Logique du temps arborescent pour la spécification et la preuve de programmes

Susanne Graf

THESE

présentée à

l'Institut National Polytechnique de Grenoble

pour obtenir le grade de
DOCTEUR DE 3ème CYCLE
«informatique»

par

Susanne GRAF

LOGIQUES DU TEMPS ARBORESCENT POUR LA SPECIFICATION ET LA PREUVE DE PROGRAMMES.

Thèse soutenue le 29 février 1984 devant la commission d'examen.

J. MOSSIERE	Président
K. APT	Examinateurs
G. BOUDOL	
Ph. JORRAND	
M. NIVAT	
J. SIFAKIS	

INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

Année universitaire 1982-1983

Président de l'Université : D. BLOCH

Vice-Président : René CARRE
Hervé CHERADAME
Marcel IVANES

PROFESSEURS DES UNIVERSITES :

ANCEAU François	E.N.S.I.M.A.G.
BARRAUD Alain	E.N.S.I.E.G.
BAUDELET Bernard	E.N.S.I.E.G.
BESSON Jean	E.N.S.E.E.G.
BLIMAN Samuel	E.N.S.E.R.G.
BLOCH Daniel	E.N.S.I.E.G.
BOIS Philippe	E.N.S.H.G.
BONNETAIN Lucien	E.N.S.E.E.G.
BONNIER Etienne	E.N.S.E.E.G.
BOUVARD Maurice	E.N.S.H.G.
BRISSONNEAU Pierre	E.N.S.I.E.G.
BUYLE BODIN Maurice	E.N.S.E.R.G.
CAVAIGNAC Jean-François	E.N.S.I.E.G.
CHARTIER Germain	E.N.S.I.E.G.
CHENEVIER Pierre	E.N.S.E.R.G.
CHERADAME Hervé	U.E.R.M.C.P.P.
CHERUY Arlette	E.N.S.I.E.G.
CHIAVERINA Jean	U.E.R.M.C.P.P.
COHEN Joseph	E.N.S.E.R.G.
COUMES André	E.N.S.E.R.G.
DURAND Francis	E.N.S.E.E.G.
DURAND Jean-Louis	E.N.S.I.E.G.
FELICI Noël	E.N.S.I.E.G.
FOULARD Claude	E.N.S.I.E.G.
GENTIL Pierre	E.N.S.E.R.G.
GUERIN Bernard	E.N.S.E.R.G.
GUYOT Pierre	E.N.S.E.E.G.
IVANES Marcel	E.N.S.I.E.G.
JAUSSAUD Pierre	E.N.S.I.E.G.
JOUBERT Jean-Claude	E.N.S.I.E.G.
JOURDAIN Geneviève	E.N.S.I.E.G.
LACOUME Jean-Louis	E.N.S.I.E.G.
LATOMBE Jean-Claude	E.N.S.I.M.A.G.

.../...

LESSIEUR Marcel	E.N.S.H.G.
LESPINARD Georges	E.N.S.H.G.
LONGEQUEUE Jean-Pierre	E.N.S.I.E.G.
MAZARE Guy	E.N.S.I.M.A.G.
MOREAU René	E.N.S.H.G.
MORET Roger	E.N.S.I.E.G.
MOSSIERE Jacques	E.N.S.I.M.A.G.
PARIAUD Jean-Charles	E.N.S.E.E.G.
PAUTHENET René	E.N.S.I.E.G.
PERRET René	E.N.S.I.E.G.
PERRET Robert	E.N.S.I.E.G.
PIAU Jean-Michel	E.N.S.H.G.
POLOUJADOFF Michel	E.N.S.I.E.G.
POUPOT Christian	E.N.S.E.R.G.
RAMEAU Jean-Jacques	E.N.S.E.E.G.
RENAUD Maurice	U.E.R.M.C.P.P.
ROBERT André	U.E.R.M.C.P.P.
ROBERT François	E.N.S.I.M.A.G.
SABONNADIERE Jean-Claude	E.N.S.I.E.G.
SAUCIER Gabrielle	E.N.S.I.M.A.G.
SCHLENKER Claire	E.N.S.I.E.G.
SCHLENKER Michel	E.N.S.I.E.G.
SERMET Pierre	E.N.S.E.R.G.
SILVY Jacques	U.E.R.M.C.P.P.
SOHM Jean-Claude	E.N.S.E.E.G.
SOUQUET Jean-Louis	E.N.S.E.E.G.
VEILLON Gérard	E.N.S.I.M.A.G.
ZADWORNY François	E.N.S.E.R.G.

PROFESSEURS ASSOCIES

BASTIN Georges	E.N.S.H.G.
BERRIL John	E.N.S.H.G.
CARREAU Pierre	E.N.S.H.G.
GANDINI Alessandro	U.E.R.M.C.P.P.
HAYASHI Hirashi	E.N.S.I.E.G.

PROFESSEURS UNIVERSITE DES SCIENCES SOCIALES (Grenoble II)

BOLLIET Louis
Chatelin Françoise

PROFESSEURS E.N.S. Mines de Saint-Etienne

RIEU Jean
SOUSTELLE Michel

CHERCHEURS DU C.N.R.S.

FRUCHART Robert	Directeur de Recherche
VACHAUD Georges	Directeur de Recherche

.../...

ALLIBERT Michel	Maître de Recherche
ANSARA Ibrahim	Maître de Recherche
ARMAND Michel	Maître de Recherche
BINDER Gilbert	
CARRE René	Maître de Recherche
DAVID René	Maître de Recherche
DEPORTES Jacques	
DRIOLE Jean	Maître de Recherche
GIGNOUX Damien	
GIVORD Dominique	
GUELIN Pierre	
HOPFINGER Emil	Maître de Recherche
JOUD Jean-Charles	Maître de Recherche
KAMARINOS Georges	Maître de Recherche
KLEITZ Michel	Maître de Recherche
LANDAU Ioan-Dore	Maître de Recherche
LASJAUNIAS J.C.	
MERMET Jean	Maître de Recherche
MUNIER Jacques	Maître de Recherche
PIAU Monique	
PORTESEIL Jean-Louis	
THOLENCE Jean-Louis	
VERDILLON André	

CHERCHEURS du MINISTERE de la RECHERCHE et de la TECHNOLOGIRE (Directeurs et Maîtres de Recherches, ENS Mines de St. Etienne)

LESBATS Pierre	Directeur de Recherche
BISCONDI Michel	Maître de Recherche
KOBYLANSKI André	Maître de Recherche
LE COZE Jean	Maître de Recherche
LALAUZE René	Maître de Recherche
LANCELOT Francis	Maître de Recherche
THEVENOT François	Maître de Recherche
TRAN MINH Canh	Maître de Recherche

PERSONNALITES HABILITEES à DIRIGER des TRAVAUX de RECHERCHE (Décision du Conseil Scientifique)

ALLIBERT Colette	E.N.S.E.E.G.
BERNARD Claude	E.N.S.E.E.G.
BONNET Rolland	E.N.S.E.E.G.
CAILLET Marcel	E.N.S.E.E.G.
CHATILLON Catherine	E.N.S.E.E.G.
CHATILLON Christian	E.N.S.E.E.G.
COULON Michel	E.N.S.E.E.G.
DIARD Jean-Paul	E.N.S.E.E.G.
EUSTAPOPOULOS Nicolas	E.N.S.E.E.G.
FOSTER Panayotis	E.N.S.E.E.G.

.../...

GALERIE Alain	E.N.S.E.E.G.
HAMMOU Abdelkader	E.N.S.E.E.G.
MALMEJAC Yves	E.N.S.E.E.G. (CENG)
MARTIN GARIN Régina	E.N.S.E.E.G.
NGUYEN TRUONG Bernadette	E.N.S.E.E.G.
RAVAINE Denis	E.N.S.E.E.G.
SAINFORT	E.N.S.E.E.G. (CENG)
SARRAZIN Pierre	E.N.S.E.E.G.
SIMON Jean-Paul	E.N.S.E.E.G.
TOUZAIN Philippe	E.N.S.E.E.G.
URBAIN Georges	E.N.S.E.E.G. (Laboratoire des ultra-réfractaires ODEILLON)
GUILHOT Bernard	E.N.S. Mines Saint Etienne
THOMAS Gérard	E.N.S. Mines Saint Etienne
DRIVER Julien	E.N.S. Mines Saint Etienne
BARIBAUD Michel	E.N.S.E.R.G.
BOREL Joseph	E.N.S.E.R.G.
CHOVET Alain	E.N.S.E.R.G.
CHEHIKIAN Alain	E.N.S.E.R.G.
DOLMAZON Jean-Marc	E.N.S.E.R.G.
HERAULT Jeanny	E.N.S.E.R.G.
MONLLOR Christian	E.N.S.E.R.G.
BORNARD Guy	E.N.S.I.E.G.
DESCHIZEAU Pierre	E.N.S.I.E.G.
GLANGEAUD François	E.N.S.I.E.G.
KOFMAN Walter	E.N.S.I.E.G.
LEJEUNE Gérard	E.N.S.I.E.G.
MAZUER Jean	E.N.S.I.E.G.
PERARD Jacques	E.N.S.I.E.G.
REINISCH Raymond	E.N.S.I.E.G.
ALEMANY Antoine	E.N.S.H.G.
BOIS Daniel	E.N.S.H.G.
DARVE Félix	E.N.S.H.G.
MICHEL Jean-Marie	E.N.S.H.G.
OBLED Charles	E.N.S.H.G.
ROWE Alain	E.N.S.H.G.
VAUCLIN Michel	E.N.S.H.G.
WACK Bernard	E.N.S.H.G.
BERT Didier	E.N.S.I.M.A.G.
CALMET Jacques	E.N.S.I.M.A.G.
COURTIN Jacques	E.N.S.I.M.A.G.
COURTOIS Bernard	E.N.S.I.M.A.G.
DELLA DORA Jean	E.N.S.I.M.A.G.
FONLUPT Jean	E.N.S.I.M.A.G.
SIFAKIS Joseph	E.N.S.I.M.A.G.
CHARUEL Robert	U.E.R.M.C.P.P.
CADET Jean	C.E.N.G.
COEURE Philippe	C.E.N.G. (LETI)

.../...

DELHAYE Jean-Marc	C.E.N.G. (STT)
DUPUY Michel	C.E.N.G. (LETI)
JOUVE Hubert	C.E.N.G. (LETI)
NICOLAU Yvan	C.E.N.G. (LETI)
NIFENECKER Hervé	C.E.N.G.
PERROUD Paul	C.E.N.G.
PEUZIN Jean-Claude	C.E.N.G. (LETI)
TAIEB Maurice	C.E.N.G.
VINCENDON Marc	C.E.N.G.

LABORATOIRES EXTERIEURS

DEMOULIN Eric	C.N.E.T.
DEVINE	C.N.E.T. (R.A.B.)
GERBER Roland	C.N.E.T.
MERCKEL Gérard	C.N.E.T.
PAULEAU Yves	C.N.E.T.
GAUBERT C.	I.N.S.A. Lyon

Résumé:

Nous étudions les logiques du temps arborescent en tant qu'outils de spécification et de preuve des programmes. Les différentes logiques modales et temporelles sont comparées par rapport aux deux critères suivantes: puissance d'expression et décidabilité. Cette étude porte essentiellement sur la comparaison des logiques du temps arborescent et des logiques du temps linéaire. Ensuite, le problème de l'utilisation des logiques du temps arborescent en tant qu'outils de preuves des programmes est étudié. Nous proposons une logique pour la preuve constructive des propriétés des processus contrôlables de CCS. Cette logique est telle, que la relation de congruence induite est la congruence observationnelle de CCS.

Mots-clés:
Spécification de programmes, preuve de programmes, logique temporelle, temps linéaire, temps arborescent, puissance d'expression, axiomatisation, procédure de décision, CCS, preuves constructives.

Remerciements:

Je tiens à remercier

Monsieur J.Mossière, Professeur à l'Université de Grenoble, qui m'a fait l'honneur de présider le jury de cette thèse.

Monsieur P. Jorrand, Directeur de recherche au CNRS, pour m'avoir accueilli dans son équipe.

Monsieur J. Sifakis, Chargé de recherche au CNRS, pour les nombreuses discussions que nous avons eues ensemble et pour le temps qu'il a consacré à l'encadrement de cette thèse.

Monsieur K. Apt, Chargé de recherche au CNRS, Monsieur G. Boudol, Ingénieur à l'INRIA, et Monsieur M. Nivat, Professeur de l'Université Paris VII pour avoir accepté de participer au jury de cette thèse.

Je remercie également tous ceux qui par leurs suggestions ont contribué à l'amélioration de cette thèse, et particulièrement J.C. Fernandez et J. Voiron.

Je remercie aussi D. Iglésias et son équipe pour le soin qu'ils ont apporté à l'impression de cette thèse.

LOGIQUES DU TEMPS ARBORESCENT POUR LA SPECIFICATION ET LA PREUVE DE PROGRAMMES

SOMMAIRE

III. UTILISATION DES LOGIQUES DU TEMPS ARBORESCENT EN TANT QU'OUTIL DE SPECIFICATION: PUISSANCE D'EXPRESSION, DECIDABILITE

1. Introduction

2. La logique du temps arborescent LTA

3. La logique UB

4. La logique du temps arborescent conditionnelle LTAC
   - 4.1 La logique LTAC
   - 4.2 La procédure de décision de LTAC
   - 4.3 La complétude de LTAC

5. Comparaison des logiques du temps linéaire et des logiques du temps arborescent
   - 5.1 La logique LTL
   - 5.2 Un critère de comparaison entre logiques
   - 5.3 Comparaison de LTL et LTA
   - 5.4 Conclusion

IV. UTILISATION DES LOGIQUES DU TEMPS ARBORESCENT EN TANT QU'OUTIL DE PREUVE DE PROGRAMMES: APPLICATION POUR UN SOUS-ENSEMBLE DE CCS

1. Introduction

2. Caractérisation de la congruence observationnelle sur les termes finis de CCS
   - 2.1 La logique du temps arborescent relativisée LTAR
   - 2.2 Le langage de description de programmes $T[\Sigma]$
   - 2.3 Caractérisation modale de la congruence forte ~
   - 2.4 Caractérisation modale de la congruence observationnelle

I.INTRODUCTION

Ce travail est motivé par le problème de la vérification des programmes parallèles.
Vérifier un programme consiste à le comparer à ses spécifications qui, dans le cas général, peuvent être soit une description algorithmique, soit un ensemble de propriétés, représentées par des assertions sur le comportement. C'est cette deuxième forme de spécifications que nous considérons dans toute la suite.

L'expression des propriétés des programmes nécessite l'utilisation d'un langage formel, en fait une logique. Pour les programmes séquentiels, très peu de propriétés sont nécessaires pour l'expression de leurs spécifications et le problème de la définition d'un langage de spécification ne se pose pas. Par contre pour les programmes parallèles, on a besoin d'un très grand nombre de types de propriétés. Il est alors important de disposer d'un langage de spécification qui soit suffisamment puissant et pour lequel il existe des résultats permettant de détecter des incohérences ou des redondances dans les spécifications.

Les logiques ont été souvent utilisées comme langages de spécification de programmes. Dans ce cas deux approches sont possibles: soit l'utilisation

du calcul des prédicats, par exemple [AS], soit l'utilisation de logiques temporelles, par exemple [Pn].[Ab]. La deuxième approche est plus récente et présente à notre avis des avantages considérables par rapport à la première. En effet, les logiques temporelles sont des extensions du calcul propositionnel par adjonction d'opérateurs temporels qui représentent en fait des "paquets" de quantificateurs sur les états et les séquences d'exécution d'un système de transitions. Ces opérateurs définissent des modalités temporelles: par exemple "Il est possible que", "Il est inévitable que", "Il est toujours vrai que" .

Depuis quelques années un certain nombre de logiques temporelles ayant une puissance d'expression satisfaisante ont été proposées [BMP2], [EH2]. Il a été prouvé que ces logiques sont décidables. De plus, elles présentent par rapport au calcul de prédicats l'avantage d'offrir des concepts de plus haut niveau qui facilitent l'expression naturelle des propriétés. Ces faits expliquent l'intérêt grandissant porté à ces logiques.

Il existe deux familles de logiques utilisées pour la spécification des programmes: les logiques du temps arborescent et les logiques du temps linéaire. Si l'on assimile les programmes à des systèmes de transitions, les formules des logiques du temps arborescent expriment des propriétés de leurs états tandis que les formules des logiques du temps linéaire des propriétés de leurs séquences d'exécution.

L'objectif de ce travail est essentiellement l'étude des logiques du temps arborescent en tant qu'outils de spécification et de preuve de programmes.

Dans le deuxième chapitre de cette thèse nous étudions les différentes logiques modales et temporelles classiques, c'est-à-dire des logiques développées en dehors du cadre spécifique de l'étude des programmes. L'objectif est la présentation de résultats fondamentaux sur ces logiques utilisées par la suite.

Le problème de l'adéquation des logiques temporelles à la spécification des programmes est étudié au troisième chapitre. Nous présentons des résultats concernant l'axiomatisation et la décidabilité des logiques

du temps arborescent. Leur puissance de description est comparée à celle des logiques du temps linéaire. Nous avons prouvé que ces deux familles de logiques ont des puissances de description non comparables, c'est-à-dire qu'il existe des propriétés exprimables par l'une et non par l'autre et inversement. Plus particulièrement, il existe une classe importante de propriétés (notamment celles exprimant l'absence du blocage partiel) non exprimables en logique du temps linéaire.

Au quatrième chapitre nous considérons l'utilisation de ces logiques pour la preuve de programmes. Plus particulièrement, nous nous intéressons aux logiques permettant des faire des preuves constructives des programmes décrits dans un modèle algébrique. Après avoir donné les conditions nécessaires que doit satisfaire une logique pour permettre des preuves constructives, nous cherchons à définir une telle logique en prenant comme modèle de description de programmes le calcul CCS de R. Milner [Mi]. Le principal résultat de ce chapitre est la définition d'une logique du temps arborescent pour la preuve constructive des processus "contrôlables" de CCS, c'est-à-dire des processus pour lesquels il existe un processus "observationnellement" équivalent sans τ-transition. Cette logique induit une relation d'équivalence sur les termes de CCS qui est la congruence observationnelle. De plus, pour chacun des opérateurs $+$, $a\in A$ (A est l'ensemble des actions) il existe des opérateurs $\oplus$, $\circledcirc$ de la logique tels que, pour t,t' processus contrôlables et f,f' formules, on a:

- $t\models f$ implique $at\models\circledcirc f$.
- $t\models f$, $t'\models f'$ implique $t+t'\models f\oplus f'$.
- Les assertions $at\models\circledcirc f$ et $t+t'\models f\oplus f'$ sont les assertions les plus fortes que l'on puisse déduire des hypothèses $t\models f$ respectivement $t\models f$ et $t'\models f'$.

II. LOGIQUES MODALES CLASSIQUES

II.1 Introduction

La première partie de ce chapitre est consacrée à la présentation des définitions concernant les logiques en général, et les logiques modales en particulier, auxquelles nous faisons référence par la suite. Dans la deuxième partie nous présentons des logiques modales normales. En effet, les logiques modales normales sont les logiques modales les plus étudiées, et les premières logiques modales proposées pour l'expression des propriétés de programmes sont des logiques de ce type.

Après la définition générale d'une logique modale normale, qui sera utilisée pour les logiques modales à plusieurs opérateurs modaux, définies en chapitre III, nous considérons quelques logiques modales normales, obtenues en ajoutant différents axiomes à l'axiomatisation de la logique modale normale minimale K. Le treillis d'inclusions entre les calculs correspondants est également présenté. Parmi ces logiques se trouvent les logiques modales normales les plus étudiées dans la littérature. Nous donnons aussi l'axiomatisation de la logique S4.3.1 qui se trouve être l'axiomatisation de la logique du temps linéaire LTL définie en III.5.

2.1 Le langage de formules

Un langage de formules F est un ensemble de formules qui contient le sous-langage F_{CP}, l'ensemble de formules du langage propositionnel On représente par $F_{CP}(\mathbf{P})$ l'ensemble des termes construits à partir d'une signature Σ_{CP} et d'un ensemble de variables propositionnelles **P**, où

$\mathbf{P} = \{p,q,...\}$

$\Sigma_{CP} = \{v,\neg,T\}$ où v est la disjonction, $\neg$ est la négation et T la constante logique vrai.

Lorsque l'ensemble **P** n'est pas significatif on écrit simplement F_{CP}. On définit les abréviations suivantes pour $f_1,f_2 \in F_{CP}$:

$f_1 \wedge f_2 := \neg(\neg f_1 v \neg f_2)$ (la conjonction)
$f_1 \supset f_2 := \neg f_1 v f_2$ (l'implication)
$f_1 \equiv f_2 := (f_1 \supset f_2) \wedge (f_2 \supset f_1)$ (l'équivalence)
$\perp := \neg T$ (la constante faux)

Le langage de formules F_{LM} d'une logique modale LM est construit à partir d'une signature, contenant des opérateurs modaux. Dans un premier temps, on va considérer des logiques modales pour lesquelles F_{LM} est le langage des termes sur la signature $\Sigma' = \Sigma_{CP} \cup \{\Box\}$, où $\Box$ est un opérateur unaire.
Pour l'opérateur dual de $\Box$ on définit l'abréviation

$\Diamond(f) := \neg\Box(\neg f)$.

2.2 Le calcul logique

Un calcul logique est un triplet Cal=(F,Ax,Reg), où

- F est un langage logique,
- $Ax \subseteq F$ est un ensemble d'axiomes
- $Reg \subseteq \bigcup_{n\geq 2} F^n$ est un ensemble de règles d'inférences; et pour $(f_1,\ldots,f_n,f) \in Reg$ on écrit

$$\frac{\vdash f_1,\ldots,\vdash f_n}{\vdash f}$$

L'ensemble de théorèmes, noté Th, est obtenu à partir de Ax et Reg de la manière suivante:

$$Th = \{f \in F \mid \exists f_1,\ldots,f_n \in F \ f_n=f \text{ et } \forall i<n \ f_i \in Ax \text{ ou } \exists i_1,\ldots,i_{ji}<i \ (f_{i_1},\ldots,f_{i_{ji}},f_i) \in Reg\}$$

C'est-à-dire Th est l'ensemble de formules que l'on obtient en "appliquant" les règles de Reg aux axiomes ou aux formules "déjà prouvées". On appelle $f_1,\ldots,f_n$ une preuve pour f. Pour $f \in Th$ on écrit également $\vdash f$.

Tout calcul logique contient au moins les théorèmes du calcul propositionnel $CP=(F_{CP},Ax_{CP},Reg_{CP})$. Une axiomatisation possible du CP est la suivante [HU].

Ax_{CP} contient toutes les formules de la forme

(CP1) $f \vee f \supset f$

(CP2) $g \supset f \vee g$

(CP3) $g \vee f \supset f \vee g$

(CP4) $g \vee f \supset ((g \vee h) \supset (f \vee h))$

Reg_{CP} contient toutes les règles d'inférences de la forme

(MP) modus ponens: $\dfrac{\vdash f,\ \vdash f \supset g}{\vdash g} \in F_{CP}$

Si Th est récursivement énumérable, on dit que Cal est axiomatisable et on peut trouver Ax décidable [Ro]. Lorsqu'un calcul est axiomatisable, on l'appelle aussi un système logique ou un système de preuves.

On appelle Cal consistant si $\vdash f$ implique non $\vdash \neg f$. Un calcul consistant est fortement complet, si pour tout non-théorème f (c.à.d. non $\vdash f$) le calcul $Cal'=(F,Ax \cup \{f\},Reg)$ est inconsistant.

Proposition 1 [HC]

Le calcul propositionnel CP est consistant et fortement complet. □

Tous les calculs logiques, auxquels on s'intéresse par la suite, sont axiomatisables et consistants. Ils ne sont pas fortement complets, mais en fait, il n'existe pas de calcul logique modal fortement complet [HC], et cette propriété n'est pas désirable pour l'utilisation que l'on veut faire des logiques modales.

2.3 La sémantique

Un calcul logique Cal est simplement un système formel particulier. Pour obtenir une "logique" dans le sens habituel, on doit donner une sémantique au langage des formules via une interprétation. Etant donné F, un langage de formules, une interprétation est un triplet $I=(F,M,\models)$, où

- F est un langage logique
- M est une classe de modèles
- $\models$ est une relation de satisfaction, qui génère une notion de validité des formules par rapport aux modèles.

Pour le calcul propositionnel on considère $I_{CP}=(F_{CP},M_{CP},\models_{CP})$, où

- $M_{CP}=\{P \mid P\in 2^P\}$ c'est-à-dire les modèles sont des sous-ensembles de P.
- Pour chaque $P\in M_{CP}$ on définit $P\models_{CP}$ de la manière suivante:
  Si $f,f'\in F_{CP}$, alors
  (1) $P\models p$ ssi $p\in P$ et $P\models T$
  (2) $P\models \neg f$ ssi non $P\models f$
  (3) $P\models f\vee f'$ ssi $P\models f$ ou $P\models f'$
  Pour $P\models f$ on dit P satisfait f.
  On appelle f valide en I_{CP}, noté $\models f$, ssi $P\models f$ $\forall P\in M_{CP}$.

Pour une logique modale on considère $I_{LM}=(F_{LM}.M_{LM}.\models_{LM})$, où

- $M_{ML}=\{m \mid m=(W.R.P)\}$, où

  $W=\{w_i\}$ est un ensemble d'<u>états</u>.

  $R \in W \times W$ est une <u>relation d'accessibilité</u> entre états.

  $P \in W \to 2^P$ est une fonction qui associe à chaque état w_i un ensemble $P(w_i)=P_i \in 2^P$.

