academic Press, New York, p. 35-82.

[Woolf, 1987] Bevely P.Woolf,
"Representing complex knowledge in an intelligent machine tutor",
Computational Intelligence Vol.3, Number 1, Février 87, p. 45.
aussi dans
Artificial Intelligence and Human Learning,
éd. John Self, Chapman and Hall, 1988.

Annexes

Annexe 1 :
Regroupement de verbes suivant la dominance d'un trait de caractérisation ([Borillo A., 1990])

- nature non animé /humain ou animal : être placé, être rangé, gésir, gîter, habiter, nicher, percher, reposer, résider, séjourner, se tenir, vivre, trôner, figurer, régner

ex : un oiseau niche sur le toit

- nombre : être éparpillé, être dispersé, s'empiler, grouiller, foisonner, s'échelonner, se presser, pulluler, fourmiller, se répartir ;

ex : Des assiettes sales s'empilent dans l'évier.

- dimension horizontale, verticale, volume : s'étendre, s'élever, se dresser, s'étaler, reposer, être accroché, être répandu, pendre, aller (dans), contenir (des), tenir (dans)

ex : Une statue se dresse au milieu du square

- persistance dans l'état : rester, demeurer, croupir, stagner, siéger, stationner, traîner, planer, régner ;

ex : les livres traînent sur la table

- posture : être debout, être couché, être étendu, être à plat, être rangé, être posé, être adossé, être appuyé,

ex : Le chat est couché sur le tapis.

- consistance (de la cible ou du site) : baigner (dans), croupir, flotter, mariner, nager, stagner, tremper (dans)

ex : Une odeur de gaz stagne dans l'appartement.

Annexe 2 :
Classement des inférences dans la dépendance conceptuelle de Schank

Notion de Classification d'inférences.

Bien que les inférences possibles possèdent des caractéristiques communes, leurs utilités dans le cours du traitement les distinguent. Il est donc intéressant de distinguer ces inférences par type.

1) SPECIFICATION (*Specification Inferences*)

Elles permettent la recherche d'informations non formulées de manière explicite. Dans "John prit un caillou. Il frappa Bill.", il n'est pas dit (explicitement) que John a frappé Bill avec un caillou.

2) CAUSE (*Causative or Causal Inferences*)

On recherche les causes possibles d'une action ou d'un état exprimé. Si on a "John frappa Mary avec un caillou", on peut supposer que John était en colère contre Mary.

3) RESULTAT (*Resultative Inferences*)

Les effets possibles d'une action ou les conséquences d'un état peuvent être recherchés. Pour "Mary offrit à John une voiture", le résultat est la possession par John de la voiture.

4) MOTIVATION (*Motivational Inferences*)

Recherche des intentions (motivations) de certains acteurs ?

Si "John frappa Mary", c'est qu'il avait peut-être l'intention de la blesser.

5) POSSIBILITE (*Enablement Inferences*)

Elles permettraient de donner une idée de l'état du monde de référence qui a permis une action donnée.

Si "Peter alla en Europe", on peut se demander comment il a obtenu l'argent ? Quand on dit que "Jean frappe Paul", on peut normalement supposer qu'ils sont près l'un de l'autre.

6) FONCTION (*Function Inferences*)

On utilise la fonctionnalité des objets mis en œuvre. Si "John veut un livre", ce serait probablement pour le lire ou pour l'offrir ?

7) PREDICTION (*Enablement-Prediction Inferences*)

Un personnage peut désirer un état du monde pour arriver à réaliser une action que cet état lui permet. Pour "John chercha la recette de la tarte aux pommes", il est probable que John ait envie d'en faire une.

8) MANQUE (*Missing Enablement Inferences*)

Une enquête sur ce qui n'a pu permettre certaines actions peut être réalisée. "Mary ne pouvait voir l'arrivée des chevaux. Elle maudit l'homme en face d'elle." Ici, l'homme avait gêné la vue de Mary.

9) PREVENTION (*Intervention Inferences*)

Certaines actions sont réalisées dans l'intention d'éviter l'occurence d'autres actions. "Le bébé courait dans la rue. Mary courut après lui." Mary voulait éviter au bébé de se blesser.

10) ACTION (*Action Prediction Inferences*)

Certaines actions seraient logiquement liées à la réalisation des désirs d'un personnage. "John voulait des clous." Il ira probablement à la quincaillerie.

11) Propagation d'Information (*Knowledge Propagation Inferences*)

Certaines actions entraînent la propagation d'informations. "Peter dit à Bill que Mary a frappé John avec une batte." Bill sut que John a été blessé.

12) NORMALITE (*Normative Inferences*)

Il est intéressant de se pencher sur les causes qui pourraient expliquer un éventuel écart entre la situation réelle décrite et la situation normale. "John vit Mary à la plage mardi matin". Pourquoi n'était-elle pas au travail ?