- Pour tout modèle m on définit $m \models_{LM}$ de la manière suivante:

  Si $f,f' \in F_{LM}$, $p \in P$, $m=(W.R.P)$ et $w \in W$, alors

  (1) $m.w \models p$ ssi $p \in P(w)$ et $m.w \models T$,

  (2) $m.w \models \neg f$ ssi non $m.w \models f$,

  (3) $m.w \models f \vee f'$ ssi $m.w \models f$ ou $m.w \models f'$,

  Dans un premier temps on considère des logiques modales, où

  (4) $m.w \models \Box f$ ssi $\forall w' \in W$ $((w,w') \in R$ implique $m.w \models f)$

  et par conséquent

  (4') $m.w \models \Diamond f$ ssi $\exists w' \in W$ $((w,w') \in R$ et $m.w' \models f)$

  On appelle f <u>satisfaisable</u> en I_{LM}, s'il existe un $m \in M_{ML}$ dont un état w est tel que $m.w \models f$.

  On appelle f <u>vrai en m</u>, noté $m \models f$, si pour tout état w de m $m.w \models f$.

  On appelle f <u>valide</u> en I_{LM}, noté $\models f$, si $m \models f$ pour tout m de M_{LM}.

<u>Remarque</u>: On voit que la définition de la satisfaction d'une formule f, ayant comme opérateur principal un opérateur propositionnel, dans un état w d'un modèle m est compatible avec la définition de la satisfaction dans un modèle m propositionnel.

2.4 La logique

Une logique L est une paire L=(Cal,I) où CaL=(F_L,Ax,Reg) est un calcul logique et I=(F_L,M,⊨) est une interprétation.

(Cal,I) est consistant ssi

⊢f implique ⊨f $\forall f \in F_L$.

On dit également que Cal est consistant par rapport à I.

(Cal,I) est complet ssi

⊨f implique ⊢f $\forall f \in F_L$.

On dit également que Cal est complet par rapport à I.

Le cas intéressant pour nous est celui où le calcul Cal est axiomatisable et (Cal,I) est consistant et complet, c'est-à-dire

⊨f ssi ⊢f $\forall f \in F_L$

Dans ce cas, on appelle Cal une axiomatisation consistante et complète de I.

Parfois, si L=(Cal,I), nous écrivons simplement L à la place de Cal ou I.

Proposition 2 [HC]

L=(CP,I_{CP}) est consistant et complet. □

Remarque: Il ne faut pas confondre la consistance d'un calcul logique Cal, qui exprime le fait qu'une formule ne peut pas être à la fois un théorème et un non-théorème, et la consistance de Cal par rapport à I, qui signifie que tout théorème de Cal est valide en I. (La même remarque peut être fait pour la complétude).

Soit L=(Cal,I) une logique consistante et complète. L (respectivement I) a la propriété de modèle fini (pmf) si, pour chaque formule satisfaisable en I, il existe un modèle fini qui la satisfait. Dans le cas des logiques modales ceci signifit qu'il existe un modèle m=(W,R,*P*) où W est fini.

et il existe $w \epsilon W$ tel que $m.w \models f$.
Ainsi, pour chaque non-théorème f de Cal, $\neg f$ est satisfaisable dans un modèle fini. Ceci implique que l'ensemble des non-théorèmes est récursivement énumérable. Etant donné que l'ensemble des théorèmes Th est récursivement énumérable (Cal est axiomatisable), on obtient que Th est décidable. Dans ce cas on dit que L est décidable.

Les logiques considérées par la suite ont la pmf, donc elles sont décidables. Dans ce cas, le problème de la preuve de la consistance et de la complétude de L=(Cal,I) peut être abordé de la manière suivante.

1. Pour prouver la consistance de Cal par rapport à I, on montre

- $\forall f \epsilon Ax_{Cal} \vdash f$ et
- $\frac{\vdash f_i \ i \epsilon I}{\vdash f} \epsilon Reg_{Cal}$ et $\vdash f_i$ pour $i \epsilon I$ implique $\vdash f$.

On prouve donc par induction structurelle que tout théorème est une formule valide.

2. Pour prouver la complétude de Cal par rapport à I, on peut utiliser le fait que l'ensemble des formules valides de I est décidable. Dans ce cas on trouve une procédure de décision pour I, i.e. un algorithme qui décide si une formule f donnée est valide en I, et qui, dans le cas contraire, donne un contre-modèle pour f, c'est-à-dire un modèle (fini) $m \epsilon M_I$ tel qu'il existe un état $w_0 \epsilon W$ et $m.w_0 \models f$. Les procédures de décision des logiques modales peuvent être basées sur le même principe. Pour décider la validité de f on part de l'hypothèse qu'il existe un modèle m et un état $w_0 \epsilon W$ tels que $m.w_0 \models \neg f$. A partir de cette hypothèse, on essaye de construire m de manière systématique. Un échec signifie que la formule f est valide. La complétude de Cal par rapport à I est ensuite montrée par induction sur les règles de construction de contre-modèle.
La procédure de décision pour LTAC en III.4 illustre cette méthode générale.

II.3 Les logiques modales normales

Presque toutes les logiques modales "classiques" sont des logiques modales normales. Dans ce paragraphe, partant de la logique modale normale minimale $K=(Cal_K, I_K)$, on obtient des logiques modales normales $K'=(Cal_K', I')$ en ajoutant des axiomes à Ax_K. On montre que ces axiomes supplémentaires induisent des restrictions sur la classe de modèles I' et plus particulièrement sur les propriétés des relations R des modèles $m=(W,R,P)$.

3.1 Le calcul modal normal

Un calcul modal normal $Cal_N=(F_{LM}, Ax, Reg)$, où F_{LM}, définie comme en II.2.1, contient les axiomes et règles d'inférence de CP et de plus les règles de la forme suivante:

$$(N^n)\quad \frac{\vdash f_1 \wedge \ldots \wedge f_n \supset f}{\vdash \Box f_1 \wedge \ldots \wedge \Box f_n \supset \Box f} \in F^2_{LM} \forall n \in \mathbf{N}.$$

Le calcul modal minimal est noté par Cal_K. A partir de ces axiomes et règles on peut déduire les théorèmes et règles suivants.

Proposition 1

Tout calcul modal normal Cal_N contient les théorèmes et règles suivants.

(1) $\frac{\vdash f}{\vdash \Box f}$ (règle de généralisation)

(2) $\frac{\vdash f \supset g}{\vdash \Box f \supset \Box g}$

(3) $\frac{\vdash f \equiv g}{\vdash \Box f \equiv \Box g}$

(4) $\vdash \Box T$

(5) $\vdash \Box(f \wedge g) \equiv \Box f \wedge \Box g$ (distributivité de $\Box$ par rapport à $\wedge$)

(6) $\vdash \Box(f \supset g) \supset (\Box f \supset \Box g)$ (axiome de normalité)

Preuve: Les preuves de (1)-(4) sont triviales, les preuves de (5) et (6) peuvent être trouvées en [Ch]. □

La signification de (1) et (6) devient claire par la proposition suivante qui exprime que les règles (N^n) peuvent être remplacées par (1) et (6) dans l'axiomatisation de tout calcul modal normal.

Proposition 2 [Ch]

Un calcul modal est normal ssi Th_{Cal} contient les axiomes de la forme (1) et les règles de la forme (6) de la proposition 1. □

Remarque: La définition d'un calcul modal peut être considérée comme la définition d'un opérateur modal normal □. Dans les logiques à plusieurs opérateurs modaux considérées par la suite, on dira que □ est normal si (1) et (6) de la proposition 1 découlent de l'axiomatisation.

3.2 La logique modale normale minimale K et d'autres logiques modales normales

La logique modale normale minimale est définie par $K=(Cal_K, I_K)$, où I_K est défini comme $I_{LM}=(F_{LM}, M_{LM}, \models)$ en II.2.3. Pour cette logique on a,

Proposition 3 [Ch]

La logique $K=(Cal_K, I_K)$ est consistante, complète et décidable. □

Par la suite, on propose 5 axiomes différents avec lesquels on peut enrichir Cal_K pour obtenir 15 calculs Cal' différents.
A chacun des axiomes correspond une restriction M'_{LM} de la classe de modèles M_{LM} telle que l'on a toujours pour $I'=(F_{LM}, M'_{LM}, \models)$ la consistance et la complétude de (Cal', I'). Les restrictions sur les classes de modèles sont en effet dans tous les cas des restrictions sur les

relations d'accessibilité R, admises dans les modèles $m=(W,R,P) \in M'_{LM} \subseteq M_{LM}$. Ci-dessous, on donne une liste de cinq axiomes et entre paranthèses la restriction sur R qui en découle:

(D) $\Box f \supset \Diamond f$ (R totale)

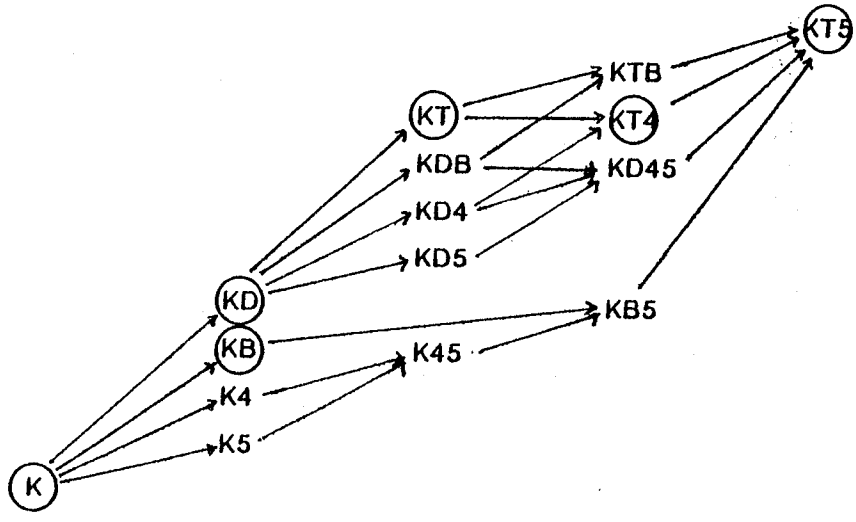
(T) $\Box f \supset f$ (R réflexive)

(B) $f \supset \Box\Diamond f$ (R symétrique)

(4) $\Box f \supset \Box\Box f$ (R transitive)

(5) $\Diamond f \supset \Diamond\Box f$ (R euclidienne c.à.d. $(w,w') \in R$ et $(w,w'') \in R$ implique $(w',w'') \in R$ et $(w'',w') \in R$)

Dans le graphe suivant, une logique K', obtenue par adjonction des axiomes à Cal_K, est notée par K suivi des noms des axiomes ajoutés. Par exemple, par KT4 est notée la logique obtenue par adjonction des axiomes (T) et (4) à Cal_K, et en posant $I_{KT4}=(F_{LM},M_{KT4},\models)$, où M_{KT4} contient les modèles de M_{LM} où R est réflexive et transitive.
Le graphe suivant représente seulement les logiques dont les calculs sont différents. Ici le critère d'inclusion entre logiques est simplement l'inclusion entre les ensembles de théorèmes correspondants. Ainsi, les inclusions et égalités utilisées pour le graphe suivant [Ch] sont faciles à prouver, et correspondent bien aux inclusions et égalités entre les classes de modèles, induites par les restrictions sur les relations R mentionnées.



Les logiques les plus étudiées dans la littérature parmi ces logiques sont les suivantes:

KD: souvent appelé D
KT: souvent appelé T
KB: connu comme le 'Browersche' sytème B
KT4: connu comme le système de Lewis S4
KT5: connu comme le système de Lewis S5

A part ces logiques, surtout des extensions de S4 ont été proposées. Ici, on en mentionne deux, contenues en S5, parce qu'elles sont intéressantes par la suite.
La première, connue comme S4.3, est obtenue par adjonction de l'axiome suivant à Cal_{S4}.

(L) $\Box(\Box f \vee g) \vee (\Box g \vee f)) \supset (\Box f \vee \Box g)$

(L) est équivalent à l'axiome

(L') $\Diamond f \wedge \Diamond g \supset \Diamond(f \wedge \Diamond g \vee g \wedge \Diamond f)$

qui est plus facile à interpréter.
Cet axiome induit une restriction des relations réflexives et transitives des modèles de S4, aux relations d'ordre linéaire dans le sens suivant:

$(w,w') \in R$ et $(w,w'') \in R$ implique $(w',w'') \in R$ ou $(w'',w') \in R$
et $(w,w') \in R$ et $(w',w) \in R$ implique $w=w'$.

Si les modèles considérés sont des systèmes de transitions (discrets), la restriction aux relations admises par (L) implique la validité d'un autre axiome qui n'est pas déductible du calcul de S4.3:

(DL) $\Box(\Box(f \supset \Box f) \supset f) \supset (\Diamond \Box f \supset f)$

(DL) est équivalent à

(DL') $\Box(\neg f \supset \Diamond(p \wedge \Diamond(\neg f))) \supset (\neg f \supset \Box\Diamond \neg f)$

qui est plus facile à interpréter.
On peut voir que (DL') est valide dans les modèles discrets où R est une relation d'ordre linéaire. Par contre, si W est un intervalle de

nombres réels, (DL') n'est plus valide.

La logique obtenue par adjonction de (DL) à $Cal_{S4.3}$ est connue comme S4.3.1. Ce calcul est mentionné ici parce qu'il est une axiomatisation de la logique du temps linéaire LTL, inroduite en III.5. De plus, pour l'opérateur non-normal SOME des logiques du temps arborescent introduites en III, les formules correspondant à (L') et (DL') sont valides.

III. UTILISATION DES LOGIQUES DU TEMPS ARBORESCENT EN TANT QU'OUTIL DE SPECIFICATION: PUISSANCE D'EXPRESSION, DECIDABILITE.

III.1 Introduction

Dans ce paragraphe, nous étudions le problème de l'adéquation des logiques temporelles à la spécification des programmes. Nous nous limitons aux logiques qui ont une modalité exprimant l'"inévitabilité", cette modalité étant nécessaire pour la description des propriétés des programmes parallèles ou non-déterministes.
Les différentes logiques considérées sont comparées par rapport aux deux critères suivants: puissance d'expression et décidabilité. Nous ne retiendrons que celles satisfaisant les deux critères. En effet, lorsqu'une logique est utilisée comme outil de spécification, il est nécessaire qu'elle permette l'expression des propriétés importantes du comportement. D'autre part, une puissance d'expression excessive peut entrainer l'indécidabilité.

Nous présentons une famille de trois logiques du temps arborescent, toutes extensions d'une logique de base LTA. Pour toutes ces logiques la propriété de décidabilité a été prouvée. Une procédure de décision

est donnée pour LTAC, la logique la plus générale de cette famille. Nous donnons également une axiomatisation de cette logique.

Le travail de comparaison est effectué entre LTA et la logique du temps linéaire LTL, base de toutes les logiques du temps linéaire utilisées pour la spécification des programmes. Du point de vue puissance d'expression ces deux logiques ne sont pas comparables si l'on prend comme classe commune de modèles les systèmes de transitions. Ceci renforce le résultat de L. Lamport [La1] pour une classe plus générale de modèles, contenant des comportements non réalisables par des programmes. Nous prouvons aussi que LTL admet la même axiomatisation que S4.3.1.

III.2 La logique du temps arborescent LTA

Dans le chapitre II on a considéré des logiques modales normales, c'est-à-dire des logiques avec un opérateur modal distributif par rapport à la conjonction. Nous nous intéressons aux logiques possédant un opérateur exprimant l'inévitabilité, une propriété qui ne peut pas être exprimée par un opérateur normal. Par la suite, nous considérons donc des logiques ayant un opérateur normal ALL pour exprimer les propriétés d'invariants et un opérateur (non-normal) SOME dont le dual INEV permet d'exprimer l'inévitabilité.

Le langage de formules F_{LTA} est défini à partir de la signature

$$\Sigma_{LTA} = \Sigma_{CP} \cup \{SOME, ALL\}$$

sur l'ensemble de variables propositionnelles P, où SOME et ALL sont des opérateurs modaux unaires.

Pour $f \in F_{LTA}$, on définit les abréviations suivantes pour les opérateurs duaux de ALL et SOME:

$$POT(f) := \neg ALL(\neg f)$$

$$INEV(f) := \neg SOME(\neg f)$$

La classe de modèles M_{LTA} contient des modèles $m=(W,R,P)$ de M_{LM}, où R est une relation totale, qui représente une relation d'accessibilité directe.

Pour chaque état $w \in W$ on définit EX(w), l'ensemble des séquences maximales à partir de w via R, c'est-à-dire

$$EX(w) := \{s \in W^{\infty} \mid s = s_0 s_1 \ldots \text{ et } s_0 = w \text{ et } (s_i, s_{i+1}) \in R \ \forall i \in N\}.$$

On note $EX(m) := \bigcup_{w \in W} EX(w)$.

Remarque: La définition de EX(w) nous permet de considérer en même temps la relation d'accessibilité directe R et sa fermeture réflexive et transitive R^*.

La relation de satisfaction $\models_{LTA}$ est définie comme $\models_{LM}$ en II.2.3 par les règles (1) à (3) en ajoutant pour les opérateurs modaux les règles suivantes:

Si $m \in M_{LTA}$, $w \in W$ et $f \in F_{LTA}$, alors

(ALL) $m,w \models ALL(f)$ ssi $\forall s \in EX(w)\ \forall n \in N\ m,s_n \models f$

(SOME) $m,w \models SOME(f)$ ssi $\exists s \in EX(w)\ \forall n \in N\ m,s_n \models f$

Par conséquent, on a pour les duaux

(POT) $m,w \models POT(f)$ ssi $\exists s \in EX(w)\ \exists n \in N\ m,s_n \models f$

(INEV) $m,w \models INEV(f)$ ssi $\forall s \in EX(w)\ \exists n \in N\ m,s_n \models f$

Ceci signifie que

- un état satisfait ALL(f) ssi tous les états s_n de toutes les séquences maximales s partant de w satisfont f, c.à.d. ssi tous les états w' tels que $(w,w') \in R^*$ satisfont f.
- un état w satisfait POT(f) ssi il existe un état s_n, appartenant à une séquence maximale s partant de w, qui satisfait f, c.à.d. ssi il existe un état w' tel que $(w,w') \in R^*$ et w' satisfait f, ssi à partir de w il est 'possible' d'atteindre un état w' satisfaisant f.
- un état w satisfait SOME(f) ssi il existe une séquence maximale s partant de w dont tous les états satisfont f.
- un état w satisfait INEV(f) ssi toutes les séquences maximales s partant de w contiennent un état s_n satisfaisant f, c.à.d. ssi à partir de w il est 'inévitable' en suivant une séquence maximale d'atteindre un état w' satisfaisant f.

Remarque: La signification de ALL(f) et POT(f) correspond à la signification de $\Box f$ et $\Diamond f$ dans le chapitre précédent dans des modèles où la relation d'accessibilité directe R est remplacée par sa fermeture réflexive et transitive R^*, c'est-à-dire dans les modèles de S4. Pour définir la signification de SOME(f) et INEV(f), il est indispensable de tenir compte des deux relations R et R^*.

Dans la proposition suivante, on donne d'une part des formules et règles valides en LTA qui montrent la relation avec les logiques modales

normales, et d'autre part des formules valides permettant de constituer le treillis des modalités.

Proposition 1

Les formules et règles suivantes sont valides en LTA:

(V1) $ALL(f \supset g) \supset (ALL(f) \supset ALL(g))$

(V2) $ALL(f \wedge g) \equiv ALL(f) \wedge ALL(g)$

(V3) $ALL(f) \vee ALL(g) \supset ALL(f \vee g)$

(V4) $POT(f) \vee POT(g) \equiv POT(f \vee g)$

(V5) $POT(f \wedge g) \supset POT(f) \wedge POT(g)$

(V6) $ALL(f) \supset f$

(V7) $ALL(f) \equiv ALL(ALL(f))$

(V8) $f \supset POT(f)$

(V9) $POT(f) \equiv POT(POT(f))$

(V10) $ALLPOTALLPOT(f) \equiv POTALLPOT(f)$

(V11) $POTALLPOTALL(f) \equiv ALLPOTALL(f)$

(R1) $\dfrac{\vdash f}{\vdash ALL(f)}$

(V12) $SOME(f \wedge g) \supset SOME(f) \wedge SOME(g)$

(V13) $INEV(f) \vee INEV(g) \supset INEV(f \vee g)$

(V14) $SOME(f) \supset f$

(V15) $SOME(f) \equiv SOMESOME(f)$

(V16) $f \supset INEV(f)$

(V17) $INEV(f) \equiv INEVINEV(f)$

(V18) $SOME((SOME(g) \vee f) \vee (SOME(f) \vee g)) \equiv SOME(f) \vee SOME(g)$

(V19) $INEV(f) \wedge INEV(g) \equiv INEV(f \wedge INEV(g) \vee g \wedge INEV(f))$

(V20) $SOME(SOME(f \supset SOME(f)) \supset f) \supset (INEVSOME(f) \supset f)$

(V21) $SOME(\neg f \supset INEV(f \wedge INEV(\neg f))) \supset (\neg f \supset SOMEINEV(f \wedge INEV(\neg f)))$

(V22) $INEVSOMEINEV(f) \equiv SOMEINEV(f)$

(V23) $SOMEINEVSOME(f) \equiv INEVSOME(f)$

(R2) $\dfrac{\vdash f \supset g}{\vdash SOME(f) \supset SOME(g)}$

(V24) $ALL(f) \supset SOME(f)$

(V25) $INEV(f) \supset POT(f)$

(V26) $ALL(f \supset g) \supset (SOME(f) \supset SOME(g))$

(V27) $ALL(f) \wedge SOME(g) \supset SOME(f \wedge g)$

(V28) $ALLSOME(f) \equiv ALL(f)$

(V29) $POTINEV(f) \equiv POT(f)$

(V30) $INEVALLINEV(f) \equiv ALLINEV(f)$

(V31) $ALLINEVALL(f) \equiv INEVALL(f)$

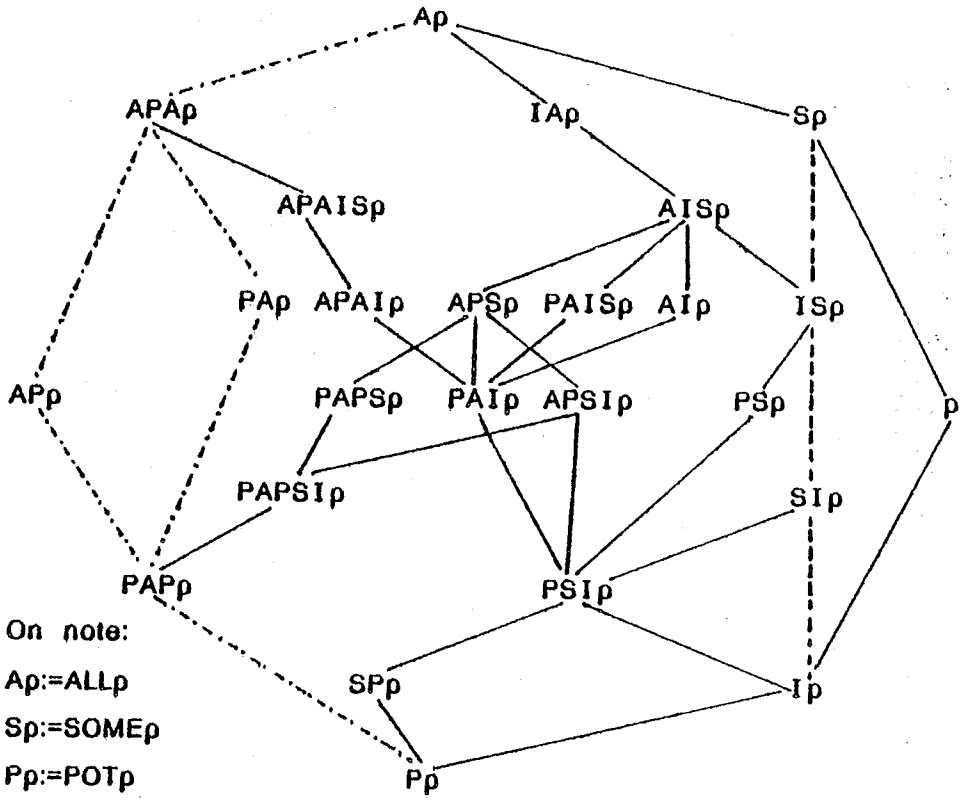
Preuve: Toutes les propriétés peuvent être prouvées facilement à l'aide de la définition de $\models$ et des remarques suivantes. □

Remarque: La validité de (V1) et (R1) nous indiquent que ALL est un opérateur normal. (V2) à (V5) sont valides pour tout opérateur normal. (V6) et (V7) confirment la remarque que ALL correspond à $\Box$ de S4. (V8) à (V11) correspondent à des théorèmes de S4. Le fait que l'impliquation (V12) est propre, montre que SOME n'est pas un opérateur normal, et (V12) et (R2) indiquent que SOME est un opérateur monotone [Ch]. (V13) est le dual de (V12). (V14) et (V15) correspondent à (V6) et (V7), et (V16) et (V17) sont leurs duaux. (V18), (V19) sont les axiomes (L), (L') de S4.3 pour SOME. Etant donné que l'on considère seulement des modèles discrets, on a également la validité de (V20) et (V21) qui sont les axiomes (DL) et (DL') de S4.3.1 pour SOME. Les égalités (V22) et (V23) qui ne sont pas valides pour ALL sont dues à (V18). (V24) à (V29) montrent la relation entre les deux opérateurs modaux ALL et SOME. (V26) est une autre façon d'exprimer la monotonie de SOME.

Remarque: Si l'on a prouvé la validité de (V1), (V6), (V7), (R1) qui correspondent aux axiomes et règles de S4, de (V26), (V27) et (V14), (V15), (V18) et (V20), on peut déduire la validité des autres formules. Ces axiomes et règles ne constituent pas par contre une axiomatisation de LTA, et en [Ca] il est remarqué que LTA n'a probablement pas une axiomatisation finitaire, mais admet une séquence infinie d'axiomes qui sont des généralisations de plus en plus compliquées de (V19).

Il est intéressant d'étudier les modalités indépendantes d'une logique temporelle. Une modalité est une séquence d'opérateurs unaires. En LTA on peut se restreindre aux séquences d'opérateurs modaux parce que toute négation peut toujours être éliminée à l'aide de l'opérateur dual de chaque opérateur modal. Les égalités (V7), (V10) et leur duaux (V9), (V11) garantissent qu'avec l'opérateur ALL et son dual POT on ne peut construire qu'un nombre fini de modalités différentes. On obtient le même résultat pour SOME et INEV à partir des égalités (V15), (V22) et des propriétés duales (V17), (V23). De (V28) à (V31) on déduit que le nombre de modalités 'mixtes' est également fini. Plus précisément, en LTA toute modalité est équivalente à une modalité de longueur inférieure ou égale à cinq. Le treillis des modalités de LTA, qui contient seulement des modalités différentes, est donné ci-dessous. Les implications et égalités utilisées peuvent être déduites des formules valides de la proposition 1. Le fait que les implications sont propres, doit être prouvé à l'aide de contre-exemples.

Le treillis des modalités de LTA:



On note:

Aρ:=ALLρ

Sρ:=SOMEρ

Pρ:=POTρ

Iρ:=INEVρ

III.3 La logique UB

Pour LTA nous n'avons pas pu donner une axiomatisation finitaire. Ben-Ari, Manna et Pnueli ont présenté en [BMP1] une logique **UB** contenant les opérateurs ALL et SOME de LTA. De plus, **UB** contient un opérateur PRE, dont le dual est $\tilde{\text{PRE}}$, exprimant 'il existe un successeur directe'. En **UB** on peut donc exprimer ce qui se passe dans un pas de calcul. En effet, il n'est pas désirable que l'on puisse spécifier ce qui se passe dans un pas de calcul [La2], parce que dans les spécifications on veut précisément faire abstraction de la notion de 'pas de calcul'. Dans les spécifications exprimées en termes de formules de **UB**, il est donc raisonnable de ne pas utiliser les opérateurs PRE et $\tilde{\text{PRE}}$, ce qui revient à donner des spécifications en termes de formules de LTA.

L'adjonction de l'opérateur PRE permet par contre de donner une axiomatisation finitaire de **UB** [BMP2].

Le <u>langage de formules</u> F_{UB} est défini à partir de la signature

$$\Sigma_{UB} = \Sigma_{LTA} \cup \{\text{PRE}\}$$

sur un ensemble de variables propositionnelles **P**, où PRE est un opérateur unaire. Si $f \in F_{UB}$, on définit l'abréviation suivante pour l'opérateur dual de PRE:

$$\tilde{\text{PRE}}(f) := \neg \text{PRE}(\neg f).$$

La <u>classe de modèles</u> M_{UB} est M_{LTA}.

La <u>relation de satisfaction</u> $\models_{UB}$ est une extension de $\models_{LTA}$, obtenue en ajoutant pour les formules de la forme PRE(f) la règle suivante:

Si $f \in F_{UB}$, $m \in M_{LTA}$ et $w \in W$, alors

(PRE) $m,w \models \text{PRE}(f)$ ssi $\exists s \in EX(w)$ $m,s_1 \models f$.

Par conséquent, on a pour le dual

($\tilde{\text{PRE}}$) $m,w \models \tilde{\text{PRE}}(f)$ ssi $\forall s \in EX(w)$ $m,s_1 \models f$.

Ceci signifie que

- un état w satisfait PRE(f) ssi il existe un successeur direct de w qui satisfait f.
- un état w satisfait $\tilde{PRE}$(f) ssi tous les successeurs directs de w satisfont f.

Ainsi, on a défini une interprétation $I_{UB}=(F_{UB},M_{UB},\Vdash_{UB})$.

Définition 1

Considérons le calcul Cal_{UB}, obtenu en ajoutant aux axiomes et règles du CP les axiomes et règles suivants:

(A1) $ALL(f \supset f') \supset (ALL(f) \supset ALL(f'))$
(A2) $\tilde{PRE}(f \supset f') \supset (\tilde{PRE}(f) \supset \tilde{PRE}(f'))$
(A3) $ALL(f) \supset f \wedge \tilde{PRE}ALL(f)$
(A4) $ALL(f \supset \tilde{PRE}(f)) \supset (f \supset ALL(f))$

(E1) $ALL(f \supset f') \supset (SOME(f) \supset SOME(f'))$
(E2) $\tilde{PRE}(f) \supset PRE(f)$
(E3) $SOME(f) \supset f \wedge PRESOME(f)$
(E4) $ALL(f \supset PRE(f)) \supset (f \supset SOME(f))$

(R) $\dfrac{\vdash f}{\vdash ALL(f)}$

(A1) respectivement (A2) correspondent à l'axiome de normalité de II.3 pour ALL respectivement $\tilde{PRE}$. De (A3) et (A4) on déduit que ALL(f) est le plus grand point fixe de $X \equiv \tilde{PRE}(X)$ contenu en f. (A4) peut être considéré comme une loi d'induction.
(E1) est (V26) de III.2 et exprime la monotonie de SOME. (E2) est valide du fait que les relations R des modèles considérés sont totales. (E3) et (E4) correspondent à (A3) et (A4) et permettent de déduire que SOME(f) est le plus grand point fixe de X≡PRE(X) contenu en f. (E4) peut être considéré comme une loi d'induction comme (A4).

Etant donné que l'interprétation des opérateurs modaux ALL et SOME en UB est la même qu'en LTA sur la même classe de modèles, il

est évident que toute formule valide en LTA est valide en UB. On a même que toute formule valide en UB, qui appartient à F_{LTA}, est valide en LTA. Pour la plupart des formules de la proposition 1 de III.2 il est facile de montrer qu'elles sont des théorèmes de Cal_{UB}. La preuve pour (V18) par exemple peut être trouvée en [BMP2]. Après la proposition suivante la preuve pour (V21) est donnée, d'une part parce que cette formule n'est pas mentionnée en [BMP2] et d'autre part parce que sa déduction n'est peut-être pas évidente.
Dans la proposition suivante on donne des formules valides, qui sont également des théorèmes, n'appartenant pas à F_{LTA}.

Proposition 1

Les formules et règles suivants appartiennent à Cal_{UB}.

(T1) $P\tilde{R}E(f \wedge f') \equiv P\tilde{R}E(f) \wedge P\tilde{R}E(f')$

(T2) $P\tilde{R}E(f) \vee P\tilde{R}E(f') \supset P\tilde{R}E(f \vee f')$

(T3) $PRE(f) \vee PRE(f') \equiv PRE(f \vee f')$

(T4) $PRE(f \wedge f') \supset PRE(f) \wedge PRE(f')$

(T5) $P\tilde{R}E(f) \wedge PRE(f') \supset PRE(f \wedge f')$

(T6) $ALL(f) \equiv f \wedge P\tilde{R}EALL(f)$

(T7) $POT(f) \equiv f \vee PREPOT(f)$

(T8) $P\tilde{R}EALL(f) \equiv ALLP\tilde{R}E(f)$

(T9) $SOME(f) \equiv f \wedge PRESOME(f)$

(T10) $INEV(f) \equiv f \vee P\tilde{R}EINEV(f)$

(T11) $SOME(f \supset P\tilde{R}E(f)) \supset (f \supset SOME(f))$

(T12) $PRESOME(f) \supset SOMEPRE(f)$

Pour les modalités $M=\{ALL, SOME, P\tilde{R}E, POT, INEV, PRE\}$ les règles d'inférences

$$(R1^M) \quad \frac{\vdash f}{\vdash M(f)}$$

$$(R2^M) \quad \frac{\vdash f \supset f'}{\vdash M(f) \supset M(f')}$$

Preuve: La plupart des preuves ne sont pas difficiles. A titre d'exemple on peut trouver les preuves pour (T8) et (T9) en [BMP2].

Remarque: Les théorèmes (T1) à (T4) découlent du fait que $\tilde{PRE}$ est un opérateur normal. (T5) peut être déduit de l'axiome (E2). (T6) peut être déduit des axiomes (A3) et (A4), et donne une caractérisation de ALL(f), et (T7), la formule duale, donne une caractérisation de POT(f). De manière analogue, (T9) et (T10) peuvent être déduits de (E3) et (E4). (T11) est une loi d'induction comme (E4).

Maintenant, nous donnons donc la preuve pour (V18) de III.2, qui montre l'utilisation des lois d'induction (A4) et (E4) et des caractérisations (T6), (T7), (T9) et (T10).

Proposition 2

(T13) $SOME(\neg f \supset INEV(f \wedge INEV(\neg f)) \supset (\neg f \supset SOME INEV(f \wedge INEV(\neg f))$

(T14) $ALL(\neg f \supset INEV(f \wedge INEV(\neg f)) \supset (\neg f \supset ALL INEV(f \wedge INEV(\neg f))$

sont des théorèmes de UB.

Preuve: On prouve (T13). On écrit simplement S.I à la place de SOME. INEV.

On note $r := I(f \wedge I(\neg f))$.

(1)	$\vdash \neg f \wedge S(\neg f \supset r)) \supset \neg f \wedge r \wedge S(\neg f \supset r)$	par (V14) et (CP)
(2)	$\vdash \neg f \wedge r \wedge S(\neg f \supset r) \supset PRE(r \wedge S(\neg f \supset r))$	par (E3), (T10) et (T5)
(3)	$\vdash f \wedge r \wedge S(\neg f \supset r) \supset PRE(I(\neg f) \wedge S(\neg f \supset r))$	par (E3), (T10) et (T5)
(4)	$\vdash f \wedge I(\neg f) \wedge S(\neg f \supset r) \supset r \wedge S(\neg f \supset r)$	par (V16)
(5)	$\vdash \neg f \wedge I(\neg f) \wedge S(\neg f \supset r) \supset \neg f \wedge r \wedge S(\neg f \supset r)$	par (1) et (CP)
(6)	$\vdash I(\neg f) \wedge S(\neg f \supset r) \supset r \wedge S(\neg f \supset r)$	par (4),(5) et (CP)
(7)	$\vdash f \wedge r \wedge S(\neg f \supset r) \supset PRE(r \wedge S(\neg f \supset r))$	par (3),(6),(R2) et (CP)
(8)	$\vdash r \wedge S(\neg f \supset r) \supset PRE(r \wedge S(\neg f \supset r))$	par (2),(7) et (CP)
(9)	$\vdash r \wedge S(\neg f \supset r) \supset S(r \wedge S(\neg f \supset r))$	par (8),(R),(E4) et (CP)
(10)	$\vdash r \wedge S(\neg f \supset r) \supset S(r)$	par (9) et (CP)
(11)	$\vdash \neg f \wedge S(\neg f \supset r) \supset S(r)$	par (1),(10) et (CP)
(12)	$\vdash S(\neg f \supset r) \supset (\neg f \supset S(r))$	par (11) et (CP) q.e.d.

La preuve de (T14) est tout à fait analogue, où PRE est remplacé par $\tilde{PRE}$, l'utilisation de (T5) par (T1), (E4) par (A4) etc. □

UB a une infinité de modalités indépendantes, parce qu'il n'existe aucune 'loi de réduction' pour les opérateurs PRE et $\tilde{\text{PRE}}$. Pour les modalités construites à partir des opérateurs ALL et SOME on obtient évidemment le même treillis que pour LTA.

On a le théorème suivant.

Théorème 1 [BMP2]

La logique $UB=(Cal_{UB}, I_{UB})$ est consistante, complète et décidable. □

La procédure de décision présentée dans le paragraphe suivant est une procédure de décision pour UB et également pour LTA.

III.4 La logique du temps arborescent conditionnelle LTAC

Les logiques LTA et UB ne sont pas suffisamment puissantes pour exprimer certaines propriétés intéressantes de programmes. En particulier, elles ne permettent pas d'exprimer des propriétés du type "il est possible (respectivement inévitable) d'atteindre un état satisfaisant une propriété p en maintenant vraie une condition c jusqu'à ce que p soit satisfait", c'est-à-dire des propriétés d'atteignabilité inévitable ou possible sous une condition.
Les propriétés de ce type n'étant pas exprimables en UB, nous introduisons l'extension LTAC de UB [QS2] et donnons une axiomatisation complète de LTAC et une procédure de décision, généralisation de celle donnée en [BMP2] pour UB. Cette procédure est également une procédure de décision pour UB, parce que les formules valides de la restriction F_{UB} de F_{LTAC} sont exactement les formules valides de F_{UB}. La logique LTAC est également la logique CTL de [CE] et [EH1]. En [EH2] plusieurs extensions de LTAC sont proposées.

4.1 La logique LTAC

Le langage de formules F_{LTAC} est défini à partir de la signature $\Sigma_{LTAC} = \Sigma_{UB}$ sur un ensemble de variables propositionnelles P, où ALL et SOME sont des opérateurs binaires.

Si $g,f \in F_{LTAC}$, on définit les abréviations suivantes pour les opérateurs duaux de ALL et SOME par:

$$POT(g,f) := \neg ALL(\neg g, \neg f)$$
$$INEV(g,f) := \neg SOME(\neg g, \neg f)$$

La classe de modèles M_{LTAC} est M_{LTA}.

La relation de satisfaction $\models_{LTAC}$ est définie comme $\models_{LM}$ en II.2.3 par les règles (1) à (3) en ajoutant pour les opérateurs modaux les règles

suivantes:

Si $m \in M_{LTAC}$, $w \in W$ et $g,f \in F_{LTAC}$, alors

(ALL) $m,w \models ALL(g,f)$ ssi $\forall s \in EX(w)\ \forall n \in N\ (\forall i<n\ m,s_i \models \neg g$ implique $m,s_n \models f)$

(SOME) $m,w \models SOME(g,f)$ ssi $\exists s \in EX(w)\ \forall n \in N\ (\forall i<n\ m,s_i \models \neg g$ implique $m,s_n \models f)$

(PRE) $m,w \models PRE(f)$ ssi $\exists s \in EX(w)\ m,s_1 \models f$

Par conséquent, on obtient pour les duaux

(POT) $m,w \models POT(g,f)$ ssi $\exists s \in EX(w)\ \exists n \in N\ (\forall i<n\ m,s_i \models g$ et $m,s_n \models f)$

(INEV) $m,w \models INEV(g,f)$ ssi $\forall s \in EX(w)\ \exists n \in N\ (\forall i<n\ m,s_i \models g$ et $m,s_n \models f)$

($\tilde{PRE}$) $m,w \models \tilde{PRE}(f)$ ssi $\forall s \in EX(w)\ m,s_1 \models f$.

Les interprétations de PRE et $\tilde{PRE}$ sont donc les mêmes que pour UB. Pour les autres opérateurs modaux on a

- Un état satisfait ALL(g,f) ssi tout état s_n de toute séquence maximale s partant de w satisfait f, tant que tous les états s_i de la séquence s qui précèdent s_n satisfont la "condition" $\neg g$.
- Un état w satisfait SOME(g,f) ssi tout état s_n d'une séquence maximale s partant de w satisfait f, tant que tous les états s_i de s précédant s_n satisfont $\neg g$.
- Un état w satisfait POT(g,f) ssi il existe une séquence maximale s partant de w dont un état s_n satisfait f, et tous les états s_i de s précédant s_n satisfont g, c'est-à-dire ssi à partir de w il est <u>possible</u> d'atteindre un état w' qui satisfait f en maintenant la condition g vérifiée pour tous les états précédant w' sur un chemin de w à w'.
- Un état satisfait INEV(g,f) ssi toutes les séquences maximales s partant de w contient un état s_n qui satisfait f et tous les états s_i de s précédant s_n satisfont g, c'est-à-dire ssi à partir de w il est <u>inévitable</u> d'atteindre un un état w' qui satisfait f en passant par un chemin de w à w' dont tous les états précédant w' satisfont la condition g.

Ainsi, on a défini une interprétation $I_{LTAC} = (F_{LTAC}, M_{LTAC}, \models_{LTAC})$.

L'interprétation I_{LTAC} est donc une "généralisation" de I_{UB}, parce que, comme on voit facilement, les opérateurs modaux unaires de UB peuvent être définis de la manière suivante:

Définition 1

$$ALL(f) := ALL(\bot,f)$$
$$POT(f) := POT(T,f)$$
$$SOME(f) := SOME(\bot,f)$$
$$INEV(f) := INEV(T,f)$$

Considérons le calcul Cal_{LTAC}, obtenu en ajoutant aux axiomes et règles du CP les axiomes et règles suivants:

(A1') $ALL(g,f\supset f') \supset (ALL(g,f)\supset ALL(g,f'))$
(A2') $P\tilde{R}E(f\supset f') \supset (P\tilde{R}E(f)\supset P\tilde{R}E(f'))$
(A3') $ALL(g,f) \supset f\wedge(\neg g\supset P\tilde{R}EALL(g,f))$
(A4') $ALL(\bot,f\wedge\neg g\supset P\tilde{R}E(f)) \supset (f\supset ALL(g,f))$
(A5') $ALL(\bot,g\supset g') \supset (ALL(g,f)\supset ALL(g',f))$

(E1') $ALL(g,f\supset f') \supset (SOME(g,f)\supset SOME(g,f'))$
(E2') $P\tilde{R}E(f) \supset PRE(f)$
(E3') $SOME(g,f) \supset f\wedge(\neg g\supset PRESOME(g,f))$
(E4') $ALL(\bot,f\wedge\neg g\supset PRE(f)) \supset (f\supset SOME(g,f))$
(E5') $ALL(\bot,g\supset g') \supset (SOME(g,f)\supset SOME(g',f))$

(R') $\dfrac{\vdash f}{\vdash ALL(\bot,f)}$

Visiblement, on peut déduire les axiomes et règles de Cal_{UB} à partir de cette axiomatisation en substituant toujours $\bot$ pour g et en utilisant ensuite les abréviations définies précédemment. Les axiomes (A5') et (E5') signifient que ALL et SOME sont monotones pour le premier argument. Cal_{UB} est donc contenu en Cal_{LTAC}. Dans la proposition suivante, nous donnons donc seulement des théorèmes et règles de LTAC qui n'appartiennent pas à F_{UB}, mais qui peuvent être vus comme des généralisations des théorèmes et règles de UB, donnés dans la proposition 1 de III.3.

Proposition 1

Les formules et règles suivants appartiennent à Cal_{LTAC}.

(U1) $ALL(g,f) \wedge ALL(g,f') \equiv ALL(g,f \wedge f')$

(U2) $ALL(g,f) \equiv f \wedge (\neg g \supset P\tilde{R}EALL(g,f))$

(U3) $POT(g,f) \equiv fv(g \wedge PREPOT(g,f))$

(U4) $ALL(g,f) \equiv ALL(g,ALL(g,f))$

(U5) $ALL(g \wedge g',f) \supset ALL(g,f) \wedge ALL(g',f)$

(U6) $ALL(\perp,f) \equiv ALL(f,f)$

(U7) $SOME(g,f \wedge f') \supset SOME(g,f) \wedge SOME(g,f')$

(U8) $SOME(g,f) \equiv f \wedge (\neg g \supset PRESOME(g,f))$

(U9) $INEV(g,f) \equiv fv(g \wedge P\tilde{R}EINEV(g,f))$

(U10) $SOME(g,f) \equiv SOME(g,SOME(g,f))$

(U11) $INEV(g,f) \wedge INEV(g',f') \equiv INEV(g \wedge g', f \wedge INEV(g',f') v f' \wedge INEV(g,f))$

(U12) $SOME(\perp, f \wedge \neg g \supset P\tilde{R}E(f)) \supset (f \supset SOME(g,f))$

(U13) $SOME(g \wedge g',f) \supset SOME(g,f) \wedge SOME(g',f)$

(U14) $SOME(\perp,f) \equiv SOME(f,f)$

(U15) $ALL(g,f) \supset SOME(g,f)$

(U16) $ALL(g,f) \wedge SOME(g',f') \supset SOME(g \wedge g', f \wedge f')$

(U17) $INEV(g,ALL(g,f)) \supset ALL(\neg g, INEV(g,f))$

Pour $M \in \{ALL, SOME, INEV, POT\}$

$(R1^M)$ $\dfrac{\vdash f}{\vdash M(g,f)}$

$(R2^M)$ $\dfrac{\vdash f \supset f', \vdash g \supset g'}{\vdash M(g,f) \supset M(g',f')}$

$(R3^A)$ $\dfrac{\vdash f \wedge \neg g \supset P\tilde{R}E(f)}{\vdash f \supset ALL(g,f)}$

$(R3^S)$ $\dfrac{\vdash f \wedge \neg g \supset PRE(f)}{\vdash f \supset SOME(g,f)}$

Preuve: Les preuves sont analogues aux preuves des théorèmes correspondants de UB. Les preuves, pour les propriétés concernant les premiers arguments des modalités binaires, sont obtenues à partir de (A5') et (E5'). (U6) et (U14) sont obtenus en utilisant (A5') respectivement

(E5') dans un sens, et (U2) et (A3') respectivement (U8) et (E3') dans le sens inverse. Les règles (R3) sont obtenues à partir de (R') et (A4') respectivement (E4') et (MP). □

Une modalité a été définie comme une séquence d'opérateurs unaires. En LTAC, on peut définir des modalités en fixant un argument d'un opérateur binaire à une constante T ou ⊥ comme en définition 1. Etant donné que

$$ALL(T,f) \equiv f \equiv POT(\bot,f),$$

$$ALL(g,\bot) \equiv POT(g,\bot) \equiv \bot \text{ et}$$

$$ALL(g,T) \equiv POT(g,T) \equiv T$$

et de manière analogue pour les opérateurs SOME et INEV, les opérateurs unaires définis en définition 1 sont les seuls que l'on puisse définir raisonnablement. Ce sont les modalités de LTA, et on obtient le même treillis de modalités que pour LTA.

4.2 La procédure de décision de LTAC

La procédure de décision de LTAC est une extension de la procédure de décision pour **UB**, présentée en [BMP2]. Pour une formule f donnée, cet algorithme effectue une recherche systématique d'un contre-modèle pour f, et ne fournit pas un tel modèle ssi f est une formule valide.

Definition 2

Une structure est un triplet S = (S,R,*F*) dont la seule différence à un modèle est que pour $s \in S$ $F(s) \subseteq F_{LTAC}$ n'est pas nécessairement complet, c'est-à-dire $p \in P$ et $p \notin F(s)$ n'implique pas nécessairement $\neg p \in F(s)$.

Notation: On écrit $f \in s$ pour $f \in F(s)$.

Definition 3

Une formule $f \in F_{LTAC}$ est en forme positive si les seules occurrences de la négation en f sont appliquées à des variables propositionnelles. Ceci est toujours possible en utilisant les opérateurs duaux des opérateurs primitifs.
(Par exemple $\neg$(ALLpvSOMEPOT(q,p)) $\equiv$ POT($\neg$p)$\wedge$INEVALL($\neg$q,$\neg$p).)

Dans l'algorithme suivant on ne considère que des formules en forme positive. Ceci et les caractérisations (U2), (U3), (U8) et (U9) permettent de classifier toutes les formules en formules primitives, α-formules et β-formules.
Les formules primitives sont les variables propositionnelles ou leurs négations et les "X-formules" de la forme PRE(f) ou $\tilde{\text{PRE}}$(f). Les α-formules sont de la forme $\alpha' \wedge \alpha''$ et les β-formules de la forme $\beta' \vee \beta''$.

Définition 4

On classifie toute formule positive, non-primitive, en α- ou β-formules de la manière suivante:

α	α'	α''
$f \wedge f'$	f	f'
$ALL(g,f)$	f	$g \vee P\tilde{R}EALL(g,f)$
$SOME(g,f)$	f	$g \vee PRESOME(g,f)$

β	β'	β''
$f \vee f'$	f	f'
$POT(g,f)$	f	$g \wedge PREPOT(g,f)$
$INEV(g,f)$	f	$g \wedge P\tilde{R}EINEV(g,f)$

Remarque: On a pour toute α-formule α, $\alpha \equiv \alpha' \wedge \alpha''$ et pour toute β-formule β, $\beta \equiv \beta' \vee \beta''$.

Définition 5

Une structure $H = (H,R,F)$ est une structure de Hintikka pour LTAC ssi $\forall s \in H$:

(H1) $\neg p \in s$ implique $p \notin s$ (consistance)
(H2) $\alpha \in s$ implique $\alpha' \in s$ et $\alpha'' \in s$, où α une α-formule
(H3) $\beta \in s$ implique $\beta' \in s$ ou $\beta'' \in s$, où β une β-formule
(H4a) $P\tilde{R}E(f) \in s$ implique $\forall t \in H$ (sRt implique $f \in t$)
(H4b) $PRE(f) \in s$ implique $\exists t \in H$ (sRt et $f \in t$)
(H5a) $POT(g,f) \in s$ implique $\exists x = s_0 \ldots \in H^{\infty}$, $s_0 = s$, $\exists n$ $s_n \models f$ et $\forall i < n$ $s_i \models g$
(H5b) $INEV(g,f) \in s$ implique $\forall x = s_0 \ldots \in H^{\infty}, s_0 = s$, $\exists n$ $s_n \models f$ et $\forall i < n$ $s_i \models g$

H est une structure de Hintikka pour f ssi $\exists s \in H$ tel que $f \in s$.

Proposition 2

Une formule $f \in F_{LTAC}$ est satisfaisable ssi il existe une structure de Hintikka pour f.

Preuve:

'=>' Evidemment, tout modèle est une structure de Hintikka.