13) DUREE (*State Duration Inferences*)

On peut se poser des questions sur la durée d'un état.

"John a prêté, il y a longtemps, un livre à Mary". Il est probable qu'elle l'ait rendu.

14) CARACTERISTIQUE (*Feature Inferences*)

Un état ou une situation donnée peut apporter une information supplémentaire sur les traits d'un personnage. "Le lit d'Andy est mouillé." C'est probablement un bébé.

15) SITUATION (*Situation Inferences*)

De certaines situations bien connues, on peut déduire certaines connaissances. "Mary va à un bal masqué." Elle est probablement déguisée.

16) (*Utterance-Intent Inferences*)

Certaines choses peuvent être déduites suivant la manière dont est présentée l'information. Pourquoi le locuteur l'a-t-il dit ? Pourquoi de cette manière ? Aurait-il besoin d'aide ?

Annexe 3 : Formalisme à la Prolog

Jean Fargues suppose que l'on dispose au départ d'un ensemble de clauses de Horn sous la forme : G <- G1, G2, .., Gn (G; G1, .., Gn sont des graphes conceptuels.)

La résolution du but G necessite celle des sous-buts G1, ..., Gn.

Si n est égal à 0, la clause de Horn est une clause de type fait.

- d'une clause "buts" du type "<- A1, A2, ..., An"

Les inférences menées s'appuient sur le théorème suivant (démontré par Sowa, dans [Sowa, 1984]) : "Si $u \leq v$, alors $\Phi u \leq \Phi v$."

Supposons :

- une clause G <- G1, G2, .., Gn
- un but : <- A

Si l'on a une projection p de A dans G, on a alors $G \leq A$ (et donc $\Phi G \supset \Phi A$), le but A sera remplacé par la pile de sous-buts "<- G1, G2, ..., Gn"

La notion de projection présente des analogies avec le "matching" en Prolog. Jean Fargues étend la notion de projection pour définir une opération de matching suffisamment générale.

Pour lui, résoudre le but/graphe conceptuel "<- A" (où A est un graphe conceptuel) revient à résoudre A ou éventuellement une restriction de A.

Par analogie avec la résolution des buts/Prolog, les concepts jouent le rôle des variables Prolog et la restriction de ces concepts celui de l'instanciation des variables. Les opérations de substitution pour la résolution des buts/Prolog corrspondent à des opérations de restrictions sur certains concepts des buts/graphes conceptuels.

La notion de compatibilité de concepts lui permettra de définir l'opération d'unification dans les graphes.

Deux concepts C1 et C2 sont dits **compatibles** s'il existe une restriction commune C3 non triviale de C1 et C2.

Exemple :

[PERSONNE : 'John'] et [GARCON : *x] sont dits compatibles grâce à [GARCON :' John'].

Si le graphe v "matche" avec un graphe u, il existe alors un sous-graphe u' de u tel que :

1 - les relations conceptuelles de v et u' sont les mêmes

2 - Si la relation conceptuelle r relie les concepts ci et cj dans u' et les cconcepts di et dj dans v, alors ci (respectivement cj) doit être compatible avec di (respectivement dj).

Lors d'un appariement, deux opérations de substitutions ont alors lieu : l'un (e1) sur le graphe u et l'autre (e2) sur le graphe v. Si l'on note ri est la restriction commune maximale des concepts ci et di, la substitution e1 (respectivement e2) consiste à substituer le concept ci (respectivement di) de u (respectivement v) en ri.

On notera que toute projection est un matching, mais que tout matching n'est pas nécéssairement une projection. L'algorithme de résolution similaire au mécanisme de résolution de Prolog est défini (voir figure 1).

```
Soit P la pile de buts initiaux.
Pile de buts courants <- P
FAIRE soit <- A1,A2,...,Ap la pile de buts courants
soit G<- G1,G2,...,Gn ou  G <- une clause
But courant <-A
  SI il existe un matching entre G et A1
  ALORS
     Soient e1 et e2 les substitutions associées respectivement à G et A1 par ce
     matching.
     Propager les substitutions dans la partie droites si celle-ci n'est pas vide
     (i-e construire e1(G1),...,e1(Gn))
     Propager les substitutions dans la pile de buts restants
     (i-e construire e2(A2),...,e2(Ap) ainsi que dans P (i-e construire e2(P) )
     La nouvelle pile de buts <- e1(G1),...,e1(G1),e2(A2),... e2(Ap)
     ou <- e2(A2),...,A2(Ap) si n=0 (clause fait)
  FINSI
FINFAIRE

Résultat = pile P à laquelle ont été appliquées les substitutions
                au cours de la résolution.
```

figure 1 : Algorithme de résolution donné par Jean Fargues

Jean Fargues nous fait remarquer ici que les substitutions effectuées sont du type concept/concept, et qu'il est possible d'envisager des substitutions concept/graphe.