'<=' Soit H=(H,R,$\mathcal{F}$) une structure de Hintikka pour f. Elle peut être étendue à un modèle m=(H,R,$\mathcal{P}$) par: $\forall s \in H$, $\forall p \in P$

$p \in \mathcal{P}(s)$ ssi $p \in \mathcal{F}(s)$ ou $\neg p \notin \mathcal{F}(s)$

$\neg p \in \mathcal{P}(s)$ ssi $\neg p \in \mathcal{F}(s)$

Preuve par induction sur la structure de formules en forme positive:

(1) $f \in s$ implique $m,s \models f$

(2) $\neg f \in s$ implique $m,s \models \neg f$

Ceci est vrai pour $f \in P$ par construction de m. La preuve structurelle est facile à établir par la définition de la relation de satisfaction pour les formules de la forme $f \wedge f'$, fvf' PRE(f), $\widetilde{PRE}$(f), POT(f), INEV(f), ALL(f), SOME(f) et les caractérisations (U2), (U3), (U8) et (U9). □

Notations:

1. Un tableau T=(T,R,$\mathcal{F}$) est comme une structure où $\mathcal{F}$(s) est un ensemble de formules positives quelconques.
2. On appelle $s \in T$ un noeud de T.
3. Pour $s \in T$, on écrit VF(s) pour $\bigvee_{f \in \mathcal{F}(s)} f$ et $\Lambda F(s)$ pour $\bigwedge_{f \in \mathcal{F}(s)} f$.

Supposons, que l'on veuille prouver la validité d'une formule $f \in F_{LTAC}$. On essaye alors de construire un modèle pour $\neg f$. D'après la proposition 2, il suffit pour ceci de construire une structure de Hintikka pour $\neg f$. Elle sera obtenue à partir d'un tableau T, obtenu en appliquant à un noeud initial s_0, tel que $F(s)=\{\neg f\}$, les "règles" (H1) à (H4) de la définition 5 jusqu'à stabilisation. Pour la structure que l'on obtient à partir de T, il reste à vérifier (H5).

Définition 6 (construction d'un tableau pour f)

Soit f une formule en forme positive. Pour garantir que la relation R du tableau construit est totale, on peut remplacer f par $f \wedge ALL(\bot,T)$, une formule équivalente à f.

1. Soit s_0 la racine d'un tableau T. On définit $F(s_0):=\{f\}$.
2. (α-règle)
   Si $s\in T$ et $\alpha\in F(s)$, une α-formule à laquelle la α-règle n'a pas encore été appliquée, on crée un successeur s_1 de s, où $F(s_1)=F(s)\cup\{\alpha',\alpha''\}$.
   On appelle s un <u>α-noeud</u> pour α.
3. (β-règle)
   Si $s\in T$ et $\beta\in F(s)$, une β-formule à laquelle la β-règle n'a pas encore été appliquée, on crée deux successeurs s_1 et s_2 de s, où $F(s_1)=F(s)\cup\{\beta'\}$ et $F(s_2)=F(s)\cup\{\beta''\}$.
   On appelle s un <u>β-noeud</u> pour β.
4. Soit $s\in T$ un noeud pour lequel les règles (2) et (3) ne sont plus appliquables et tel que $X_s=\{PRE(f_1),\ldots PRE(f_k),P\tilde{R}E(g_1),\ldots P\tilde{R}E(g_l)\}$ est l'ensemble des X-formules de s. On crée k successeurs s_i de s tels que $F(s_i)=\{f_i,g_1,\ldots g_l\}$ pour $i=1\ldots k$.
   On appelle s un <u>X-noeud</u>.

<u>Définition 7</u> (règles de terminaison)

(T1) Si s est un noeud tel que $p\in F(s)$ et $\neg p\in F(s)$ pour une formule primitive $p\in P$, s est <u>fermé</u> et on n'applique plus de règles à s.

(T2) Si s_i est un successeur d'un X-noeud s, s est descendant d'un noeud r_i, successeur direct d'un X-noeud r, et $F(s_i)=F(r_i)$, alors on enlève s_i et ajoute un arc de s à r_i.

<u>Proposition 3</u> (terminaison)

L'algorithme de construction de tableau se termine.

<u>Preuve</u>: Si n est le nombre de sous-formules de f, il n'existe pas plus de 2^n noeuds différents. Alors, sur toute branche du tableau (T2) doit être appliqué dans une profondeur finie, dans le cas où (T1) n'est pas appliqué. □

L'algorithme de marquage suivant élimine tous les noeuds qui ne peuvent pas être complétés pour satisfaire (H1) à (H4).

Définition 8 (algorithme de marquage provisoire)

On applique les règles de marquage suivantes jusqu'à stabilisation et obtention d'un tableau T'.

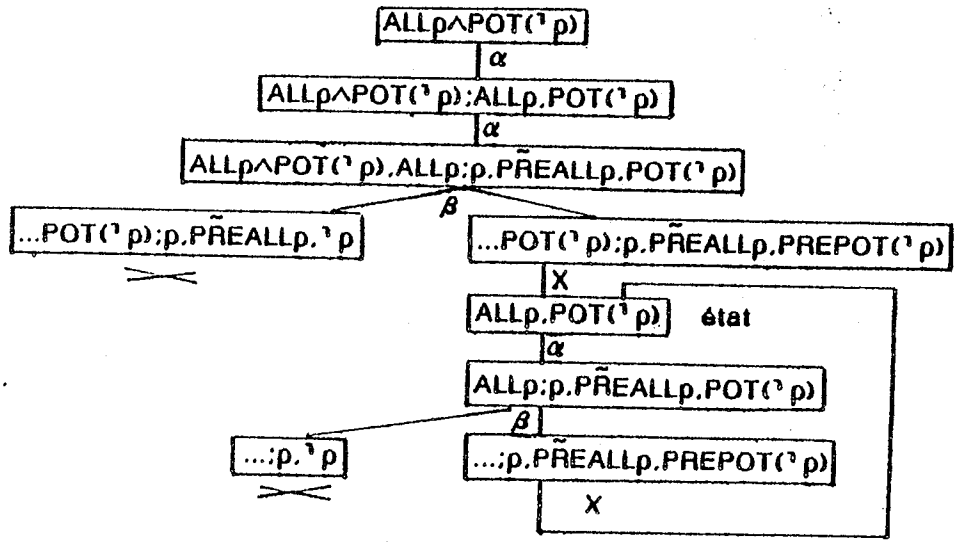
M1. On marque tous les noeuds fermés.
M2. On marque un α-noeud s si s_1 est marqué.
M3. On marque un β-noeud s si s_1 et s_2 sont marqués.
M4. On marque un X-noeud s si un des s_i est marqué.

Définition 9 (une structure provisoire)

On définit une structure S=(S,R,F) à partir de T':

1. S est l'ensemble des X-noeuds de T', c'est-à-dire des X-noeuds non marqués, que l'on appelle aussi des états.
2. Pour s,rϵS on a sRr ssi il existe un chemin $ss_1...s_nr$ en T' où $s_1,....,s_n$ sont des α- et β-noeuds.
3. pϵF(s) ssi pϵF(s).

La structure S, ainsi définie, satisfait apparemment les conditions (H1) à (H4) de la définition 5, mais pas nécessairement (H5). Par exemple le tableau ci-dessous pour ALL(p)$\wedge$POT($\neg$p) contient un circuit non marqué dans lequel la satisfaction de POT($\neg$p) est infiniment ajournée.



Pour déterminer des inconsistances de cette nature on associe un rang rg(h,s) aux occurences des "formules futures" de la forme POT(f,g), INEV(f,g), PREPOT(g,f) ou $\tilde{P}$REINEV(g,f) dans un état s.

Définition 10 (algorithme d'association de rang rg)

On applique les règles d'association de rang suivantes jusqu'à stabilisation. Soit rg(h,s) ne pas encore défini, alors

(R1) Si h est de la forme POT(g,f) ou INEV(g,f), s est un β-noeud pour h et s_1 n'est pas marqué, alors rg(h,s):=1.

(R2) Si h est de la forme POT(g,f) ou INEV(g,f), s est un β-noeud pour h, s_1 est marqué et rg(h',s_2) est défini - où h'=PREPOT(g,f) ou h'=$\tilde{P}$REINEV(g,f) respectivement - alors rg(h,s):=rg(h',s_2)+1.

(R3) Si s est un β-noeud non pour h et rg(h,s_i) est défini pour i=1 ou i=2, alors rg(h,s):=rg(h,s_i)+1.

(R4) Si s est un α-noeud et rg(h,s_1) est défini, alors rg(h,s):=rg(h,s_1)+1.

(R5POT) Si h=PREPOT(g,f), s est un X-noeud et rg(POT(g,f),s_i) est défini pour un successeur s_i, alors rg(h,s):=rg(POT(g,f),s_i)+1.

(R5INEV) Si h=$\tilde{P}$REINEV(g,f), s est un X-noeud et rg(INEV(g,f),s_i) est défini pour tous les successeurs s_i, alors rg(h,s):=max[rg(INEV(g,f),s_i)]+1.

Cet algorithme se termine parce que le nombre de formules futures h est fini et tout rg(h,s) n'est défini qu'une fois.
Dans la règle (R5INEV), on ne tient pas compte de la condition g de INEV(g,f) parce que tout noeud contenant $\tilde{P}$REINEV(g,f) contient également g. Ceci, parce que $\tilde{P}$REINEV(g,f) ne peut être obtenu qu'en appliquant la α-règle à g$\wedge\tilde{P}$REINEV(g,f), une formule obtenue par application de la β-règle à INEV(g,f).
On peut faire une remarque analogue pour la règle (R5POT).

Proposition 4

(1) $rg(h.s)=1$ ssi s est un β-noeud pour h et s_1 n'est pas marqué.
(2) Si s est un β-noeud pour h, s_1 est marqué et s_2 n'est pas marqué, alors $rg(h.s_2)<rg(h.s)$.
(3) Si s est un β-noeud non pour h (un α-noeud), alors $rg(h.s_i)<rg(h.s)$ pour un (le seul) successeur de s.
(4) Si s est un X-noeud, alors $rg(INEV(g.f).s_i)<rg(INEV(g.f).s)$ pour tout successeur s_i de s et $rg(POT(g.f).s_i)<rg(POT(g.f).s)$ pour le successeur s_i de s, défini pour PREPOT(g.f).

Preuve: Evidente. □

Maintenant, on complète l'algorithme de marquage, défini précédemment, de telle sorte que l'on obtienne un tableau à partir duquel on peut définir une structure de Hintikka.

Définition 10 (algorithme de marquage)

- On applique (M1) à (M4) de la définition 8 jusqu'à stabilisation.
- (M5): On applique l'algorithme d'association de rang et on marque tout noeud s contenant une formule future h telle que rg(h.s) n'est pas défini.
- Si par (M5) un noeud a été marqué, on réapplique de nouveau (M1) à (M4) et ainsi de suite jusqu'à stabilisation et obtention d'un tableau définitif T'.

Cet algorithme se termine parce qu'à chaque étape au moins un noeud est éliminé.

Théorème 1

Soit T' le tableau définitif, obtenu par l'algorithme précédent en mettant $F(s_0)=\{f\}$. Alors, la racine s_0 n'est pas marquée ssi il existe une structure de Hintikka (finie) pour f, c'est-à-dire f est satisfaisable.

Ceci signifie que l'algorithme décrit est une procédure de décision pour LTAC, et LTAC a la propriété de modèles finis.

Preuve:

<=: Une conséquence du théorème de complétude, donné en 2.3.

=>: Une conséquence de l'algorithme de construction de contre-modèle, donné ci-dessous. □

La structure S obtenue à partir du tableau définitif T' n'est toujours pas nécessairement une structure de Hintikka, parce qu'il peut toujours y avoir des circuits dont tous les états contiennent les formules INEV(g,f) et $\neg$f. Mais ces circuits peuvent toujours être quittés par un choix alternatif à un β-noeud, sinon le circuit serait marqué inconsistant. L'algorithme proposé effectue un "déroulement" de la structure S', tel que de tels circuits sont parcourus seulement un nombre fini de fois.

Définition 12 (algorithme de déroulement)

Cet algorithme travaille sur le tableau définitif T', obtenu par l'algorithme de marquage, qui ne contient que des noeuds non-marqués. Si $s_0 \in T'$, on construit un tableau T* de la manière suivante:

Notations:

Si $s \in T'$, alors s', s* etc. sont des occurrences de s en T*.

ch(s',r') est le chemin de s' à r' qui ne contient pas r'.

Un noeud alternatif est un β-noeud dont les deux successeurs appartiennent à T'.

T* est obtenu à partir de T' en appliquant les règles suivantes:

(D0) $s'_0 \in T^*$.

(D1) Si s n'est pas un noeud alternatif et $s' \in T^*$, alors pour tout successeur (direct) s_j de s s'_j est un successeur de s' en T*.

(D2) Si s est un noeud alternatif, $s' \in T^*$ et k le nombre d'occurrences de s en $ch(s'_0,s')$, alors

(1) Si k=0, le successeur de s' en T* soit s'_j (arbitrairement).

(2) Si k>0 et s''_1 (s''_2) était le successeur de la k-1-ième occurrence de s en $ch(s'_0,s')$ le successeur de s' est s'_2 (s'_1). C'est-à-dire, on choisit alternativement s_1 et s_2.

(D3) Si s' est une occurrence de s en T" et s" une occurrence précédente sur $ch(s'_0,s')$, et tout noeud alternatif apparaissant en ch(s",s') y a au moins deux occurrences, c'est-à-dire tout successeur alternatif a été choisi au moins une fois, alors on identifie s' et s".

Il est évident que cet algorithme se termine, parce que le nombre de noeuds alternatifs est fini. Si un noeud s a un nombre infini d'occurrences dans une branche, alors il en est de même pour ses successeurs directs et ainsi de suite. On obtient donc:

Proposition 5

La structure S" obtenue à partir de T" est une structure de Hintikka (pour f).

Preuve: Il reste à vérifier (H5). Pour tout état s'ϵS" on a:
Si PR̃EINEV(g,f) ou PREPOT(g,f) appartient à s et f n'appartient pas à s, alors g appartient à s. Pour vérifier (H5), on ne doit donc pas tenir compte de g et la preuve est celle donnée en [BMP2] pour vérifier la satisfaction de INEV(f) et de POT(f). □

4.3 La complétude de LTAC

Théorème 3 (complétude)

La logique LTAC définie par LTAC=(Cal_{LTAC}, I_{LTAC}) est consistante et complète.

Preuve:

1) Comme nous avons déjà mentionné, les axiomes et règles de Cal_{LTCA} sont valides. D'où on obtient la consistance.

2) Pour montrer la complétude, c'est-à-dire que toute formule valide est un théorème, on associe à chaque noeud s du tableau T' une formule $f(s)=\neg\Lambda F(s)$, i.e. $f(s)=\bigvee_{f\in F(s)}\neg f$. Pour $F(s_0)=\{\neg f\}$ on obtient donc $f(s_0)=f$. On sait que si f est valide, alors la racine s_0 est marquée. Si l'on peut prouver que la formule f(s) pour tout noeud marqué est un théorème, on obtient:

$$\forall f\in F_{LTAC}\ \models f \text{ implique } \vdash f,$$

ce qui complète la preuve. La preuve de $\vdash f(s)$ pour tout noeud s marqué est obtenue par induction sur la structure du tableau T' à partir des quatre lemmes suivants.

Lemme 1

Si s est un noeud fermé, alors $\vdash f(s)$.

Lemme 2

Si s est un α-,(β-),(X-)noeud et $\vdash f(s_1)$, ($\vdash f(s_1)$ et $\vdash f(s_2)$), ($\vdash f(s_i)$ pour un i), alors $\vdash f(s)$.

Lemme 3

Si t est un état, marqué parce que rg(PREPOT(f,g),t) n'est pas défini, alors $\vdash f(t)$.

Lemme 4

Si t est un état, marqué parce que rg(PR̃EINEV(g,f),t) n'est pas défini, alors ⊢f(t).

Il est évident que l'on peut déduire des lemmes 1 à 4 :

s_0 marqué implique ⊢f(s_0),

ce qui complète la preuve de la complétude.

Il reste donc à prouver les lemmes:

Le lemme 1 peut être prouvé par un raisonnement purement propositionnel comme en [BMP2].

Le lemme 2 peut être prouvé exactement comme le lemme correspondant en [BMP2], si l'on utilise les caractérisations (U2), (U3), (U8) et (U9) à la place de (T6), (T7), (T9) et (T10) de III.3.

Preuve du lemme 3

Soit [t]* les états accessibles à partir de t en prenant aux X-noeuds le successeur contenant POT(g,f). Ainsi, tous les états s∈[t]* contiennent h=PREPOT(g,f) et rg(h,s) n'est pas défini.

G(s)= $\{f_i \mid \tilde{PRE}(f_i \in F(s))\}$

W(s)= ΛG(s) .

$W' = \bigvee_{s \in [t]^*} W(s)$

[s] les états de [t]*, accessibles à partir de s en passant seulement par des α- et β-noeuds, c'est-à-dire les états successeurs directs.

Soit s∈[t]* et s_1 le noeud créé à partir de s pour PREPOT(g,f). Alors, $F(s_1)$=F(s)∪{POT(g,f)} et tout chemin en [t]* passe par s_1.

- Pour tout α- ou β-noeud r on a:
  Si r est un α-noeud pour $\alpha=\alpha'\wedge\alpha''$, alors
  $\Delta F(r_1)\equiv\Delta F(r)\wedge\alpha'\wedge\alpha''\equiv\Delta F(r)$.
  Si r est un β-noeud pour $\beta=\beta'\vee\beta''$, alors
  $\Delta F(r_1)\vee\Delta F(r_2)\equiv\Delta F(r)\wedge\beta'\vee\Delta F(r)\wedge\beta''\equiv\Delta F(r)$.
  Ainsi, on obtient
  (1) $\vdash\Delta F(s_1) \equiv \bigvee_{r\in[s]}\Delta F(r)$ par la définition de [s].

- Pour tout $s\in[t]^*$ $h=PREPOT(g,f)\in s$ et rg(h,s) n'est pas défini. Sans perte de généralité on peut supposer que la première règle appliquée en s_1 est la β-règle pour $POT(g,f)\equiv fvg\wedge PREPOT(g,f)$ et s_{1_1} est inconsistant (sinon rg(h,s) serait défini). Ceci implique:
  a) $\vdash f(s_{1_1})$ par induction, i.e.
  $\vdash\neg f(s_{1_1})\vee\neg f$ équivalent à
  $\vdash W(s)\supset\neg f$ (2).
  Parce que (2) est vrai pour tout $s\in[t]^*$, alors
  $\vdash W'\supset\neg f$ (3) ce qui implique par $(R2^{\wedge})$
  $\vdash ALL(\neg g,W')\supset ALL(\neg g,\neg f)$ (4)

  b) Tous les $s'\in[s]$ sont des successeurs de s_{1_2}. $F(s_{1_2})$ contient la α-formule $g\wedge PREPOT(g,f)$ et sans perte de généralité on peut supposer que s_{1_2} est un α-noeud pour cette formule et r est son successeur, donc $F(r)-\{PREPOT(g,f)\}=G(s)\cup\{g\}$. Ceci implique que tous les successeurs de s_{1_2}, inclus les états $s'\in[s]$ contiennent la formule g ou une formule g' telle que $g'\supset g$ (comme en (a)) et la formule primitive PREPOT(g,f). On a
  $F(r)-\{PREPOT(g,f)\} = G(s)\cup\{g\}$. Ainsi, on obtient par (1)
  $\vdash g\wedge\Delta G(s) \supset \bigvee_{s'\in[s]}\Delta(F(s')-\{PREPOT(g,f)\})$.
  Parce que $\{\tilde{PRE}(f_i)\mid f_i\in G(s')\}\subseteq F(s')-\{PREPOT(g,f)\}$, alors
  $\vdash g\wedge\Delta G(s) \supset \bigvee_{s'\in[s]}\bigwedge_{f_i\in G(s')}\tilde{PRE}(f_i)$.
  Par (T1) de III.3 et la définition de W(s), on obtient
  $\vdash g\wedge W(s) \supset \bigvee_{s'\in[s]}\tilde{PRE}(W(s'))$ (5)

Parce que (5) est vrai pour tout $s\in[t]^*$, on a

$\vdash g \wedge W' \supset \bigvee_{s \in [t]^*} P\tilde{R}E(W(s))$, ce qui implique par (T2) et la définition de W'

$\vdash g \wedge W' \supset P\tilde{R}E(W')$. Par (R'), (A4') et (MP) on obtient

$\vdash W' \supset ALL(\neg g, W')$. Par (4) et (CP) on obtient

$\vdash W' \supset ALL(\neg g, \neg f)$. Du fait que $\vdash W(t) \supset W'$ on obtient

$\vdash W(t) \supset ALL(\neg g, \neg f)$ équivalent à

$\vdash \neg W(t) \vee \neg POT(g,f)$ équivalent à

$\vdash f(t)$ □

Preuve du lemme 4

A partir de t, on choisit un ensemble d'états S qui joue le rôle de $[t]^*$ de la preuve précédente. On considère le tableau T au moment après la définition rg(INEV(g,f),..) et avant l'application de (M5) et on choisit S' de la manière suivante:

(1) $t \in S'$

(2) $\forall s \in S'$ $h = INEV(g,f)$ ou $h = P\tilde{R}EINEV(g,f) \in S'$ et rg(h,s) non défini.

(3) $\forall s \in S'$, si s est un α-noeud, alors $s_1 \in S'$.

(4) $\forall s \in S'$, si s est un β-noeud, alors $s_i \in S'$ si s_i non marqué pour i=1,2.

(5) $\forall s \in S'$, si s un X-noeud, alors $]s_1 \in S'$.

On définit S comme l'ensemble des états de S'. La preuve qu'un tel ensemble S d'états non marqués à cet étape de l'algorithme existe, utilise la proposition 4, et peut être faite comme en [BMP2].

Soit $s \in S$ et $s_i \in S'$ le noeud construit à partir de s pour la formule $PRE(g_s) \in F(s)$. Alors, on pose

$G'(s) = G(s) \cup \{g_s\}$, i.e. $G'(s) = F(s_i)$

$Z(s) = \wedge G'(s)$

$Z' = \bigvee_{s \in S} Z(s)$

$[s]$ l'ensemble des états successeurs directs de $s \in S$

Par un raisonnement analogue à celui de la preuve précédente, on

obtient

$\vdash Z' \supset \neg f$ ce qui implique par $(R2'^{,S})$

$\vdash SOME(\neg g, Z') \supset SOME(\neg g, \neg f)$ (1).

Egalement, comme dans la preuve précédente, on obtient

$\vdash \Delta G'(s) \wedge g \supset \bigvee_{s' \in [s]} \left(\bigwedge_{f_i \in G(s')} \tilde{PRE}(f_i) \wedge PRE(g_s) \right)$ ce qui implique par

(T5) et la définition de Z(s)

$\vdash g \wedge Z(s) \supset \bigvee_{s' \in [s]} PRE(Z(s'))$ (2).

Parce que (2) est vrai pour tout $s \in S$, on obtient

$\vdash g \wedge Z' \supset \bigvee_{s \in S} PRE(Z(s))$. Par (T3) et la définition de Z' on obtient

$\vdash g \wedge Z' \supset PRE(Z')$. Par (R'),(E4') et (MP) on obtient

$\vdash Z' \supset SOME(\neg g, Z')$ ce qui implique par (1)

$\vdash Z' \supset SOME(\neg g, \neg f)$. Du fait que $Z(t) \supset Z'$ on obtient

$\vdash Z(t) \supset SOME(\neg g, \neg f)$ (3) Du fait que $INEV(g,f) \in G'(t)$ on obtient

$\vdash Z(t) \supset \neg SOME(\neg g, \neg f)$ ce qui implique par (3)

$\vdash \neg Z(t)$ équivalent à

$\vdash f(t)$ □

III.5 Comparaison des logiques du temps arborescent et des logiques du temps linéaire

En II.3 nous avons comparé des logiques en comparant leurs calculs. Le critère habituel de comparaison de logiques sur des langages de formules comparables est l'inclusion de leurs ensembles de théorèmes. La comparaison que nous effectuons ici, est d'une nature différente. En effet, les théorèmes ne sont pas du tout intéressants pour l'expression des propriétés parce qu'ils ne permettent pas de distinguer des modèles. Ici, on veut comparer les "propriétés" exprimables par les différentes logiques. Ceci est possible pour des logiques sur des langages de formules non comparables à condition de définir leurs interprétations sur la même classe de modèles. Une propriété est en effet définie par l'ensemble des modèles qui la satisfont. Ainsi, deux formules sont "expressivement équivalentes" si elles sont satisfaites par les mêmes ensembles de modèles.

Les modèles des logiques du temps arborescent considérées sont des systèmes de transitions. Les modèles des logiques du temps linéaire sont des ensembles de séquences. Pour obtenir une classe commune de modèles, on prend pour les logiques du temps linéaire seulement les ensembles de séquences maximales d'un système de transitions. Ce choix est raisonnable parce que ce sont les ensembles de séquences décrivant des comportements réalisables par un programme.

5.1 La logique LTL

Le Langage de formules F_{LTL} est $F_{LM'}$ défini en II.2.

La classe de modèles M_{LTL} est $M_{LTA'}$ défini en III.2

La relation de satisfaction $\models_{LT}$ est un sous-ensemble de $EX(m)\times F_{LTL}$ pour tout $m\in M_{LTL}$, c'est-à-dire si $s\in EX(m)$ alors $m.s\models f$ exprime le fait que la séquence s satisfait la formule f en m.

On définit les notations suivantes:

Si $s=s_0s_1... \in EX(m)$, on note par

$s^n := s_n s_{n+1}...$ la sous-séquence de s commençant en s_n

$first(s) := s_0$, le premier élément de la séquence.

Ainsi, on définit la relation de satisfaction $\models$ par:

Si $m\in M_{LTL}$, $s\in EX(m)$, $f,f'\in F_{LTL}$ et $p\in P$, on a

(1) $m.s\models p$ ssi $p\in first(s)$,

(2) $m.s\models \neg f$ ssi $m.s\not\models f$,

(3) $m.s\models f\vee f'$ ssi $m.s\models f$ ou $m.s\models f'$,

(4) $m.s\models \Box f$ ssi $\forall n\in N\ m.s^n\models f$.

Par conséquent, on obtient pour le dual

(5) $m.s\models \Diamond f$ ssi $\exists n\in N\ m.s^n\models f$.

Une formule f est <u>satisfaisable</u> ssi il existe un modèle m, et il existe $s\in EX(m)$ tels que $m.s\models f$.

Une formule f est <u>vrai en m</u> ssi $\forall s\in EX(m)\ m.s\models f$.

Une formule f est <u>valide</u> ssi $\forall m\in M_{LTL}\ m\models f$.

De cette définition on déduit donc que pour une formule f sans occurrence d'opérateurs modaux

$m.s\models_{LTL} f$ ssi $m.first(s)\models_{LM} f$ et en conséquence

$m.s\models_{LTL}\Box f$ ssi $m.first(s)\models_{LM}\Box f$ et

$m.s\models_{LTL}\Diamond f$ ssi $m.first(s)\models_{LM}\Diamond f$.

Tandis que pour des formules f contenant des opérateurs modaux ceci n'est en général pas le cas.

Ainsi, on a défini une interprétation $I_{LTL}=(F_{LTL}, M_{LTL}, \models_{LTL})$.

<u>Le calcul Cal_{LTL}</u>

A chaque modèle $m=(W,R,P)\in M_{LTL}$ on peut associer un modèle $m'=(W',R',P')\in M_{S4.3.1}$, où R' est une relation d'ordre linéaire et totale, par

- $W'=EX(m)$
- $(s,s')\in R'$ ssi $\exists n\ s'=s^n$
- $P'(s)=P(first(s))$.

Inversement, à chaque modèle $m=(W,R,P)\in M_{S4.3.1}$ on peut associer un modèle $m^1 = (W^1,R^1,P^1)$, défini par

- $W^1=W$
- $P^1=P$
- $R^1 = \{(w,w')\in R \mid \nexists w''\in R\ w''\neq w'$ et $w''\neq w$ et $(w,w'')\in R$ et $(w'',w')\in R\}$
  Parce que R est une relation d'ordre linéaire et totale, pour tout $w\in W^1$ existe un et un seul $w'\in W^1$ tel que $(w,w')\in R^1$.

Pour $m\in M_{LTL}$, $f\in F_{LM}$, $s\in EX(m)$ on a apparemment
$m,s\models_{LTL} f$ ssi $m,s\models_{S4.3.1} f$.

Pour $m\in M_{S4.3.1}$, $f\in F_{LM}$, $w\in W$ on a
1) Il existe une séquence unique $s\in EX(w)$
2) $m,w\models_{S4.3.1} f$ ssi $m^1,s\models_{LTL} f$.

De plus, on a pour $m\in M_{S4.3.1}$ $(m^1)'=m$.
Pour $m\in M_{LTL}$ on a en général $(m')^1\neq m$, mais $EX((m')^1)=EX(m)$ et
$m,s\models_{LTL} f$ ssi $(m')^1,s\models_{LTL} f$ $\forall s\in EX(m)$.

Une formule f est donc satisfaisable en I_{LTL} ssi elle est satisfaisable en $I_{S4.3.1}$, d'où on obtient que f est valide en I_{LTL} ssi f est valide en $I_{S4.3.1}$.

Théorème 1

La logique $LTL=(Cal_{S4.3.1}, I_{LTL})$ est consistante et complète. □

5.2 Un critère de comparaison de deux logiques

Comme on a déjà remarqué, comparer la 'signification' des formules d'une logique revient à comparer les ensembles des modèles qui les satisfont. On définit alors le critère de comparaison suivant.

Définition 1

Soient I_1, I_2 deux interprétations (deux logiques L_1 et L_2) sur la même classe de modèles M. Soient F_1 et F_2 les langages de formules respectifs. On définit pour $f \in F_1$ et $g \in F_2$ que f est expressivement équivalent à g, noté $f \equiv g$, ssi

$$\forall m \in M \ (m \models_{I_1} f \text{ ssi } m \models_{I_2} f).$$

Remarque: Ce critère définit également une équivalence entre les formules d'une même logique ou interprétation.

Comme critère de comparaison de la puissance d'expression de deux logiques on obtient alors,

Définition 2

Soient I_1, I_2 deux interprétations (deux logiques L_1, L_2) sur la même classe de modèles M. Soient F_1 et F_2 les langages de formules respectifs. On dit que I_1 est plus expressive que I_2, noté $I_1 \geq I_2$, respectivement L_1 plus expressive que L_2, ssi

$$\forall g \in F_2 \ \exists f \in F_1 \ g \equiv f.$$

Remarque: Ceci est l'équivalence forte, définie en [La1]. Pour la comparaison de LTA et LTL on ne peut pas raisonnablement définir une relation d'équivalence plus fine entre formules qui respecte la capacité des formules d'exprimer des propriétés des états et des séquences respectivement, parce que les formules de LTA sont des assertions sur des états et les formules de LTL sont des assertions sur des séquences, et à ce niveau les modèles ne sont pas comparables. Par ailleurs, nous pensons que l'équivalence forte est le critère approprié parce que les propriétés que l'on veut exprimer dans les spécifications de programmes sont des propriétés globales.

Pour obtenir notre résultat d'incomparabilité, on utilise le lemme suivant.

Lemme 1

Sous les mêmes hypothèses que en définition 1, on a:

(I) $\exists g \in F_1\ \exists m,m' \in M$ ($m \models_{I_2} g$ et $m' \not\models_{I_1} g$) et

(II) $\forall f \in F_1$ ($m \models_{I_1} f$ implique $m' \models_{I_1} f$)

implique $L_1 \not\geq L_2$.

Preuve: Evidente □

5.3 La comparaison de LTL et de LTA

Théorème 1

LTL $\not\geq$ LTA.

Preuve: Nous considérons la formule g=POT(p) de LTA, où $p \in P$ et deux modèles m,m' tels que

(I) $m' \models_{LTA} POT(p)$ et $m \not\models_{LTA} POT(p)$ et

(II) $m' \models_{LTL} f$ implique $m \models_{LTL} f$ $\forall f \in F_{LTL}$.

Ceci prouve le théorème 2 par le lemme 1.

Il reste donc à trouver des modèles m,m' satisfaisant (I) et (II). Considérons les modèles m,m' sur P={p}, définis par:

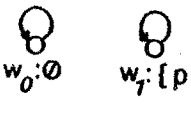
m = (W,R,P), où

$W=\{w_0,w_1\}$.

$R=\{(w_0,w_0),(w_1,w_1)\}$.

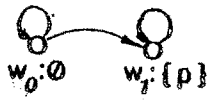
P est défini par $P(w_0)=\emptyset$ et $P(w_1)=\{p\}$.

représenté par



On obtient $EX(m)=\{w^{\infty}_0,w^{\infty}_1\}$.

$m'=(W',R',P')=(W,R\cup\{(w_0,w_1)\},P)$,
représenté par



On obtient $EX(m')=EX(m)\cup\{w_0^n w_1^\infty \mid n>0\}$.

Il est évident que

$m'\models POT(p)$ et $m\not\models POT(p)$ (I)

Du fait que $P=P'$ et $EX(m)\subseteq EX(m')$, on obtient

$\forall g\in F_{LTL}\ \forall s\in EX(m)$ $m',s\models g$ ssi $m,s\models g$,

d'où on obtient (II), ce qui complète la preuve. □

Remarque: La preuve du théorème 1 ne dépend pas de la définition de l'opérateur modal de LTL, mais découle de la propriété suivante:

Pour $m,m'\in M$, tels que $W=W'$ et $P=P'$ on a

$EX(m)\subseteq EX(m')$ implique $\forall g\in F_{LTL}$ ($m'\models g$ implique $m\models g$).

Cette propriété est vraie pour toute logique du temps linéaire, obtenue en enrichissant LTL avec d'autres opérateurs modaux. Ceci nous permet de formuler la proposition suivante.

Proposition 1

Pour toute logique du temps linéaire LTL', obtenue en enrichissant LTL avec d'autres opérateurs modaux, on a

LTL' ⊁ LTA □

Théorème 2

LTA ⊁ LTL.

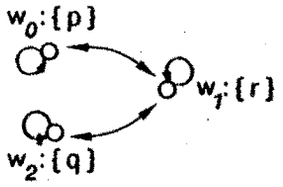
Preuve: Nous considérons la formule $g=\Box p\vee\Box q\vee\Diamond r$ de LTL, où $p,q,r\in P$

et deux modèles $m,m' \in M$ tels que

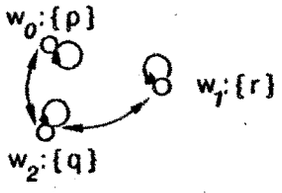
$m \models_{LTL} g$ et $m' \not\models_{LTL} g$ et
$m \models_{LTA} f$ implique $m' \models_{LTA} f$ $\forall f \in F_{LTA}$.

Ceci prouve le théorème 2 par le lemme 1.
Il reste donc à trouver des modèles m,m' satisfaisant (I) et (II).
Considérons les modèles m,m' sur $P=\{p,q,r\}$ définis par:

$m=(W,R,P)$, où W,R et P sont définis par la représentation suivante:



$m'=(W',R',P')=(W,R',P)$, où R' est défini par la représentation suivante:



Il est facile de voir que pour chaque séquence $s \in EX(m)$

$m,s \not\models \Box p$ et $m,s \not\models \Box q$ implique $m,s \models \Diamond r$,

d'où on obtient que $m \models \Box p \vee \Box q \vee \Diamond r$.
Tandis que pour la séquence $s=(w_0 w_2)^\infty \in EX(m')$ on a

$m',s \not\models \Box p \vee \Box q \vee \Diamond r$, d'où $m' \not\models \Box p \vee \Box q \vee \Diamond r$ (I).

Nous prouvons (II) par induction sur la structure des formules.

(1) $f \in P$ implique $\forall w \in W$ $(m,w \models f$ ssi $m',w \models f)$

(2) Soient $f,f' \in F_{LTA}$ tels que

(α) $\forall w \in W$ $(m,w \models f$ ssi $m',w \models f)$ et $(m,w \models f'$ ssi $m',w \models f')$

On obtient

a) $g=\neg f$ implique pour tout $w \in W$
$m,w \models g$ ssi $m,w \not\models f$

ssi $m'.w \not\models f$ par (α)

ssi $m'.w \models g$.

b) $g = f \vee f'$ implique pour tout $w \in W$

$m.w \models g$ ssi $m.w \models f$ ou $m.w \models f'$

ssi $m'.w \models f$ ou $m'.w \models f'$ par (α)

ssi $m'.w \models g$.

c) $g = ALL(f)$ implique pour tout $w \in W$

$m.w \models g$ ssi $\forall w' \in W$ $m.w' \models f$ parce que R fortement connexe.

ssi $\forall w' \in W$ $m'.w' \models f$ par (α)

ssi $m'.w \models g$ parce que R' fortement connexe.

d) $g = SOME(f)$ implique $\forall w \in W$

$m.w \models g$ ssi $m.w \models f$ parce que $\forall w \in W$ $w^{\infty} \in EX(m)$

ssi $m'.w \models f$ par (α)

ssi $m'.w \models g$ parce que $\forall w \in W$ $w^{\infty} \in EX(m')$.

Par (1) et (2) et la définition de F_{LTA} on obtient

$\forall f \in F_{LTA}$ $\forall w \in W$ $m.w \models f$ ssi $m'.w \models f$,

d'où on obtient (II), ce qui complète la preuve. □

Remarque: La preuve du théorème 2 dépend essentiellement de la définition des opérateurs modaux de LTA. En effet, cette preuve ne peut pas être étendue pour prouver p.ex. LTAC ⊃ LTL.

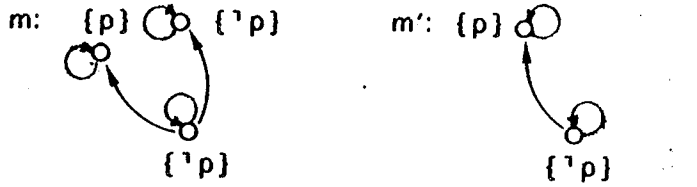
5.4 Conclusion

Même en restreignant la classe de modèles M_{LTL} aux ensembles de séquences générées par un système de transitions, nous avons prouvé l'incomparabilité des logiques LTL et LTA comme en [La1].
Pour renforcer ce résultat nous avons montré qu'il ne peut pas exister une logique du temps linéaire, extension de LTL, qui soit plus expressive

que LTA.

Cependant, nous n'avons pas trouvé une preuve formelle qu'il ne puisse pas exister une logique du temps arborescent, extension de LTA et plus expressive que LTL. En effet, en [EH2] une logique $ETCL^{+n}$ est proposée et il est démontré qu'elle est plus expressive que LTL. Mais cette logique ne peut pas être considérée comme une logique du temps arborescent, étant donné que pour certaines formules la relation de satisfaction est définie sur des séquences. En [EH2] il est également démontré, pour une définition d'une "équivalence expressive" plus fine, que LTAC n'est pas plus expressive que LTL. Cette preuve ne peut pas être utilisée pour prouver LTAC $\succ$ LTL, mais tous ces résultats renforcent l'hypothèse qu'il ne peut pas exister une logique modale du temps arborescent, extension de LTA, dont tous les opérateurs sont définissables comme des points fixes d'opérateurs construits à partir de PRE et $\widetilde{PRE}$, qui soit plus expressive que LTL.

La comparaison effectuée montre qu'il n'est pas possible d'exprimer en LTL et toute extension linéaire de cette logique des classes importantes de propriétés exprimables en termes de "POT", par exemple l'absence de blocage partiel. Considérons les deux modèles suivants:

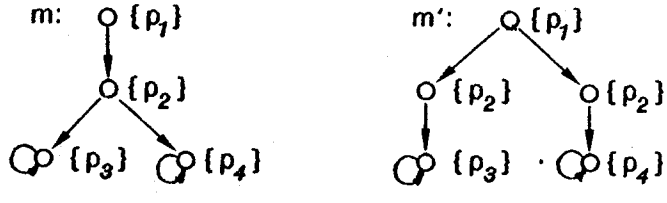


Si p est un prédicat exprimant le fait qu'une action t est exécutable à partir d'un état, on a:

$m \not\models ALLPOT(p)$ et $m' \models ALLPOT(p)$,

c'est-à-dire t est toujours exécutable dans m' mais pas en m. Pourtant ces deux modèles ne sont pas distinguables en LTL.

Cette impossibilité des logiques du temps linéaire d'exprimer POT implique l'impossibilité de distinguer des modèles qui ont les mêmes séquences mais pas la même structure. Par exemple les modèles



m et m' ne sont pas distinguables en LTL. Donc si LTL est utilisé comme outil de preuve, m et m' sont équivalents modulo la relation d'équivalence induite sur les modèles. L'utilisation des relations d'équivalence de ce type a souvent été critiquée (voir [MI]).
C'est pour ces raisons que nous avons décidé de prendre des logiques du temps arborescent en tant qu'outils de la spécification et de la preuve de programmes.

IV UTILISATION DES LOGIQUES DU TEMPS ARBORESCENT EN TANT QU'OUTIL DE PREUVE DES PROGRAMMES: APPLICATION POUR UN SOUS-ENSEMBLE DE CCS

IV.1 Introduction

Dans ce chapitre nous étudions le problème de l'adéquation des logiques du temps arborescent à la preuve des programmes non déterministes. Plus particulièrement, nous cherchons des logiques permettant de faire des preuves constructives des programmes décrits dans un modèle algébrique.

Soit L un langage de description de programmes et F l'ensemble des formules d'une logique, utilisée comme langage de spécification. Pour pouvoir définir une méthode constructive de preuve des propriétés des programmes dans L, il est nécessaire que

(1) Pour tout opérateur n-aire o du langage, il existe un opérateur Θ de la logique.

(2) Pour $\{t_i\}_{i=1}^n$ un ensemble de programmes dans L et $\{f_i\}_{i=1}^n$ un ensemble de formules de F, on a:

$t_i \models f_i$ pour $i=1...n$ implique $o(t_1,...t_n) \models \Theta(f_1,...f_n)$.

De plus, cette assertion est la plus forte que l'on puisse déduire des hypothèses.

Dans le cas général, le langage L peut être considéré comme une algèbre dont les opérateurs correspondent aux constructeurs de programmes, munie d'une relation de congruence ~. Une telle relation définit un concept d'équivalence opérationnelle qui est supposé être le plus adéquat et satisfaisant en pratique pour la comparaison des programmes.
L'utilisation des formules de F pour l'expression des propriétés induit une relation d'équivalence $\overset{F}{\equiv}$ sur L:
Pour deux programmes $t,t' \in L$

$t \overset{F}{\equiv} t'$ ssi $\forall f \in F$ $(t \models f$ ssi $t' \models f)$,

autrement dit, t et t' ne sont pas distinguables par des formules de F. Ces deux relations d'équivalence $\overset{F}{\equiv}$ et ~ ne doivent pas être choisies indépendamment. F étant le langage de spécification, on doit avoir $\sim \subseteq \overset{F}{\equiv}$, c'est-à-dire si deux programmes sont équivalents, alors ils ne peuvent pas être distingués par des spécifications. Si en plus, F est utilisé comme langage de preuve, alors il est nécessaire que $\overset{F}{\equiv} \subseteq \sim$, c'est-à-dire si deux programmes ne peuvent pas être distingués par des formules de F, alors ils sont opérationnellement équivalents. Par conséquent, l'utilisation de F comme langage de preuve implique $\overset{F}{\equiv} = \sim$ (3).

Dans ce chapitre nous cherchons une logique permettant de faire des preuves constructives en prenant comme langage de description des programmes un sous-ensemble de CCS de R.Milner [Mi] et comme relation, la congruence observationnelle $\overset{\mathcal{O}}{\sim}$. Cette logique doit satisfaire les exigences (1), (2) et (3).
Dans un premier temps, on considère le langage de formules F_A d'une logique du temps arborescent très générale pour laquelle les arbres étiquetés sur un vocabulaire A (processus de CCS) constituent une classe de modèles. On définit un sous-ensemble F' de F_A tel que les restrictions de $\overset{\mathcal{O}}{\sim}$ et $\overset{F'}{\equiv}$ sur les processus finis coïncident. F' est défini à l'aide d'une fonction $|\ |$ qui associe à chaque processus fini de CCS une formule $|t|$ de F' telle que l'on a pour tous les processus t,t':

$t' \models |t|$ ssi $t \overset{\mathcal{O}}{\sim} t'$ (4).

En prenant F' les formules générées par l'image de $\|\ \|$, on a $\mathcal{E}' = \mathcal{S}$. La fonction $\|\ \|$ est définie par induction sur la structure des processus finis de CCS en associant à Nil, + et l'ensemble des actions $\{a\}_{a\in A}$ respectivement une constante $\|Nil\|$ et des opérateurs sur les formules $\oplus$ et $\{@\}_{a\in A}$. Ainsi, on obtient une caractérisation modale des termes finis de CCS. Des caractérisations similaires sont présentées dans [HM], [BR] et [St]. Leurs résultats sont toutefois moins forts, étant donné qu'ils permettent de caractériser une classe de congruence ou d'équivalence comme une conjonction infinie de formules et cette caractérisation n'est pas constructive.

Ayant obtenu une caractérisation modale des termes finis de CCS, nous essayons de généraliser ce résultat pour les termes infinis (obtenus par définition récursive). Cette généralisation n'a été possible que pour la sous-classe des termes contrôlables, c'est-à-dire les termes pour lesquels il existe un terme congru sans occurrence d'actions non-observables. Pour définir les termes infinis de CCS, nous considérons une extension du calcul, obtenue par adjonction d'une constante Ω. Sur cette extension on peut définir une relation de préordre $\ll$ telle que pour chaque terme t, $t \ll \Omega$ et la restriction de la relation $\ll \cap \ll^{-1}$ sur CCS est la congruence observationnelle sur les termes contrôlables. Les termes infinis sont définis comme les limites de suites décroissantes pour la relation d'ordre induite par $\ll$. Ces résultats permettent de définir une logique pour la preuve constructive des processus contrôlables. Cette logique est le treillis distributif complémenté dont les éléments atomiques sont les formules de l'image de la fonction $\|\ \|$. Les formules de cette logique représentent des unions de classes de congruence. A un arbre (terme) t donné, on peut en effet associer une formule de la logique qui représente les classes de $\mathcal{S}$ de tous les arbres qui sont des "prolongements" de t.

IV.2 Caractérisation de la congruence observationnelle sur les termes finis de CCS

2.1 La logique du temps arborescent "relativisée" LTAR

Nous introduisons une logique du temps arborescent "relativisée" LTAR(A) avec un opérateur "plus petit point fixe" comme en [Ko], paramétrée par un ensemble A: A est un ensemble de constantes, dont les éléments ne jouent pas le rôle de variables propositionnelles, mais sont des noms des relations de transitions par rapport auxquelles la logique est "relativisée". En effet, on considère une logique sans variables propositionnelles.

Le langage de formules F_A est le sous-ensemble des formules fermées de F'_A, défini à partir de la signature Σ et d'un ensemble de variables X, où

$$\Sigma = \Sigma_{CP} \cup \{<>\} \cup \{\mu x. | x \in X\} \cup A,$$

où $<>$ est un opérateur unaire, $\mu x.$ est un opérateur unaire pour tout x et A est un ensemble de constantes.
On définit les abréviations suivantes pour les opérateurs duaux de $<>$ et $\mu x.$ Si $f \in F_A$, $g \in F'_A$ alors,

$$[f] := \neg <> \neg f$$
$$\sigma x.g(x) := \mu x.\neg g(\neg x).$$

La classe de modèles M_A contient les modèles de la forme $m=(W,w_0,\{\xrightarrow{a}\}_{a' \in A'})$ qui représentent des arbres étiquetés, c'est-à-dire

- W est un ensemble d'états, les noeuds de l'arbre.
- w_0 est l'état initial, la racine de l'arbre.
- $\{\xrightarrow{a}\}_{a' \in A'}$ est un ensemble de relations d'accessibilité directe $\xrightarrow{a} \subseteq W \times W$. Parce que m représente un arbre étiqueté alors,
  - $\nexists w' \in W \; \exists a' \in A' \; w' \xrightarrow{a} w_0$ et
  - $\forall w' \in W, w' \neq w_0$ implique w' a un prédécesseur unique via $\bigcup_{a' \in A'} \xrightarrow{a}$.

$m \in M_A$ peut être vu comme un modèle particulier $m=(W,R,P)$ de $M_{LM'}$ où $R = \bigcup_{a' \in A'} \{\xrightarrow{a'}\}$ et $P=\emptyset$.

La relation de satisfaction $\models_{LTAR(A)}$ est définie comme $\models_{LM}$ en I.1.2.3 par les règles (1) à (3).

Pour une formule $f \in F_A$ et un modèle $m \in M_A$ on définit:

- f est vrai en m, noté $m \models f$, ssi $m.w_0 \models f$, où w_0 est la racine de m.
- f est valide, noté $\models f$, ssi $m \models f$ $\forall m \in M_A$.

Maintenant, on donne les règles supplémentaires qui complètent la définition de $\models$. Si $m \in M_A$, $w \in W$, $f \in F_A$, $g \in F'_A$, $a \in A$, alors

(A) $m,w \models a$ ssi $\exists w' \in W$ $(w',w) \in \xrightarrow{a'}$, noté $w' \xrightarrow{a'} w$.

(<>) $m,w \models \langle f \rangle$ ssi $\exists w' \in W$ $\exists a' \in A'$ $w \xrightarrow{a'} w'$ et $m,w' \models f$.

(μx) $m,w \models \mu x.g(x)$ ssi $\forall f \in F_A$ ($\models g(f) \supset f$ implique $m,w \models f$).

Par conséquent, on obtient pour les duaux

([ ]) $m,w \models [f]$ ssi $\forall w' \in W$ $\forall a' \in A'$ $w \xrightarrow{a'} w'$ implique $m,w' \models f$.

(σx) $m,w \models \sigma x.g(x)$ ssi $\forall f \in F_A$ ($\models f \supset \neg g(\neg f)$ implique $m,w \models f$).

Remarque:

- $m,w \models a$ signifie que w n'est pas la racine, et w et son prédécesseur unique w' sont liés via la relation $\xrightarrow{a'}$, où à tout $a \in A$ correspond $a' \in A'$ et inversement. Parce que A et A' sont isomorphes, on ne les distingue plus par la suite.
- $m,w \models \langle f \rangle$ signifie qu'il existe un successeur direct w' de w via $\bigcup_{a \in A} \xrightarrow{a}$ qui satisfait f, c'est-à-dire $\langle f \rangle$ correspond à PRE(f) de UB pour $R = \bigcup_{a \in A} \xrightarrow{a}$.
- $m,w \models \mu x.g(x)$ signifie que w satisfait le plus petit point fixe de la fonctionnelle g(x), c'est-à-dire dans cette logique on peut définir les opérateurs modaux de UB et LTAC, par exemple

$$POT(f) := \mu x.(f \vee \langle x \rangle).$$

Ainsi, on a défini une interprétation $I_{LTAR(A)}=(F_A,M_A,\models)$.

Si $m=(W,w_0,\{\xrightarrow{a}\})$ est un arbre, tout état $w \in W$ définit un sous-arbre $m_w=(W^w,w,\{\xrightarrow{a}\}^w)$, où $W^w \subseteq W$ est l'ensemble des états accessibles à

partir de w via $(\bigcup_{a\in A} \xrightarrow{a})^*$, et $\{\xrightarrow{a}\}^w$ sont les restrictions des relations $\xrightarrow{a}$ sur W^w. Ainsi, on peut considérer les relations $\xrightarrow{a}$ comme des relations entre arbres et écrire $m_w \xrightarrow{a} m_{w'}$ à la place de $w \xrightarrow{a} w'$.
Par la suite nous considérons l'ensemble des "formules futures" de FF_A, défini par la

Définition 1

FF_A = {$f\in F_A$ | toute occurrence de $a\in A$ en f se trouve à l'intérieur d'un opérateur <> ou [ ]}.