Annexe 4 :
Points d'implémentation d'un système à base de DCAS en Prolog

Cette annexe décrit quelques points d'implémentation en Prolog d'un système à base de DCAS.

1) Lorsqu'on utilise la fonction "tell" pour introduire une DCAS, le système va générer la représentation interne suivante : dp(TypeDescBrut, Id, Ref, admise,S)

Par exemple, à l'entrée :

```
[[situation,cible,*,
        [ [instance,objet,*,oeuf],[concept,act,*,casser],
          [instance,acteur,*,jean],[vdv,*,*,vrai],
          [rt,début,*,rt1],[rt,fin,*,rt2]
        ]
      ],
  [relation,prédicat,*,entre],[instance,site,1,maison1],
  [instance,site,2,maison2],
  [rt,début,*,rt1],
  [rt,fin,*,rt2],
  [vdv,*,*,vrai],
  [symbole,td,*,situation],
  [symbole,rd,*,s6],
  [*,*,*,*]]
```

le système introduira

```
dp( situation,
    s6,
    s6,
    admise,
   [quad(situation, cible, *,
         [quad(instance, objet, *, oeuf),
          quad(concept, act, *, casser),
          quad(instance, acteur, *, jean),
          quad(vdv, *, *, vrai),
          quad(rt, début, *, rt1),
          quad(rt, fin, *, rt2)]),
    quad(relation, prédicat, *, entre),
    quad(instance, site, 1, maison1),
    quad(instance, site, 2, maison2),
    quad(rt, début, *, rt1),
    quad(rt, fin, *, rt2),
    quad(vdv, *, *, vrai)]).
```

Les prédicats "quad" possèdent 4 arguments et représentent les quadruplets d'une DCAS. Le dernier des arguments (le référent) est soit un atome Prolog, soit une liste Prolog si le référent est une DCAS.

2) La routine askpDCAS est utilisé par la fonction ASK de traitement des requêtes. TER est le type de la description recherchée. C'est soit une situation, soit une proposition. La variable PartDescF1 contient une DCAS écrite sous la forme "[[a,b,c,d][a',b',c',d'],..., X]" et peut contenir une variable marquant la complétude de le description recherchée. La variable Description sera la Description trouvée par le système. La variable Chemin donnera la trace des DCAS qui ont permis d'asserter la DCAS trouvée. A la sortiede askpDCAS, la variable X est unifiée à la liste de quadruplets qu'il doit contenir.

```
askpDCAS(TER,PartDescF1,DescRF1,Chemin) :-
  f1ToF2EtCouples(TER,PartDescF1,PartDescF2,CouplesF1F2),
  one(extraitVariableEtHomogeinise(typeEntiteS(TER),
                                   descSource(PartDescF2),
                                   listeVar(LVR),descRes(DescR))),
chercheDCAS(X,
     typeEntiteR(TER),
     typeEntiteB(TEB),
     listeVarR(LVR),
     listeVarB(LVB),
     listeVarRR(LVRR),
     listeVarBR(LVBR),
     descR(DescR),
     descB(DescB),
     descRR(DescRR),
     descBR(DescBR),
     report(Report),
     chemin(Chemin),
     niveau(0),[]),
f2ToF1(CouplesF1F2),
  detruitPartieVariableRestante(LVRR,LVRR2),
  append(LVRR2,DescRR,DescRF2),
  one(phrase(phraseDescBrut(DescRF1),DescRF2)).
```