Pour les formules f appartenant à FF_A on a apparemment que $m,w\models f$ ssi $m_w \models f$, c'est-à-dire la valeur de f en w ne dépend pas de la relation $\xrightarrow{a}$ par laquelle on arrive en w, mais seulement du sous-arbre m_w. Pour les formules de FF_A on considère donc seulement la relation de satisfaction $\models \subseteq M_A \times FF_A$.

Le calcul $Cal_{LTAR(A)}$

On considère le calcul suivant, similaire à celui donné en [Ko] qui contient les axiomes et règles du CP et les axiomes et règles suivants:

(1) $[f\supset f'] \supset ([f]\supset[f'])$

(2) $[f]\wedge\langle f'\rangle \supset \langle f\wedge f'\rangle$

(3) $\dfrac{\vdash f}{\vdash [f]}$

(4) $a\wedge b \equiv \bot$ pour $a,b\in A$ et $a\neq b$

(5) $[\bigvee_{a\in A} a]$

(6) $g(\mu x.g(x)) \supset \mu x.g(x)$

(7) $\dfrac{\vdash g(f)\supset f}{\vdash \mu x.g(x)\supset f}$

Remarque: (1) à (3) signifient que [ ] est un opérateur normal. On

n'a pas $[f]\supset\langle f\rangle$ parce que les relations $\bigcup_{a\in A}\xrightarrow{a}$ ne sont pas nécessairement totales. (4) exprime le fait que tout état a, au maximum, un prédécesseur. (5) exprime le fait que si w a un successeur w', alors via $\bigcup_{a\in A}\xrightarrow{a}$. (6) et (7) expriment le fait que $\mu x.$ est un opérateur "plus petit point fixe".

Kozen donne en [Ko] une logique similaire contenant un opérateur "plus petit point fixe" et prouve qu'elle est consistante et complète. De manière analogue, on peut prouver que la logique

$$LTAR(A)=(Cal_{LTAR(A)}, I_{LTAR(A)})$$

est consistante et complète.

Propriété 1

Les formules suivantes sont des théorèmes de LTAR(A).

(T1) $\langle f\vee f'\rangle \equiv \langle f\rangle\vee\langle f'\rangle$

(T2) $\langle f\wedge f'\rangle \supset \langle f\rangle\wedge\langle f'\rangle$

(T3) $[f\wedge f'] \equiv [f]\wedge[f']$

(T4) $[f]\vee[f'] \supset [f\vee f']$

(T5) $[f]\wedge\langle f'\rangle \equiv [f]\wedge\langle f\wedge f'\rangle$

(T6) $\langle f\rangle \supset \bigvee_{a\in A}\langle a\wedge f\rangle$

(T7) $\mu x.g(x) \equiv \bigvee_{n\in N} g_n$, où $g_0\equiv\bot$, $g_{n+1}=g(g_n)$ si g est continue.

(T8) $\sigma x.g(x) \equiv \bigwedge_{n\in N} g_n$, où $g_0\equiv\bot$, $g_{n+1}=g(g_n)$ si g est continue.

LTAR a les propriétés suivantes:

(P1) $m\models\langle a\wedge f\rangle$ ssi $\exists m'\ m\xrightarrow{a}m'$ et $m'\models f$.

(P2) $m\models[\bigvee_{i\in I} a_i\wedge f_i]$ ssi $\forall m'\ \forall a\ (m\xrightarrow{a}m'$ implique $\exists i\in I\ a=a_i$ et $m'\models f_i)$.

Preuve:

(T1) à (T5) correspondent à (T1) à (T5) de III.3. (T6) est une conséquence directe de (5). (T7) et (T8) sont des conséquences du théorème de Knaster-Tarski. (P1) et (P2) découlent directement de la définition de $\models$ et de la remarque sur les formules de FF_A. □

2.2 Le langage de description de programmes $T[\Sigma]$

Nous considérons un langage de description de programmes $T[\Sigma]$ isomorphe à l'ensemble des arbres étiquetés M_A. Par conséquent, on peut considérer $T[\Sigma]$ comme une classe de modèles pour LTAR(A).

Définition 2

1) Le langage de description de programmes $T[\Sigma]$ est l'algèbre libre dont la signature est $\Sigma=\{Nil,+\}\cup A$, où Nil est une constante, A un ensemble d'opérateurs unaires et + un opérateur binaire.

2) Pour tout $a\in A$ on définit une relation $\xrightarrow{a}\subseteq T[\Sigma]\times T[\Sigma]$, qui est la plus petite relation satisfaisant

$$at \xrightarrow{a} t \quad \forall t\in T[\Sigma]$$
$$t_1 \xrightarrow{a} t' \text{ implique } t_1+t_2 \xrightarrow{a} t' \quad \forall t_1,t_2,t'\in T[\Sigma]$$
$$t_1 \xrightarrow{a} t' \text{ implique } t_2+t_1 \xrightarrow{a} t' \quad \forall t_1,t_2,t'\in T[\Sigma]$$

Ainsi, on peut associer à un terme t un arbre étiqueté $m_t = (W_t,t,\{\xrightarrow{a}\})\in M_A$, où W_t est l'ensemble des sous-termes atteignables via $(\bigcup_{a\in A}\xrightarrow{a})^*$, où les $\xrightarrow{a}$ sont définis par la définition 2 et $\{\xrightarrow{a}\}$ est l'ensemble des restrictions de ces relations sur W_t. Par la suite, on identifie les arbres étiquetés avec les termes qui les représentent. Ainsi, on écrit pour une formule $f\in FF_A$, $t\in T[\Sigma]$ $t\models f$ à la place de $m_t\models f$.

Propriétés 2

(P1) $Nil\models[\bot]$

(P2) $t\models f$ implique $at\models\langle a\wedge f\rangle\wedge[a\wedge f]$

(P3) $t\models\langle a\wedge f\rangle$ implique $t+t'\models\langle a\wedge f\rangle$

(P4) $t+t'\models f$ implique $t'+t\models f$

(P5) $t\models[f]$ et $t'\models[f']$ implique $t+t'\models[f\vee f']$

(P6) $t+Nil\models f$ ssi $t\models f$

(P7) $t+t'\models[f]$ ssi $t\models[f]$ et $t'\models[f]$

Preuve: Evidente □

Remarque: Par la suite l'opérateur $\wedge$ est souvent omis.

Définition 3 (équivalence forte)

L'équivalence forte $\sim \subseteq T[\Sigma]\times T[\Sigma]$ est la plus grande relation telle que pour tout $t_1,t_2 \in T[\Sigma]$:

$t_1 \sim t_2$ ssi $\forall a \in A$ $(t_1 \xrightarrow{a} t_1'$ implique $\exists t_2' (t_2 \xrightarrow{a} t_2'$ et $t_1' \sim t_2'))$ et
$\forall a \in A$ $(t_2 \xrightarrow{a} t_2'$ implique $\exists t_1' (t_1 \xrightarrow{a} t_1'$ et $t_1' \sim t_2'))$.

En [MI] a été démontré que $\sim$ est une congruence qui peut être caractérisée par les axiomes suivants:

(A1) $(t_1+t_2)+t_3 = t_1+(t_2+t_3)$
(A2) $t_1+t_2 = t_2+t_1$
(A3) $t+t = t$
(A4) $t+Nil = t$.

2.3 Caractérisation modale de l'équivalence forte $\sim$

Pour familiariser le lecteur avec notre approche, nous présentons d'abord une caractérisation modale de l'équivalence forte, c'est-à-dire nous définissons une fonction $|\ |\in T[\Sigma] \to FF_A$ satisfaisant la propriété suivante:

(C) $t' \models |t|$ ssi $t' \sim t$ (caractérisation)

qui est équivalente aux trois propriétés suivantes:

(C1) $t \models |t|$
(C2) $t' \models |t|$ implique $t' \sim t$
(C3) $t' \sim t$ implique $|t| = |t'|$.

Pour la fonction $|\ |$, définie par la suite, on montre (C1) à (C3), c'est-à-dire (C).

Ensuite, on suit la même démarche pour la congruence observationnelle $\mathcal{S}$ (Définition 5).

Définition 4

La fonction $|\ |\in T[\Sigma]\to FF_A$ est définie récursivement par

- $|Nil| = [\bot]$
- $|at| = \langle a\rangle|t|\rangle\wedge[a\rangle|t|]$
- $$|t+t'| = \begin{cases} |t| \text{ si } |t'|\equiv[\bot] \\ |t'| \text{ si } |t|\equiv[\bot] \\ \bigwedge_{i\in I}\langle a_i\rangle|t_i|\rangle\bigwedge_{j\in J}\langle b_j\rangle|t'_j|\rangle\wedge[\bigvee_{i\in I}a_i\rangle|t_i| \vee \bigvee_{j\in J}b_j\rangle|t'_j|] \\ \text{si } |t| = \bigwedge_{i\in I}\langle a_i\rangle|t_i|\rangle\wedge[\bigvee_{i\in I}a_i\rangle|t_i|] \text{ et} \\ \quad |t'| = \bigwedge_{j\in J}\langle b_j\rangle|t'_j|\rangle\wedge[\bigvee_{j\in J}b_j\rangle|t'_j|] \end{cases}$$

Il est facile de voir que $|\ |$ est une fonction qui associe à chaque terme t une formule de la forme

$$|t| = \begin{cases} [\bot] \text{ si } t\sim Nil \\ \bigwedge_{i\in I}\langle a_i\rangle|t_i|\rangle\wedge[\bigvee_{i\in I}a_i\rangle|t_i|]. \end{cases}$$

Exemple

On calcule $|t|$ pour $t = aNil+c(aNil+bNil)$:

$|aNil| = \langle a\rangle[\bot]\rangle\wedge[a\rangle[\bot]]$,

$|bNil| = \langle b\rangle[\bot]\rangle\wedge[b\rangle[\bot]]$,

$|aNil+bNil| = \langle a\rangle[\bot]\rangle\langle b\rangle[\bot]\rangle\wedge[a\rangle[\bot]\vee b\rangle[\bot]]$,

$|c(aNil+bNil)| = \langle c\rangle|aNil+bNil|\rangle\wedge[c\rangle|aNil+bNil|]$,

$|aNil+c(aNil+bNil)| = \langle a\rangle[\bot]\rangle\langle c\rangle|aNil+bNil|\rangle\wedge[a\rangle[\bot]\vee c\rangle|aNil+bNil|]$.

Le théorème suivant montre que pour tout terme $t\in T[\Sigma]$ la formule $|t|$ caractérise la classe de congruence forte de t.

Théorème 2

La fonction $|\ |$ caractérise la congruence forte, c'est-à-dire pour $\forall t,t' \in T[\Sigma]$

(C) $t' \models |t|$ ssi $t' \sim t$.

Preuve: On montre (C1) à (C3), définis précédemment:

(C1) $\forall t \in T[\Sigma]$ $t \models |t|$ est démontré par induction sur la structure de $T[\Sigma]$.

- $Nil \models [\bot]$ par (P1) de la propriété 2.
- $t \models |t|$ implique $at \models \langle a \rangle |t| > [a \rangle |t|]$ par (P2)
  implique $at \models |at|$ par la définition 4.
- $t \models |t|$ implique $t+Nil \models |t|$ par (P6)
  implique $t+Nil \models |t+Nil|$ par la définition 4.
- $t \models |t|$, $t' \models |t'|$, où $|t| = \bigwedge_{i \in I} \langle a_i \rangle |t_i| > [\bigvee_{i \in I} a_i \rangle |t_i|]$ et
  $|t'| = \bigwedge_{i \in J} \langle b_i \rangle |t_i'| > [\bigvee_{i \in J} b_i \rangle |t_i'|]$
  implique $t+t' \models \bigwedge_{i \in I} \langle a_i \rangle |t_i| \bigwedge_{i \in J} \langle b_i \rangle |t_i'| > [\bigvee_{i \in I} a_i \rangle |t_i| \vee \bigvee_{i \in J} b_i \rangle |t_i'|]$
  par (P3) et (P5)
  implique $t+t' \models |t+t'|$ par la définition 4.

(C2) $t' \models |t|$ implique $t' \sim t$ est démontré par induction sur la structure des formules $|t|$.

- $t \models [\bot]$ implique $\nexists t' \in T[\Sigma]$ $\exists a \in A$ $t \xrightarrow{a} t'$.
  implique $t \sim Nil$.

- Soit $|t|$ une formule telle que $\forall t' \in T[\Sigma]$ $t' \models |t|$ implique $t' \sim t$.
  Ainsi, on obtient pour tout terme $t' \in T[\Sigma]$ et $a \in A$
  $t' \models |at|$ implique $t' \models \langle a \rangle |t| > [a \rangle |t|]$ par la définition 4.
  implique $\exists t''(t' \xrightarrow{a} t''$ et $t'' \models |t|)$ et $\forall t''(t' \xrightarrow{b} t''$ implique $b=a$ et $t'' \models |t|)$,
  implique $\exists t''(t' \xrightarrow{a} t''$ et $t'' \sim t)$ et $\forall t''(t' \xrightarrow{b} t''$ implique $b=a$ et $t'' \sim t)$.
  implique $t' \sim at$ par la définition 3.

- $t \models |t_1+t_2|$ implique $t \sim t_1+t_2$ pour $|t_i|$ tels que $t' \models |t_i|$ implique $t' \sim t_i$ est obtenu par une preuve analogue.

(C3) $t' \sim t$ implique $|t'| \equiv |t|$.

On voit facilement que $|\ |$ préserve les axiomes (A1) à (A4), c'est-à-dire pour toute occurrence d'un axiome de la forme $t=t'$ on a $|t| \equiv |t'|$. Du fait que (A1) à (A4) constituent une axiomatisation complète de l'équivalence forte $\sim$, on obtient (C3). □

2.4 Caractérisation modale de la congruence observationnelle

Dans ce paragraphe des résultats concernant la caractérisation de la congruence observationnelle $\overset{\mathcal{C}}{\approx}$ de CCS [Mi] sont présentés. Dans ce cas on suppose que le vocabulaire d'étiquettes A contient une étiquette speciale τ. τ représente une action non-observable ou non-contrôlable. Comme dans la section précédente on définit une fonction $|\ | \in T[\Sigma] \to FF_A$ qui associe à chaque terme t une formule $|t|$ qui est satisfaite exactement par les termes t' observationnellement congrus à t.

D'abord, nous définissons la congruence observationnelle [Mi] et rappelons quelques propriétés importantes données en [Mi], [HM].

Définition 5

1) Soit $s=s_0 \ldots s_n$ une séquence de A^*. On écrit
$t \overset{s}{\Rightarrow} t'$ ssi $\exists t_1, \ldots, t_n \in T[\Sigma]$ $t \overset{s_0}{\Rightarrow} t_1 \ldots \overset{s_{n-1}}{\Rightarrow} t_n \overset{s_n}{\Rightarrow} t'$.

2) Soit $s=s_0 \ldots s_n$ une séquence de $(A-\{\tau\})^*$. On écrit

$$t \overset{s}{\Rightarrow} t' \text{ ssi } \begin{cases} t \xrightarrow{\tau^* s_0 \tau^* \ldots s_n \tau^*} t' \text{ si } s = s_0 \ldots s_n \\ t \xrightarrow{\tau^*} t' \text{ si } s = \epsilon, \text{ le mot vide de } A^*. \end{cases}$$

3) L'équivalence observationnelle est définie par
$\sim = \bigcap_{l \in N} \sim_l \subseteq T[\Sigma] \times T[\Sigma]$, où

- $t_1 \sim_0 t_2 \quad \forall t_1, t_2 \in T[\Sigma]$
- $t_1 \sim_{n+1} t_2$ ssi $\forall s \in (A - \{\tau\})^*$ [$t_1 \overset{s}{\Rightarrow} t_1'$ implique $\exists t_2' (t_2 \overset{s}{\Rightarrow} t_2'$ et $t_1' \sim_n t_2')$ et $t_2 \overset{s}{\Rightarrow} t_2'$ implique $\exists t_1' (t_1 \overset{s}{\Rightarrow} t_1'$ et $t_1' \sim_n t_2')$].

En effet $\sim$ est une relation d'équivalence [MI]. Par $\mathcal{S}$ est dénotée la plus grande relation de congruence sur $T[\Sigma]$ telle que $\mathcal{S} \subseteq \sim$.
La congruence observationnelle $\mathcal{S}$ peut être caractérisée par les axiomes suivants [HM].

(A1)-(A4) comme en 2.3
(A5) $a\tau t = at$
(A6) $\tau t + t = \tau t$
(A7) $a(\tau t_1 + t_2) + at_1 = a(\tau t_1 + t_2)$

Propriétés 3 [HM]

1) $\tau(t_1 + t_2) + t_1 = \tau(t_1 + t_2)$ (A8)
2) $t \mathcal{S} t'$ ssi $t \sim t'$ ou $\tau t \sim t'$ ou $t \sim \tau t'$.

Les deux définitions suivantes donnent des notations, utilisées pour la définition de la fonction | | de traduction d'un terme en sa formule caractéristique.

Définition 6

Pour les formules de la forme $f = \bigwedge_{i \in I} \langle a_i \rangle t_i \wedge [\bigvee_{i \in K} a_i \rangle t_i]$ où $t_i \in FF_A$ et $\not\vdash f \equiv \bot$, on définit:

$$\hat{f} := \bigvee_{i \in K} a_i \rangle t_i.$$

On définit également $[\hat{\bot}] := \bot$ et $\hat{\top} := \top$.

Proposition 1

$\hat{}$ est une fonction partielle de FF_A en F_A.

Preuve

Supposons $f' \equiv f$ et

(1) $f = \bigwedge_{i \in I} \langle a_i \wedge f_i \rangle \wedge [\bigvee_{i \in K} a_i \wedge f_i]$ et (2) $f' = \bigwedge_{i \in I'} \langle b_i \wedge f_i' \rangle \wedge [\bigvee_{i \in K'} b_i \wedge f_i']$

On doit prouver: $\forall m \in M_A . \forall w \in W$ $m,w \models \hat{f}$ ssi $m,w \models \hat{f}'$ (3).

Pour prouver (3) on prouve d'abord que

(1) et (2) implique $[\hat{f}] \equiv [\hat{f}']$ (4).

$\not\vdash f \equiv \bot$ implique $\exists t \in T[\Sigma]$ $t \models f$ et $\exists t' \in T[\Sigma]$ $t' \models [\hat{f}']$. Du fait que $f' \equiv f$ on obtient par les propriétés (P3) et (P5) $t+t' \models f'$ et par conséquent $t+t' \models f$. Par (P7) on obtient $t' \models [\hat{f}]$ et par symétrie (4).

Ensuite, on montre que (4) implique (3). Considérons $m \in M_A$, $w \in W$ tels que $m,w \models \bigvee_{i \in K} a_i \wedge f_i$. On peut faire la déduction suivante:

$\exists w' \in W$ $\exists i \in K$ $w' \xrightarrow{a_i} w$ et $m,w \models f_i$,	
$\exists w' \in W$ $\exists i \in K$ $m_w \xrightarrow{a_i} m_w$ et $m_w \models f_i$	parce que $f_i \in FF_A$.
$a_i m_w \models [a_i \wedge f_i]$	par (P2).
$a_i m_w \models [\bigvee_{i \in K'} b_i \wedge f_i']$	du fait que $[a_i \wedge f_i] \supset [\hat{f}]$ et (4).
$\exists i \in K'$ $a_i = b_i$ et $m_w \models f_i'$	parce que $f_i' \in FF_A$.
$m,w \models b_i \wedge f_i'$	parce que $w' \xrightarrow{b_i} w$ en m.
$m,w \models \hat{f}'$	parce que $b_i \wedge f_i' \supset \hat{f}'$.

Par symétrie, on obtient (4). □

Corollaire 1

Soient $f,f' \in FF_A$ comme dans la preuve de la proposition 1. $f \equiv f'$ et $\not\vdash f' \equiv \bot$ implique

$[\hat{f}] \equiv [\hat{f}']$ et

$\bigwedge_{i \in I} \langle a_i \wedge f_i \wedge \hat{f} \rangle \equiv \bigwedge_{i \in I'} \langle b_i \wedge f_i' \wedge \hat{f}' \rangle$.

Remarque: Soit $f \in FF_A$ telle que $\hat{f}$ est défini et $t \models f$. Alors $t' \models [\hat{f}]$ ssi $t+t' \models f$, c'est-à-dire $[\hat{f}]$ caractérise l'ensemble des termes dont l'addition à t préserve la satisfaction de f.

Définition 7

Soit $f \in FF_A$ une formule telle que $\hat{f}$ est défini. On définit E(f) par

$$E(f) := \mu x.(f \vee \langle \tau \wedge x \rangle [\tau \wedge x \vee \hat{f}]).$$

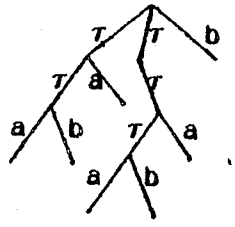
Proposition 2

E est une fonction partielle de FF_A en FF_A et

$$E(f) = \bigvee_{k \in N} X_k, \text{ où } X_0 := f$$
$$X_{k+1} := X_k \vee G(X_k, f), \text{ où}$$
$$G(x,f) := \langle \tau \wedge x \rangle [\tau \wedge x \vee \hat{f}]$$

Preuve: Etant donné que les arbres représentant des termes de FF_A sont de degré fini, les fonctionnelles $\lambda.G(x,f)$ sont continues. Ainsi, le résultat est obtenu par (T7) des propriétés 1.

Remarque: La signification de E(f) devient claire, quand nous avons prouvé que, si f caractérise la classe de congruence observationnelle d'un terme t, alors E(f) caractérise l'union des classes de congruence de t et τt. Par exemple, si t=aNil+bNil et $t \not\models f$, alors l'arbre ci-dessous représente un terme $t' \overset{\mathcal{S}}{\sim} \tau t$ et satisfait E(f).



A l'aide de ces définitions préliminaires, on peut définir la fonction de traduction de termes en formules.

Définition 8

La fonction $\| \ \| \in T[\Sigma] \to FF_A$ est définie récursivement par les quatre règles suivantes. Pour faciliter l'expression des règles, on définit en même temps un ensemble STRICT de formules de l'image de $\| \ \|$.

STRICT contient les formules représentant des termes t, qui ne sont pas congrus à un terme de la forme $\tau t'$. On dira dans ce cas que t est stricte.

Règle 1: - $|Nil| = [\bot]$
- $[\bot] \in STRICT$

Remarque: $Nil \models [\bot]$ et $\nexists t' \in T[\Sigma]$ $\tau t' \mathcal{S} Nil$.

Règle 2:
- $$|\tau t| = \begin{cases} \tau^\circ |t| \text{ si } |t| \in STRICT \\ |t| \text{ sinon} \end{cases}$$
où $\tau^\circ |t| = \langle \tau \wedge E|t| \rangle [\tau \wedge E|t| \vee |\hat{t}|]$.
- $|\tau t| \notin STRICT$.

Remarque: Le lecteur est invité de comparer cette règle avec la règle correspondante dans la définition 4, qui est $|\tau t| = \langle \tau \wedge |t| \rangle [\tau \wedge |t|]$. En règle 2 $|t|$ a été remplacé par $E|t|$ pour tenir compte de l'axiome (A5) et $|\hat{t}|$ a été ajouté (également en $E|t|$) pour préserver la satisfaction pour des termes congrus à t par (A8) de propriété 3 (en prenant $t=t_1+t_2$).
τt n'est pas stricte parce que $\tau t \mathcal{S} \tau\tau t$.

Règle 3: Pour $a \in A-\{\tau\}$
- $$|at| = \begin{cases} a^\circ |t'| \text{ s'il existe un } |t'| \text{ tel que } |t| \equiv \tau^\circ |t'| \\ a^\circ |t| \text{ sinon} \end{cases}$$
où $a^\circ |t| = \langle a \wedge E|t| \rangle [a \wedge E|t| \vee |\hat{t}| \backslash a]$.

$$\text{où } |\hat{t}| \backslash a = \begin{cases} \bigvee_{i \in I'} a \wedge f_i \text{ si } |\hat{t}| = \bigvee_{i \in I} a_i \wedge f_i \text{ et} \\ \qquad\qquad I' = \{i \in I \mid a_i = \tau\} \neq \emptyset \\ \bot \text{ si } |\hat{t}| = \bot \text{ ou } I' = \emptyset \end{cases}$$
- $|at| \in STRICT$

Remarque: Il est intéressant de comparer cette règle avec la règle correspondante en définition 4, qui est $|at| = \langle a \wedge |t| \rangle [a \wedge |t|]$. Dans la règle 3 $|t|$ est remplacé par $E|t|$ pour tenir compte de (A5). La

formule $|\hat{t}|\backslash a$ a été ajoutée pour tenir compte de (A7). En effet, si $t=t_1+\tau t_2$, alors $at \mathcal{S} at+at_2$. $[|\hat{t}|\backslash a]$ caractérise tous les termes at_2 tels que $at \mathcal{S} at+at_2$ par application de (A7). Dans le cas où $t=\tau t'$ on a $at \mathcal{S} at'$ c'est-à-dire a° peut être appliqué à $|t'|$ à la place de $|t|$. Finallement, at est stricte parce que $\exists t' \in T[\Sigma]$ tel que $at \mathcal{S} \tau t'$.

Règle 4:

$$- \; |t_1+t_2| = \begin{cases} |t_1| \text{ si } |t_2| \equiv [\bot] \\ |t_2| \text{ si } |t_1| \equiv [\bot] \\ |t_1| \oplus |t_2| \text{ sinon} \end{cases}$$

- $|t_1+t_2| \in$ STRICT ssi $\exists |t'| \; |t_1+t_2| \equiv \tau^\circ |t'|$.

Pour $|t_1| = \bigwedge_{i \in I1} \langle a_i \wedge E | t_i | \rangle [|\hat{t}_i|]$, $|t_2| = \bigwedge_{i \in I2} \langle b_i \wedge E | t_i' | \rangle [|\hat{t}_2|]$

$$|t_1| \oplus |t_2| = \bigwedge_{i \in I1'} \langle a_i \wedge E | t_i | \rangle \bigwedge_{i \in I2'} \langle b_i \wedge E | t_i' | \rangle \; [|\hat{t}_1| \vee |\hat{t}_2|]$$

Les ensembles d'indices $I1'$ et $I2'$ sont définis par:

$I1' = \{i \in I1 \mid \exists j \in I2 \; c(a_i t_i, b_j t_j')\}$.