3) chercheDCAS est la fonction qui commence la recherche d'une DCAS tout d'abord parmi les DCAS assertées, puis ensuite essaie dans les conclusions des DCAS de type règle.

```
chercheDCAS(assertees,
     typeEntiteR(TER),
     typeEntiteB(TEB),
     listeVarR(LVRClone),
     listeVarB(LVB),
     listeVarRR(LVRR),
     listeVarBR(LVBR),
     descR(DescRI),
     descB(DescBI),
     descRR(DescR2),
     descBR(DescB2),
     report(REPORT),
     chemin(RefDescription),
     niveau(N),LJI) :-
  catch(0,dp(TEB,IdDescUnique,RefDescription,admise,DescB)),
  DescBI = DescB,LVB= [],

  da(typeEntiteR(TER),
     typeEntiteB(TEB),
     listeVarR(LVRClone),
     listeVarB(LVB),
     listeVarRR(LVRR),
```

```
              listeVarBR(LVBR),
              descR(DescRI),
              descB(DescBI),
              descRR(DescR2),
              descBR(DescB2),
              report(REPORT)),
           unifieReportFinal(REPORT).

  chercheDCAS(inferees,
       typeEntiteR(TER),
       typeEntiteB(TEB),
       listeVarR(LVRClone),
       listeVarB(LVB),
       listeVarRR(LVRR),
       listeVarBR(LVBR),
       descR(DescRI),
       descB(DescBI),
       descRR(DescR2),
       descBR(DescB2),
       report(REPORT),
       chemin(Chemin),
       niveau(NiveauN),LJI) :-
    catch(0,dp(regle,IdDescUnique,RefRegle,admise,ListeQuadsRegle)),
    member(quad(TER,conclusion,NC1,DescBConclusion),
                ListeQuadsRegle,IndiceN),
    one(extraitVariableEtHomogeinise(typeEntiteS(TEB),
                                     descSource(DescBConclusion),
                                     listeVar(LVB),descRes(DescBI))),
  one(da(typeEntiteR(TER),
       typeEntiteB(TEB),
       listeVarR(LVRClone),
       listeVarB(LVB),
       listeVarRR(LVRR),
       listeVarBR(LVBR),
       descR(DescRI),
       descB(DescBI),
       descRR(DescR2),
       descBR(DescB2),
       report(ReportConclusion))),
    unifieReportFinal(ReportConclusion),
    nAppartientPasAListeJeuxInstanciations(RefRegle,DescB2,LJI),
    ajouteJeuInstanciation(RefRegle,DescB2,LJI,LJI2),
    NiveauN2 is NiveauN + 1,
    one(extraitTousLesAntecedants(ListeQuadsRegle,ListeQuadsAntecedants,
                                  IndiceN)),
    verifieTousLesAntecedants(ListeQuadsAntecedants,
                              ListeCheminPourListeAntecedants,
                              LJI2,ReportsAntecedants,NiveauN),
  %   (NiveauN = 0, unifieReportFinal(ReportsAntecedants) ; true),
    append(ReportConclusion,ReportsAntecedants,REPORT),
    (NiveauN = 0, unifieReportFinal(REPORT) ;
     NiveauN \= 0, write('NiveauN <> 0'),true),
    Chemin = [règle(RefRegle,ListeCheminPourListeAntecedants)].
```

4) La fonction da/11 (da avec une arité 11) décrit toutes les conditions d'associabilité de deux DCAS. Le prédicat ci-dessous définit l'associabilité de deux DCAS complètes.

```
  da( typeEntiteR(TER),
      typeEntiteB(TEB),
      listeVarR(LVR),
      listeVarB(LVB),
```

```
      listeVarRR(LVRR),
      listeVarBR(LVBR),
      descR(DescR),
      descB(DescB),
      descRR(DescR2),
      descBR(DescB2),
      report(REPORT)) :-
    TER = TEB, % Les deux descriptions sont supposées de même type
    LVR = [],
    LVB = [],
    LVRR = [],
    LVBR = [],
    !,
    % pas de variables dans les deux descriptions
    % Le nombre de quads doit être le même
    len(DescR,N),
    len(DescB,N),
    % on regarde quad par quad les compatibilités
    ssEnsAssociableRNVBNV(
      typeEntiteR(TER),
      typeEntiteB(TEB),
      descR(DescR),
      descB(DescB),
      descRR(DescR2),
      descBR(DescB2),
      report(REPORT)).


  ssEnsAssociableRNVBNV(
      typeEntiteR(TER),
      typeEntiteB(TEB),
      descR([]),
      descB([]),
      descRR([]),
      descBR([]),
      report([])) :- !.

  ssEnsAssociableRNVBNV(
      typeEntiteR(TER),
      typeEntiteB(TEB),
      descR([quad(QR_TE,QR_RE,QR_NR,QR_REF) | ResteDescR]),
      descB(DescB),
      descRR([quad(QRR_TE,QRR_RE,QRR_NR,QRR_REF) |ResteDescRR] ),
      descBR(DescB2),
      report(REPORT)) :-
    member(quad(QB_TE,QB_RE,QB_NR,QB_REF), DescB, N),
    qa(quadR(QR_TE,QR_RE,QR_NR,QR_REF) ,   % Quad Référence
       quadB(QB_TE,QB_RE,QB_NR,QB_REF),   % Quad Balayé
       quadRR(QRR_TE,QRR_RE,QRR_NR,QRR_REF), % Quad Référence Résultant
       quadBR(QBR_TE,QBR_RE,QBR_NR,QBR_REF), % Quad Balayé Résultant
       report(Report1)),
    removeN(DescB,N, DescBI),
    ssEnsAssociableRNVBNV(
      typeEntiteR(TER),
      typeEntiteB(TEB),
      descR(ResteDescR),
      descB(DescBI),
      descRR(ResteDescRR),
      descBR(DescBIR),
      report(Report2)),
    append([quad(QBR_TE,QBR_RE,QBR_NR,QBR_REF)], DescBIR, DescB2),
    append(Report1,Report2,REPORT).
```

5) Le prédicat qa/5 décrit les conditions d’associabilité d’un quadruplet avec un autre.