$I2' = \{i \in I2 \mid \exists j \in I1 \; (b_i t_i', a_j t_j)\}$. où c est le prédicat

c(at,bt') ssi $([\widehat{a^\circ |t|}] \supset [\widehat{b^\circ |t'|}]$ et non $[\widehat{a^\circ |t|}] \equiv [\widehat{b^\circ |t'|}])$.

Remarque: On va démontrer que $|t| = \bigwedge_{i \in I} \langle a_i \wedge E | t_i | \rangle [\bigvee_{i \in I} \widehat{a_i^\circ |t_i|}]$ est la forme la plus générale des formules $|t|$ pour $t = \sum_{i \in I} a_i t_i$. Une comparaison de cette règle avec la règle correspondante en définition 4 montre que le même principe a été appliqué, avec la différence qu'un facteur $\langle a \wedge E | t | \rangle$ peut être "éliminé" pour tenir compte des axiomes (A6) et (A7). Le prédicat c(at,bt') est défini de telle sorte que $\langle a \wedge E | t | \rangle$ est éliminé ssi il existe un facteur $\langle b \wedge E | t' | \rangle$ tel que $at+bt' \mathcal{S} bt'$, mais non $at \mathcal{S} bt'$ pour éviter que deux "branches" congrues s'éliminent mutuellement. Cette règle est due au fait qu'un arbre congru à at+bt' n'a pas obligatoirement une branche congrue à at, mais peut en avoir une. Du dernier fait a déjà été tenu compte lors de la construction de $|bt|$: c'est pour cette raison que l'on a $[|\widehat{at}|] \supset [|\widehat{bt'}|]$. Du premier fait on tient compte en éliminant le facteur $\langle a \wedge E | t | \rangle$ dans la formule pour la somme at+bt', c'est-à-dire la nécessité d'avoir une branche congrue à at.

Finallement, une somme t_1+t_2 n'est pas stricte ssi il existe un t' tel que $t_1+t_2 \mathcal{S} \tau t'$. En effet, ceci signifie t_1 (ou t_2) est congru à $\tau t'$ et t_2 (respectivement t_1) est absorbé.

Exemple 2

On calcule $|t'|$ pour le terme t', obtenu en remplaçant c par τ dans le terme t de l'exemple 1, alors $t'=aNil+\tau(aNil+bNil)$.

$|aNil| = \langle a\wedge E[\bot]\rangle[a\wedge E[\bot]]$

$|bNil| = \langle b\wedge E[\bot]\rangle[b\wedge E[\bot]]$.

$|aNil+bNil| = \langle a\wedge E[\bot]\rangle\langle b\wedge E[\bot]\rangle[a\wedge E[\bot] v b\wedge E[\bot]]$

$|\tau(aNil+bNil)| = \langle \tau\wedge E|aNil+bNil|\rangle[\tau E|aNil+bNil| v\ a\wedge E[\bot] v b\wedge E[\bot]]$

$|aNil+\tau(aNil+bNil)| = |\tau(aNil+bNil)|$.

Dans le résultat final le facteur $\langle a\wedge E[\bot]\rangle$ est éliminé parce que $c(aNil,\tau(aNil+bNil))$ est satisfait, c'est-à-dire $[a\wedge E[\bot]]\supset[|\tau(aNil+bNil)|]$, mais pas inversement. En effet, on a $\tau(aNil+bNil)+aNil \mathcal{S} \tau(aNil+bNil)$ par (A8).

Proposition 3

$||$ est une fonction de $T[\Sigma]$ en FF_A.

Preuve: Il est facile de prouver par induction structurelle que la forme générale des formules $|t|$ est

$$|t| = \bigwedge_{i\in I} \langle a_i\wedge E|t_i\rangle[\bigvee_{i\in I} \widehat{a_i^\circ|t_i|}] \text{ ou } |t|=[\bot].$$

Ceci implique que $||$ est totale. Pour prouver que $||$ est une fonction, il suffit de démontrer que $|t|\equiv\tau^\circ|t'|\equiv\tau^\circ|t''|$ implique $a^\circ|t'|\equiv a^\circ|t''|$. ceci est le seul cas où l'unicité de l'image de $||$ n'est pas tout à fait évident:

- Soit $\tau^\circ|t'|\equiv\tau^\circ|t''|$, i.e.
  $\langle\tau\wedge E|t'|\rangle[\tau\wedge E|t'|v|\hat{t'}|] \equiv \langle\tau\wedge E|t''|\rangle[\tau\wedge E|t''|v|\hat{t''}|]$.
  Parce que $E|t'|, E|t''| \in FF_A$ on obtient par le corollaire 1 que $\langle\tau\wedge|t'|\rangle\equiv\langle\tau\wedge|t''|\rangle$ ce qui implique $\langle a\wedge|t'|\rangle\equiv\langle a\wedge|t''|\rangle$ (1) et

$[\tau\wedge E|t'|v|\hat{\uparrow}|]=[\tau\wedge E|t''|v|\hat{\uparrow}|]$ ce qui implique $[a\wedge E|t'|v|\hat{\uparrow}|\backslash a]=[a\wedge E|t''|v|\hat{\uparrow}|\backslash a]$. Ainsi, on obtient par (1) $a^{\circ}|t'|=a^{\circ}|t''|$. □

Lemme 1

$|t|\in STRICT$ ssi $\exists|t'|$ tel que $|t|=\tau^{\circ}|t'|$.

Preuve: On a que $\tau^{\circ}|t|\in STRICT$ d'après la règle 2, et par les règles 1 et 3 on ne peut pas obtenir une formule $|t|=\tau^{\circ}|t'|$ pour un $|t'|$. □

Le reste du paragraphe est consacré à la preuve que $||$ caractérise la congruence observationnelle $\mathcal{S}$ sur $T[\Sigma]$.

Théorème 3

La fonction $||$ caractérise la congruence observationnelle, c'est-à-dire

(C) $\forall t,t'\in T[\Sigma]$ $t'\models|t|$ ssi $t'\mathcal{S}t$.

Preuve: Dans les trois théorèmes suivants on montre

(C1)° $\forall t\in T[\Sigma]$ $t\models|t|$.

(C2) $\forall t,t'\in T[\Sigma]$ $t'\models|t|$ implique $t'\mathcal{S}t$.

(C3) $\forall t,t'\in T[\Sigma]$ $t'\mathcal{S}t$ implique $|t'|=|t|$.

Ceci suffit pour la preuve du théorème 3. □

Théorème 4

(C1) $\forall t\in T[\Sigma]$ $t\models|t|$.

Preuve: Par induction sur la structure de $T[\Sigma]$.

1. $Nil\models[\bot]$

2. Soit $|t|$ une formule telle que $t \models |t|$.

   2.1 Si $|t| \in$ STRICT alors $|\tau t| = \tau^\circ |t|$. On obtient

   $\tau t \models \langle \tau \wedge |t| \rangle [\tau \wedge |t|]$ par (P2),

   $\tau t \models \langle \tau \wedge E|t| \rangle [\tau \wedge E|t| \vee |\hat{\uparrow} t|]$ parce que $|t| \supset E|t|$,

   $\tau t \models \tau^\circ |t|$.

   2.2 Si $|t| \notin$ STRICT alors $|t| \equiv \tau^\circ |t'|$ pour un $|t'|$ et $t \models \tau^\circ |t'|$. Des définitions 7 et 8 on déduit $\tau^\circ |t'| = G(E|t'|, |t'|)$ et $E|t'| \equiv |t'| \vee G(E|t'|, |t'|)$. Ceci implique

   $t \models E|t'|$ parce que $\tau^\circ |t'| \supset E|t'|$,

   $\exists k\ t \models X_k$ où X_k comme en proposition 2.

   $\tau t \models \langle \tau \wedge X_k \rangle [\tau \wedge X_k]$ par (P2),

   $\tau t \models \tau^\circ |t'|$ parce que $X_k \supset E|t'|$ et $\tau^\circ |t| \equiv G(E|t'|, |t'|)$.

3. Soit $|t|$ une formule telle que $t \models |t|$ et $a \in A - \{\tau\}$.

   3.1 Si $|t| \in$ STRICT alors $|at| = a^\circ |t|$ et comme en 2.1 on prouve $t \models |t|$ implique $at \models a^\circ |t|$.

   3.2 Si $|t| \notin$ STRICT alors $|t| \equiv \tau^\circ |t'|$ pour un $|t'|$ et $|at| \equiv a^\circ |t'|$. Comme en 2.2 ceci implique

   $t \models |t'|$,

   $at \models \langle a \wedge E|t'| \rangle [a \wedge E|t'| \vee |\hat{\uparrow} t| \backslash a]$ par (P2),

   $at \models a^\circ |t'|$.

4. Soient $|t_1|, |t_2|$ des formules telles que $t_1 \models |t_1|$ et $t_2 \models |t_2|$.

   4.1 Si $|t_1| = [\bot]$ ou $|t_2| = [\bot]$ alors $t_1 + t_2 \models |t_1 + t_2|$ par (P6).

   4.2 Sinon $|t_1| = \bigwedge_{i \in I1} \langle a_i \wedge E|t_i| \rangle [|\hat{\uparrow} t_i|]$ et $|t_2| = \bigwedge_{i \in I2} \langle b_i \wedge E|t_i'| \rangle [|\hat{\uparrow} t_2|]$

   Par (P3) et (P5) on obtient $t_1 + t_2 \models f$ pour

   $f = \bigwedge_{i \in I1} \langle a_i \wedge E|t_i| \rangle \bigwedge_{i \in I2} \langle b_i \wedge E|t_i'| \rangle [|\hat{\uparrow} t_1| \vee |\hat{\uparrow} t_2|]$

   et apparemment $f \supset |t_1| \oplus |t_2|$. □

<u>Preuve de la consistance</u>

La consistance de la méthode de traduction est déduite des lemmes 2 à 6, qui ont tous la même hypothèse, l'hypothèse d'induction de la preuve du théorème 5.

Soit F un ensemble de formules de l'image de $|\ |$ tel que

(1) $\forall |t| \in F\ \forall t' \in T[\Sigma]$ $|t'|$ sous-formule de $|t|$ implique $|t'| \in F$.

(2) $\forall |t| \in F\ \forall t' \in T[\Sigma]$ $t' \models |t|$ implique $t' \mathcal{S} t$.

Les lemmes suivants donnent des propriétés de F.

Lemme 2

$\forall |t| \in F\ \forall t' \in T[\Sigma]$ $t' \models [\,|\hat{t}|\,]$ implique $t + t' \mathcal{S} t$.

Preuve: Soit $t' \in T[\Sigma]$ tel que $t' \models [\,|\hat{t}|\,]$. Par le théorème 4 et les propriétés 2 (P3) et (P5) on obtient $t + t' \mathcal{S} t$. □

Lemme 3

$\forall |t| \in F\ \forall t' \in T[\Sigma]$ $t' \models E|t|$ implique $t' \mathcal{S} t$ ou $t' \mathcal{S} \tau t$.

Preuve: De la proposition 2 on déduit

$E|t| = \bigvee_{k \in N} Y_k$ où $Y_0 = |t|$ et $Y_{k+1} = G(\bigvee_{l<k} Y_l, |t|)$.

On prouve le lemme par induction sur k.

- Pour $k=0$ $Y_k = |t|$ et du fait que $|t| \in F$ $t' \models Y_k$ implique $t' \mathcal{S} t$.
- Soit k tel que $\forall l<k$ $t' \models Y_l$ implique $t' \mathcal{S} t$ ou $t' \mathcal{S} \tau t$.

  Supposons $t' \models Y_{k+1}$ où $Y_{k+1} = \langle \tau \wedge (\bigvee_{l<k} Y_l) \rangle [\,\tau \wedge (\bigvee_{l<k} Y_l) \vee |\hat{t}|\,]$. Ceci implique

  a) $\exists t_0$ $t' \xrightarrow{\tau} t_0$ et $t_0 \models \bigvee_{l<k} Y_l$ et par l'hypothèse d'induction $t_0 \mathcal{S} t$ ou $t_0 \mathcal{S} \tau t$.

  b) $\forall t_j$ $t' \xrightarrow{a} t_j$ implique ($a_j = \tau$ et $t_j \models \bigvee_{l<k} Y_l$ ou $a_j t_j \models [\,|\hat{t}|\,]$) alors

  $\forall t_j$ $t' \xrightarrow{a} t_j$ implique ($a_j t_j \mathcal{S} \tau t$ ou $a_j t_j + \tau t \mathcal{S} \tau t$) par l'hypothèse d'induction, (A5) et le lemme 2.

  t' est donc de la forme $t' = \tau t_0 + \Sigma a_j t_j$ où $\tau t_0 \mathcal{S} \tau t$ (A5) et

  $\tau t + \Sigma a_j t_j \mathcal{S} \tau t + t + \Sigma a_j t_j$ par (A6)

  $\mathcal{S} \tau t + t$ par (b)

  $\mathcal{S} \tau t$ par (A6).

  Par conséquent, $t' \mathcal{S} \tau t$. □

Lemme 4

$\forall |t| \in F \; \forall t' \in T[\Sigma]$ $t' \models \tau^\circ |t|$ implique $t' \mathcal{S}_\tau t$.

Preuve: Du fait que $\tau^\circ |t| = G(E|t|, |t|)$ on obtient que $t' \models \tau^\circ |t|$ implique $t' \models G(\bigvee_{k \in N} Y_k, |t|)$. Du fait que G est continu, on obtient $t' \models \bigvee_{k \in N} G(\bigvee_{i \leq k} Y_i, |t|)$ ce qui est équivalent à $]k \geq 0$ $t' \models G(\bigvee_{i \leq k} Y_i, |t|)$ équivalent à $]k \geq 0$ $t' \models Y_{k+1}$

De la preuve du lemme 3 on déduit $t' \mathcal{S}_\tau t$. □

Lemme 5

$\forall |t| \in F \; \forall t' \in T[\Sigma]$ $t' \models a^\circ |t|$ implique $t' \mathcal{S} at$.

Preuve: $a^\circ |t| = \langle a \wedge E|t| \rangle [a \wedge E|t| \vee |\hat{t}| \backslash a]$. $t' \models a^\circ |t|$ implique alors

a) $]t_0$ $t' \xrightarrow{a} t_0$ et $t_0 \models E|t|$ ce qui implique par le lemme 3 $t_0 \mathcal{S} t$ ou $t_0 \mathcal{S}_\tau t$, alors
$]t_0$ $t' \xrightarrow{a} t_0$ et $at_0 \mathcal{S} at$ par (A5).

b) $\forall t_j$ $t' \xrightarrow{a} t_j$ implique $a_j = a$ et ($t_j \models E|t|$ ou $at_j \models [|\hat{t}| \backslash a]$).
De $t_j \models E|t|$ on déduit comme en a) $at_j \mathcal{S} at$ et de $at_j \models [|\hat{t}| \backslash a]$ on déduit par la définition de $|\hat{t}| \backslash a$ que $\tau t_j \models [|\hat{t}|]$. Par le lemme 2 on obtient $t + \tau t_j \mathcal{S} t$, ce qui implique par (A7)
$at_j + at \; \mathcal{S} \; at_j + a(t + \tau t_j) \; \mathcal{S} \; at$.

De a) et b) on déduit que t' est de la forme $t' = at_0 + \Sigma at_j$ où $at_0 \mathcal{S} at$ et $at + \Sigma at_j \mathcal{S} at$ par (A3) et (A7). D'où on obtient $t' \mathcal{S} at$.

Lemme 6

$\forall |t_1|, |t_2| \in F$, $\forall t \in T[\Sigma]$ $t \models |t_1 + t_2|$ implique $t \mathcal{S} t_1 + t_2$.

Preuve: On a que $|t_j|$ pour $j = 1.2$ et $|t_1 + t_2|$ de la forme
$|t_j| = \bigwedge_{i \in I_j} \langle a_{ji} \wedge E|t_{ji}| \rangle [|\hat{t}_j|]$ pour $j = 1.2$.

$|t_1| \oplus |t_2| = \bigwedge_{i \in I_1} \langle a_{1i} \wedge E|t_{1i}| \rangle \bigwedge_{i \in I_2} \langle a_{2i} \wedge E|t_{2i}| \rangle [|\hat{t}_1| \vee |\hat{t}_2|]$ où

$I1' = \{i \in I1 \mid \exists j \in I2\ c(a_{1i}t_{1i}, a_{2j}t_{2j})\}$ et $I2' = \{i \in I2 \mid \exists j \in I1\ c(a_{2i}t_{2i}, a_{1j}t_{1j})\}$.
Du fait que $\sum_{i \in Ij} a_{ji}t_{ji} \models |t_j|$ et $|t_j| \in F$ pour $j=1,2$ on déduit que

$\sum_{i \in Ij} a_{ji}t_{ji} \mathcal{S} t_j$ pour $j=1,2$ donc $\sum_{i \in I1} a_{1i}t_{1i} + \sum_{i \in I2} a_{2i}t_{2i} \mathcal{S}\ t_1+t_2$ (1).

Supposons que $t \in T[\Sigma]$ et $t \models |t_1| \oplus |t_2|$. Soit $t = \sum_{i \in J} b_i t_i''$. Ceci implique

(2) $\forall i \in Ij\ \exists t_{ji}'\ t \xrightarrow{a} t_{ji}'$ et $a=a_{ji}$ et $t_{ji}' \models E|t_{ji}|$.
I.e. $at_{ji}' \mathcal{S} a_{ji}t_{ji}$ comme en lemme 6 pour $j=1,2$.

(3) $\forall t''(t \xrightarrow{b} t''$ implique $bt'' \models [\,|\hat{t}_1|\,]$ ou $bt'' \models [\,|\hat{t}_2|\,]$.
I.e. $t_1+t_2+bt'' \mathcal{S} t_1+t_2$ par le lemme 2.

(4) $\forall i \in I1-I1'\ \exists i \in I2'\ [\widehat{a_{1i} \circ |t_{1i}|}] \supset [\widehat{a_{2j} \circ |t_{2j}|}]$, ce qui implique $a_{1i}t_{1i} \mathcal{S} a_{2j}t_{2j}$ par le lemme 2.

(5) $\forall i \in I2-I2'\ \exists i \in I1'\ a_{1i}t_{1i} \mathcal{S} a_{2j}t_{2j}$ par symétrie.

Ainsi, on obtient

$t \mathcal{S} \sum_{i \in I1'} a_{1i}t_{1i}' + \sum_{i \in I2'} a_{2i}t_{2i}' + \sum_{i \in J} b_i t_i''$

$\mathcal{S} \sum_{i \in I1'} a_{1i}t_{1i} + \sum_{i \in I2'} a_{2i}t_{2i} + \sum_{i \in J} b_i t_i''$ par (2)

$\mathcal{S} \sum_{i \in I1} a_{1i}t_{1i} + \sum_{i \in I2} a_{2i}t_{2i} + \sum_{i \in J} b_i t_i''$ par (4) et (5)

$\mathcal{S} t_1 + t_2 + \sum_{i \in J} a_i t_i''$ par (1)

$\mathcal{S} t_1+t_2$ par (3). □

Théorème 5

(C2) $\forall t, t' \in T[\Sigma]$ $t' \models |t|$ implique $t' \mathcal{S} t$.

Preuve: Par induction sur la structure des formules.

1. $t' \models [\bot]$ implique $t' \mathcal{S}$ Nil (évident).
2. Soit F un ensemble de formules $|t|$ de l'image de $|\ |$ tel que

(1) $\forall |t| \in F\ \forall t' \in T[\Sigma]$ $|t'|$ sous-formule de $|t|$ implique $|t'| \in F$
(2) $\forall |t| \in F\ \forall t' \in T[\Sigma]$ $t' \models |t|$ implique $t' \mathcal{S} t$.

Pour toute formule $|t|$, obtenue à partir de $|t'| \in F$ par application de a°, τ° ou $\oplus$ ces propriétés sont préservées d'après les lemmes 4,5 et 6. Toute formule $|t|$ peut être obtenue à partir de $[\bot]$ par des applications de a°, τ° et $\oplus$, ce qui complète la preuve.

La preuve de la complétude

Pour démontrer la complétude de la méthode de traduction, il suffit de démontrer $|t| \equiv |t'|$ pour les axiomes (A1) à (A7) qui sont de la forme t=t'. Ceci est suffisant parce que l'on sait que (A1) à (A7) constituent une axiomatisation complète de $\mathcal{S}$. La complétude est prouvée à partir des lemmes 7 à 10 suivants.

Lemme 7

$\forall t,t' \in T[\Sigma]$ $[|\hat{t}|] \equiv [|\hat{t'}|]$ implique $t \mathcal{S} t'$.

Preuve: Du théorème 4 on déduit $t \models [|\hat{t}|]$ et $t' \models [|\hat{t'}|]$ ce qui implique $t \models [|\hat{t'}|]$ et $t' \models [|\hat{t}|]$. A cause du théorème 5 le lemme 2 est appliquable et on obtient $t+t' \mathcal{S} t$ et $t+t' \mathcal{S} t'$ ce qui implique $t \mathcal{S} t'$. □

Lemme 8

(A1') $|(t_1+t_2)+t_3| \equiv |t_1+(t_2+t_3)|$
(A2') $|t_1+t_2| \equiv |t_2+t_1|$
(A3') $|t+t| \equiv |t|$
(A4') $|t+Nil| \equiv |t|$

Preuve: (A2'), (A3') et (A4') sont évidents. Il reste à prouver (A1') dans le cas où $|t_i| \neq [\perp]$ pour i=1,2,3, c'est-à-dire on doit prouver $(|t_1| \oplus |t_2|) \oplus |t_3| \equiv |t_1| \oplus (|t_2| \oplus |t_3|)$.
Dans ce cas chaque $|t_i|$ est de la forme
$|t_i| = \bigwedge_{j \in J} \langle a_{ij} \wedge E | t_{ij} | \rangle [|\hat{t_i}|]$ pour i=1,2,3.
Un terme de la forme $\langle a \wedge E|t| \rangle$ de $|t_1|$ est éliminé en $|t_1| \oplus |t_2|$ ssi il est éliminé en $|t_1| \oplus (|t_2| \oplus |t_3|)$ parce que la relation définie par le prédicat c(at,bt') est transitive et antisymétrique □.

Lemme 9

(A5') $|a\tau t| \equiv |at|$.

Preuve: Si $|t| \in$STRICT, alors $|t| = \tau^\circ |t'|$ pour un $|t'| \in$STRICT. Ceci implique $|\tau t| = \tau^\circ |t'|$, $|a(\tau t)| = a^\circ |t'|$ et $|at| = a^\circ |t'|$.
Si $|t| \in$STRICT, alors $|\tau t| = \tau^\circ |t|$ ce qui implique $|a(\tau t)| = a^\circ |t|$ et $|at| = a^\circ |t|$. On a donc toujours $|at| = |a\tau t|$. □

Lemme 10

(A6') $|\tau t + t| = |\tau t|$

(A7') $|a(t_1 + \tau t_2) + at_2| = |a(t_1 + \tau t_2)|$

Preuve: Par induction sur la structure des formules.
Soit F un ensemble de formules $|t|$ de l'image de $|\ |$ tel que

1) $\forall |t| \in F\ \forall t' \in T[\Sigma]$ $|t'|$ sous-formule de $|t|$ implique $|t'| \in F$.
2) $\forall |t| \in F\ \forall a,b \in A$ $a^\circ |t| \in F$ implique $b^\circ |t| \in F$.
3) $\forall |t| \in F\ \forall t' \in T[\Sigma]$ $t' \overset{\beta}{\sim} t$ implique $|t'| = |t|$.

- $F = \{[\bot]\}$ est un tel ensemble.
- On considère un tel ensemble F, et on prouve que

$F' = F \cup \{|at|\ |\ |t| \in F, a \in A\} \cup \{|t+t'|\ |\ |t|, |t'| \in F\}$

satisfait 1),2) et 3). 1) et 2) sont satisfaits d'après la définition de F. On doit donc prouver 3). A cause des lemmes 8 et 9, il suffit de démontrer (A6') et (A7') de la forme $|t_1| = |t_2|$ où $|t_1| = a^\circ |t_1'|$ ou $|t_1| = |t_1'| \oplus |t_1''|$ pour $|t_1'|, |t_1''| \in F$. On doit montrer:

a) Pour $|t_1|$ (ou $|t_2|$) de la forme $\tau^\circ |t|$ on doit prouver:

(a1) $\forall |t_1| \in F\ |\tau t_1| = |\tau t_1 + t_1|$ (A6')

(a2) $\forall |t_2|, |t_1 + \tau t_2| \in F\ |\tau(t_1 + \tau t_2)| = |\tau(t_1 + \tau t_2) + \tau t_2|$ (A7')

Pour ceci il suffit de prouver

(a3) $\forall |t_2|, |t_1 + t_2| \in F\ |\tau(t_1 + t_2)| = |\tau(t_1 + t_2) + \tau t_2|$ (A8')

b) Pour $|t_1|$ (ou $|t_2|$) de la forme $a^\circ |t|$ on doit prouver:

(b) $\forall |t_2|, |t_1 + \tau t_2| \in F\ |a(t_1 + \tau t_2)| = |a(t_1 + \tau t_2) + at_2|$ (A7')

c) Les cas où $|t_1|$ (ou $|t_2|$) dans des occurrences de (A6') ou (A7') sont de la forme $|t| \oplus |t'|$ ont déjà été considérés en a) et b).

Preuve de (a3):

Si $|t_1+t_2| \in F$ et $|t_1+t_2| \notin$STRICT, alors $|\tau(t_1+t_2)| \equiv |t_1+t_2| \in F$ et rien est à prouver.

Si $|t_1+t_2|=[\bot]$ ou $|t_2|=[\bot]$, on obtient facilement le résultat.

Sinon $|t_2|$ est de la forme

$|t_2| = \bigwedge_{i\in I} \langle a_i \wedge E|t_i| \rangle [\bigvee_{i\in I} \widehat{a_i \circ |t_i|}]$ et

$|\tau(t_1+t_2)|$ est de la forme

$|\tau(t_1+t_2)| = \langle \tau \wedge E|t_1+t_2| \rangle [\tau \wedge E|t_1+t_2| \vee |\hat{t}_1| \vee |\hat{t}_2|]$.

On a $\forall i\in I\ [a_i \circ |t_i|] \supset [|\tau(t_1+t_2)|]$, ce qui implique

- $\forall i\in I\ c(a_i t_i, \tau(t_1+t_2))$ d'où l'on obtient le résultat
- ou $\exists i\in I\ [\widehat{a_i \circ |t_i|}] \equiv [|\widehat{\tau(t_1+t_2)}|]$. Ceci implique $a_i=\tau$ et $[|\hat{t}_2|] \equiv [\widehat{\tau \circ |t_i|}]$, et par $|t_2| \in F$ et le lemme 7 on obtient $|t_2| \equiv \tau \circ |t_i|$, une contradiction à l'hypothèse que $|t_2| \in$STRICT.

Preuve de (b):

Si $|t_1+\tau t_2| \in F$ et $|t_1+\tau t_2| \notin$STRICT, alors $|t_1+\tau t_2| \equiv \tau \circ |t'|$, pour un $|t'| \in F$ ce qui implique $|a(t_1+\tau t_2)| = a \circ |t'| \in F$ par (2).

Si $|t_1+\tau t_2| \in$STRICT, on peut supposer sans perte de généralité que $|t_2| \in$STRICT. On a que $|at_2|$ et $|a(t_1+\tau t_2)|$ sont de la forme

$|at_2| = \langle a \wedge E|t_2| \rangle [a \wedge E|t_2| \vee |\hat{t}_2| \setminus a]$ et

$|a(t_1+\tau t_2)| = \langle a \wedge E|t_1+\tau t_2| \rangle [a \wedge E|t_1+\tau t_2| \vee |\hat{t}_1| \setminus a \vee |\hat{t}_2| \setminus a \vee a \wedge E|t_2|]$

On a donc $[\widehat{a \circ |t_2|}] \supset [\widehat{a \circ |t_1+\tau t_2|}]$ ce qui implique

- ou $c(at_2, a(t_1+\tau t_2))$ d'où l'on obtient le résultat
- ou $[\widehat{a \circ |t_2|}] \equiv [\widehat{a \circ |t_1+\tau t_2|}]$. Par le lemme 7 on obtient $at_2 \mathcal{S} a(t_1+\tau t_2)$, ce qui implique $t_2 \approx t_1+\tau t_2$. Par la propriété 3 (2) ceci est le cas ssi $\tau t_2 \mathcal{S} t_1+\tau t_2$ ou $t_2 \mathcal{S} \tau(t_1+\tau t_2)$ ou $t_2 \mathcal{S} t_1+\tau t_2$. Le dernier cas implique $t_2 \mathcal{S} \tau t_2$. Du fait que $|t_2|, |t_1+\tau t_2| \in F$ on obtient $\tau \circ |t_2| \equiv |t_1+\tau t_2|$ ou $|t_2| \equiv \tau \circ |t_1+\tau t_2|$ ou $|t_2| \equiv \tau \circ |t_2|$, et toutes les trois possibilités contredisent $|t_2|, |t_1+\tau t_2| \in$STRICT. □

Théorème 6

(C3) $\forall t,t' \in T[\Sigma]$ $t \mathcal{S} t'$ implique $|t| \equiv |t'|$.

Preuve: Les lemmes 8,9 et 10. □

IV.3 Sémantique des termes récursifs de CCS

3.1 Les termes récursifs contrôlables de CCS

Considérons le langage de termes construit à partir de la signature $\Sigma=\{Nil,+\}\cup A$ définie en 2.2 et d'un ensemble de variables X de la manière suivante:

- Nil, $x\in X$ sont des termes,
- si t,t' sont des termes, alors
  at, t+t' et recx.t sont des termes.

On représente par $T[\Sigma,X]$ l'ensemble des termes fermés et bien gardés de ce langage. t est bien gardé signifie que toute occurrence d'une variable x dans t est sous la portée d'un opérateur $a\in A-\{\tau\}$, par exemple t=recx.a(x+recy.by) est bien gardé, mais pas t'=recx.ax+x. Comme en 2.2 pour tout $a\in A$ une relation $\xrightarrow{a}\subseteq T[\Sigma,X]\times T[\Sigma,X]$ est définie comme la plus petite relation satisfaisant

$at \xrightarrow{a} t$

$t_1 \xrightarrow{a} t'$ implique $t_1+t_2 \xrightarrow{a} t'$

$t_1 \xrightarrow{a} t'$ implique $t_2+t_1 \xrightarrow{a} t'$

$t[recx.t/x] \xrightarrow{a} t'$ implique $recx.t \xrightarrow{a} t'$ si x libre en t.

C'est-à-dire recx.t(x) est la plus petite solution de l'équation x=t(x). Le problème étudié est celui de la solution d'équations récursives de ce type. En suivant l'approche classique, nous caractérisons ces solutions par des limites de suites ordonnées. Pour cela, nous étendons CCS par l'adjonction d'une constante Ω représentant le processus le "moins défini". Sur cette extension nous définissons une relation de préordre $\ll$ telle que $\Omega:=\ll\cap\ll^{-1}$ est une relation de congruence dont la restriction sur CCS est la congruence observationnelle. Pour des raisons qui deviennent

évidentes par la suite, nous n'avons appliqué cette approche qu'au sous-ensemble $CT[\Sigma,X]$ des termes contrôlables de CCS. On définit

$$CT[\Sigma,X] := \{t\in T[\Sigma,X] \mid \exists t'\in T[\Sigma-\{\tau\},X]\ t \mathcal{S} t'\}$$ et également
$$CT[\Sigma] := \{t\in T[\Sigma] \mid \exists t'\in T[\Sigma-\{\tau\}]\ t \mathcal{S} t'\}.$$

Un terme est donc contrôlable s'il est congru à un terme t' sans occurrence de τ. D'après [Mi] Chap.9 les termes contrôlables t sont tels que l'exécution d'une τ-transition mène à un terme t' tel que $t \approx t'$.

3.2 L'extension $CT_\Omega[\Sigma]$ de CCS

Considérons le langage de termes $T_\Omega[\Sigma]$, l'algèbre libre sur la signature $\Sigma\cup\{\Omega\}$ où Ω est une constante.

Représentons par $\mathfrak{Q}$ la congruence induite par les axiomes (A1) à (A7) de 2.4 sur $T_\Omega[\Sigma]$. Evidemment, on a $\mathfrak{Q}/T[\Sigma] = \mathcal{S}$.

L'ensemble des termes contrôlables $CT_\Omega[\Sigma]$ de $T_\Omega[\Sigma]$ est défini par

$$CT_\Omega[\Sigma] := \{t\in T_\Omega[\Sigma] \mid \exists t'\in T_\Omega[\Sigma-\{\tau\}]\ t \mathfrak{Q} t'\}$$

Par la suite les termes de $CT_\Omega[\Sigma]$ seront interprétés comme des ensembles de termes de $CT[\Sigma]$ qui sont des unions de classes de $\mathcal{S}$. Les termes de $CT_\Omega[\Sigma]$ représentent des ensembles de processus de CCS tels que l'exécution d'une τ-transition à partir de n'importe quel sous-terme t mène à un terme t' tel que $t' \approx t$.
Prouvons d'abord deux propriétés de $CT_\Omega[\Sigma]$, qui sont importantes pour la suite.

Propriété 1

$\forall t\in CT_\Omega[\Sigma]$ $t \mathfrak{Q} at'$ ssi $t=\Sigma at_i$ où $at_i \mathfrak{Q} at'$.

Preuve: Evidemment $t\Omega at'$ implique $t=\Sigma at_j$. Supposons, que pour un i $at_j\Omega at'$. Ceci implique qu'il existe un j tel que $at_j\Omega at'$ et $at_j+t_j\Omega at_j$ par une application de (A7): sinon on aurait également $at_j\Omega at_j$. C'est-à-dire il existe $t_j'.t_j'$ tels que $at_j'\Omega at_j$ et $at_j\Omega a(t_j'+\tau t_j')$ ce qui implique $at_j+at_j\Omega at_j$ par (A7). Mais du fait que $at_j\Omega at_e$ pour un t_e sans occurrence de τ, ceci implique $a(t_j'+\tau t_j')\Omega at_j'$. D'où l'on déduit $at_j\Omega at'$. □

Propriété 2

$\forall t\epsilon CT_\Omega[\Sigma]$ $t\Omega t_1+t_2$ implique $\exists t_1'.t_2'$ tels que $t\Omega t_1'+t_2'$ seulement par (A1) à (A4) et $t_1'\Omega t_1$ et $t_2'\Omega t_2$.

Preuve: Notons par $t\Omega' t'$ le fait que $t\Omega t'$ seulement par (A1) à (A4). Si $t_1\Omega' t_1'+\Omega$, alors $t\Omega' t'+\Omega$ pour un t', parce qu'il n'existe pas d'axiome pour éliminer Ω. Supposons donc que $t\Omega t_1+t_2$, $t_1=\Sigma a_j t_j$ et $t_2=\Sigma b_j t_j'$. t est également de la forme $t=\Sigma c_j t_j''$. De la preuve de la propriété 1 on déduit que $\Sigma c_j t_j''\Omega'\Sigma a_j t_j+\Sigma b_j t_j$. □

Définition 1

Soit $t\epsilon T_\Omega[\Sigma]$ avec $n>0$ occurrences de Ω et $\vec{p}=(p_1,\ldots p_n)$ un n-uplet de termes de $T[\Sigma]$. On définit l'opération de substitution $*$ telle que $t*\vec{p}$ est le terme de $T[\Sigma]$ obtenu à partir de t en substituant dans l'ordre les Ω par les p_j. Pour $t\epsilon T[\Sigma]$ on définit $t*\vec{p}=t$.

Définition 2

On définit la fonction prol: $CT_\Omega[\Sigma] \rightarrow 2^{CT[\Sigma]}$ telle que

pour $t\epsilon CT_\Omega[\Sigma]$ $\quad prol(t)=\{t'\epsilon CT[\Sigma] \mid \exists t_c\ t_c\Omega t\ \exists\vec{p}\ t'\mathcal{R}t_c*\vec{p}\}$.

On appelle prol(t) le prolongement de t.

Propriété 3

prol(Ω) = CT[Σ].

Preuve: évidente. □

Propriété 4

Pour $t\in CT_\Omega[\Sigma]$ prol(at) = $\{t' \mid t' \mathcal{R} \Sigma_{i\in I} at_i$ pour $I\subseteq N$ fini et $t_i\in prol(t)\}$.

Preuve:

- Si $t' \mathcal{R} \Sigma at_i$ et $t_i\in prol(t)$, on a:
  $t_i \mathcal{R} t_i' * \vec{p}_i$ pour $t_i' \Omega t$ et $t_i' * \vec{p}_i$ défini.
  Donc $t' \mathcal{R} \Sigma a(t_i' * \vec{p}_i) \mathcal{R} (\Sigma at_i') * \vec{p}$ où $\vec{p} = (\vec{p}_1, \ldots \vec{p}_n)$. Ceci implique $t'\in prol(at)$.

- Si $t'\in prol(at)$ alors $t' \mathcal{R} t'' * \vec{p}$ où $t'' \Omega at$. Par la propriété 1 on a $t''=\Sigma at_i$ où $at_i \Omega at$. On peut supposer que t ne contient pas de τ, mais on n'a pas nécessairement $t \Omega t_i$ parce que t_i peut dans le cas général être de forme $t_i=\tau t_i'+t_i''$ où $t_i' \Omega t$ ou $t_i' \Omega \tau t$. De la preuve de la propriété 2 on déduit que $a(\tau t_i'+t_i'') \Omega at_i$ et également $a(\tau t_i'+t_i'') * \vec{p} \mathcal{R} at_i * \vec{p}$, parce que l'élimination de τ n'a pas changé le nombre et l'ordre des occurrences de Ω en t_i. Ainsi, on peut obtenir par élimination successive des τ un t_i''' tel que $t_i''' \Omega t$ et $t_i''' * \vec{p} \mathcal{R} t * \vec{p}$. On obtient alors,
  $t' \mathcal{R} (\Sigma at_i) * \vec{p} = \Sigma a(t_i * \vec{p}_i) \mathcal{R} \Sigma a(t_i''' * \vec{p}_i)$ où $t_i''' \Omega t$.
  Ceci implique $t_i''' * \vec{p}_i \in prol(t)$, d'où on déduit le résultat. □

Propriété 5

Pour $t_1, t_2\in CT_\Omega[\Sigma]$ $prol(t_1+t_2) = \{t \mid t \mathcal{R} t_1'+t_2'$ pour $t_i'\in prol(t_i)$ $i=1,2\}$.

Preuve:

- Si $t' \mathcal{R} \Sigma at_1'+t_2'$ et $t_i'\in prol(t_i)$ pour $i=1,2$, alors
  $t_i \mathcal{R} t_i'' * \vec{p}_i$ où $t_i'' \Omega t_i$ pour $i=1,2$. Ceci implique
  $t_1'+t_2' \mathcal{R} t_1'' * \vec{p}_1 + t_2'' * \vec{p}_2 \mathcal{R} (t_1''+t_2'') * (\vec{p}_1, \vec{p}_2)$. Donc $t\in prol(t_1+t_2)$.
- Si $t\in prol(t_1+t_2)$ alors $t \mathcal{R} t'' * \vec{p}$ où $t'' \Omega t_1+t_2$. D'après la propriété 2 on

a $t^{"}\Omega t_1^{"}+t_2^{"}$ uniquement par les axiomes (A1) à (A4) et $t_1^{"}\Omega t_1$ et $t_2^{"}\Omega t_2$. Ceci implique que l'on peut obtenir de façon évidente à partir de $\vec{p}$ un $\vec{p}'$ tel que $t^{"}*\vec{p} \mathcal{S} (t_1^{"}+t_2^{"})*\vec{p}' \mathcal{S} t_1^{"}*\vec{p}_1'+t_2^{"}*\vec{p}_2' = t_1'+t_2'$ où $\vec{p}'=(\vec{p}_1'.\vec{p}_2')$ et $t_1'\epsilon prol(t_1)$. Ceci complète la preuve. □

Propriété 6

Pour $t\epsilon CT_\Omega[\Sigma]$ prol(t) est une union de classes de la congruence $\mathcal{S}$ sur $CT_\Omega[\Sigma]$.

Preuve: Evidente à partir des propriétés 3 à 5.

Définition 3

Sur $CT_\Omega[\Sigma]$ on définit les relations binaires $\ll$ et Ω telles que

a) $t\ll t'$ ssi $prol(t)\subseteq prol(t')$

b) $\Omega = \ll\cap\ll^{-1}$.

Evidemment, $\ll$ est une relation de préordre sur $CT_\Omega[\Sigma]$ et Ω est une relation d'équivalence. Remarquons que les restrictions de Ω et de $\ll$ sur $CT[\Sigma]$ sont la restriction de $\mathcal{S}$ sur $CT[\Sigma]$.

Propriété 7

Pour $t_i.t_i'\epsilon CT_\Omega[\Sigma]$ pour i=1.2 on a

a) $t_1\ll t_1'$ implique $at_1\ll at_1'$

b) $t_1\ll t_1'$ et $t_2\ll t_2'$ implique $t_1+t_2\ll t_1'+t_2'$.

Preuve:

a) $t_1\ll t_1'$ est équivalent à $prol(t_1)\subseteq prol(t_1')$ d'où l'on obtient $t_1^{"}\epsilon prol(at_1)$ implique $t_1^{"}\mathcal{S}\Sigma at_{//}$ où $t_{//}\epsilon prol(t_1)$ par la propriété 4. Donc $t_{//}\epsilon prol(t_1')$, ce qui implique $t_1^{"}\epsilon prol(at_1')$. Ceci implique $prol(at_1)\subseteq prol(at_1')$ ce qui est équivalent à $at_1\ll at_1'$.

b) $t_i\ll t_i'$ pour i=1.2 est équivalent à $prol(t_i)\subseteq prol(t_i')$ pour i=1.2. $t\epsilon prol(t_1+t_2)$ implique $t\mathcal{S}t_1^{"}+t_2^{"}$ où $t_1^{"}\epsilon prol(t_1)$ par la propriété 5. Ceci

implique $t_j'' \in prol(t_j')$, donc $t \in prol(t_1'+t_2')$. Comme en a) on obtient $t_1+t_2 \ll t_1'+t_2'$. □

Remarquons, que $\mathfrak{Q} \subseteq \Omega$ et cette inclusion est propre. Evidemment, deux termes t_1, t_2 tels que $t_1 \mathfrak{Q} t_2$ ont les mêmes prolongements d'après la définition 2. Mais on n'a pas $\Omega \subseteq \mathfrak{Q}$ parce que par exemple pour $t_1 = a\Omega + aa\Omega + aaa\Omega$ et $t_2 = a\Omega + aaa\Omega$ on a $t_1 \Omega t_2$, mais pas $t_1 \mathfrak{Q} t_2$.

3.3 Propriétés "opérationnelles" de $\ll$ sur $CT_\Omega[\Sigma,X]$.

Dans ce paragraphe nous donnons des propriétés opérationnelles de $\ll$ sur $CT_\Omega[\Sigma,X]$ dont nous avons besoin dans les paragraphes suivants. Pour ceci on définit pour tout $a \in A$ une relation $\xrightarrow{a} \subseteq T_\Omega[\Sigma,X] \times T_\Omega[\Sigma,X]$ et pour $a \in A \setminus \{\tau\}$ une relation $\stackrel{a}{\Rightarrow} \subseteq CT_\Omega[\Sigma,X] \times CT_\Omega[\Sigma,X]$, telles que Ω n'a pas de $\xrightarrow{a}$ ou $\stackrel{a}{\Rightarrow}$ dérivations.
On définit une relation $\overset{W}{\approx} \subseteq CT_\Omega[\Sigma,X] \times CT_\Omega[\Sigma,X]$ comme la limite de relations $\overset{W}{\approx}_k \subseteq CT_\Omega[\Sigma,X] \times CT_\Omega[\Sigma,X]$ telle que la restriction de $\overset{W}{\approx}$ sur $CT[\Sigma]$ est l'équivalence faible [BR] qui est $\overset{\mathcal{F}}{\approx}$ sur $CT[\Sigma]$. C'est à l'aide des relations $\overset{W}{\approx}_k$ et $\overset{W}{\approx}$ que l'on distingue Ω et Nil.

Pour tout $a \in A$ on définit une relation $\xrightarrow{a}$ exactement comme en 2.2. et pour tout $a \in A-\{\tau\}$ une relation $\stackrel{a}{\Rightarrow}$ est définie comme la restriction de $\xrightarrow{a}(\xrightarrow{\tau})^*$ sur $CT_\Omega[\Sigma,X] \times CT_\Omega[\Sigma,X]$.

Notation

Pour $t \in CT_\Omega[\Sigma,X]$ on note par $l(t)$ la profondeur de l'arbre qui représente t, c'est-à-dire la longueur maximale d'une séquence $s \in (A-\{\tau\})^*$ telle que $t \stackrel{s}{\Rightarrow} t'$. Par Ω-$p(t)$ on note la profondeur **minimale** à laquelle se trouve une occurrence de Ω, c'est-à-dire Ω-$p(t)$ est la longueur minimale des séquences $s \in (A-\{\tau\})^*$ telles que $t \stackrel{s}{\Rightarrow} \Omega$. Pour $t \in CT[\Sigma]$, on pose Ω-$p(t) := \infty$.

Définition 4

On définit la relation $\overset{W}{\sim} = \bigcap_{k\in N} \overset{W}{\sim}_k \subseteq CT_\Omega[\Sigma.X]\times CT_\Omega[\Sigma.X]$, où les $\overset{W}{\sim}_k$ pour $k\in N$ sont définis par

- $t_1 \overset{W}{\sim}_0 t_2$ ssi $\Omega-\rho(t_i)>0$ pour $i=1,2$
- $t_1 \overset{W}{\sim}_{k+1} t_2$ ssi $\forall a\in A-\{\tau\}$ $(t_1 \overset{a}{\Rightarrow} t_1'$ implique $\exists t_2'(t_2 \overset{a}{\Rightarrow} t_2'$ et $t_1' \overset{W}{\sim}_k t_2')$ et
  $(t_2 \overset{a}{\Rightarrow} t_2'$ implique $\exists t_1'(t_1 \overset{a}{\Rightarrow} t_1'$ et $t_1' \overset{W}{\sim}_k t_2')$.

Les relations $\overset{W}{\sim}_k$ sont des relations d'équivalence pour tout $k\in N$. En effet, $\overset{W}{\sim}$ est l'équivalence faible sur CCS parce que $\overset{W}{\sim}$ n'est défini que pour des termes t tels que $\Omega-\rho(t)=\infty$, c'est-à-dire $t\in CT[\Sigma]$.

Proposition 1

$\forall t\in CT_\Omega[\Sigma.X]$ $\Omega-\rho(t)>k$ on a $\forall t',t''\in prol(t)$ $t' \overset{W}{\sim}_k t''$.

Preuve: par induction sur $k\in N$.

- Pour $t\in CT_\Omega[\Sigma.X]$ tel que $\Omega-\rho(t)>0$ on a évidemment $\forall t',t''\in prol(t)$ $t' \overset{W}{\sim}_0 t''$.
- Supposons que pour un $k\in N$ $\forall t\in CT_\Omega[\Sigma.X]$ tel que $\Omega-\rho(t)>k$ $\forall t',t''\in prol(t)$ $t' \overset{W}{\sim}_k t''$.
- Soit $t\in CT_\Omega[\Sigma.X]$ tel que $\Omega-\rho(t)>k+1$ et $t',t''\in prol(t)$.
  t est donc de la forme $t \overset{\Omega}{=} \sum_{i\in I} a_i t_i$ où $t_i\in CT_\Omega[\Sigma.X]$ et $\Omega-\rho(t_i)>k$. Par les propriétés 4 et 5 on obtient

  $prol(t) = \{t' \mid t' \overset{\mathcal{S}}{=} \sum_{i\in I} \sum_{j\in J_i} a_i t_{ij}'$ où $J_i\neq\emptyset$ et $t_{ij}'\in prol(t_i)\}$.

  Pour $t',t''\in prol(t)$ on obtient donc:

  $t' \overset{a}{=\!>} t_a'$ implique $\exists t_a''$ $t'' \overset{a}{=\!>} t_a''$ $\exists i\in I$ $a=a_i$ et $t_a',t_a''\in prol(t_i)$ ce qui implique par l'hypothèse d'induction $t_a' \overset{W}{\sim}_k t_a''$.
  On obtient alors
  $t' \overset{a}{=\!>} t_a'$ implique $\exists t_a''$ $t'' \overset{a}{=\!>} t_a''$ et $t_a' \overset{W}{\sim}_k t_a''$.
  $t'' \overset{a}{=\!>} t_a''$ implique $\exists t_a'$ $t' \overset{a}{=\!>} t_a'$ et $t_a' \overset{W}{\sim}_k t_a''$.
  Ceci implique $t' \overset{W}{\sim}_{k+1} t''$. □

Proposition 2

$\forall t_1, t_2 \in CT_\Omega[\Sigma]$ tels que $\Omega - p(t_i) > k$ pour i=1,2, on a

$t_1 \overset{W}{\approx}_k t_2$ $\forall t_i' \in prol(t_i)$ pour i=1,2 implique $t_1 \overset{W}{\approx}_k t_2$.

Preuve: On fait une preuve par induction sur k.

- Pour k=0 rien est à prouver.
- Supposons que pour un $k \in \mathbf{N}$ $\forall t_1, t_2 \in CT_\Omega[\Sigma]$ tels que $\Omega - p(t_i) > k$ pour i=1,2, on a

  $t_1 \overset{W}{\approx}_k t_2$ $\forall t_i' \in prol(t_i)$ pour i=1,2 implique $t_1 \overset{W}{\approx}_k t_2$.
- Soient $t_1, t_2 \in CT_\Omega[\Sigma]$ tels que $\Omega - p(t_i) > k+1$ pour i=1,2 et $t_1 \overset{W}{\approx}_{k+1} t_2$ $\forall t_i' \in prol(t_i)$ pour i=1,2.

  On prouve $t_1 \overset{W}{\approx}_{k+1} t_2$.

  $t_1 \overset{a}{\Rightarrow} t_{1a}$ implique $\forall \vec{p_1}$ $t_1 * \vec{p_1} \overset{a}{\Rightarrow} t_{1a} * \vec{p_1}$ ce qui implique $\forall \vec{p_2}$ $\exists t_{2p}$ $(t_2 * \vec{p_2} \overset{a}{\Rightarrow} t_{2p} * \vec{p_2}$ et $t_{1a} * \vec{p_1} \overset{W}{\approx}_k t_{2p} * \vec{p_2})$ parce que $t_1 * \vec{p_1} \overset{W}{\approx}_{k+1} t_2 * \vec{p_2}$. Parce que $\Omega - p(t_{1a}) > k$ et $\Omega - p(t_{2p}) > k$, à partir de la proposition 1 et du fait que $t_{1a} * \vec{p_1} \overset{W}{\approx}_k t_{2p} * \vec{p_2}$ on obtient $t_{1a} \overset{W}{\approx}_k t_{2p}$ $\forall t_{1a}' \in prol(t_{1a})$, $\forall t_{2p}' \in prol(t_{2p})$.

  Ceci implique par l'hypothèse d'induction $t_{1a} \overset{W}{\approx}_k t_{2p}$.

  Par un raisonnement symétrique on obtient également

  $t_2 \overset{a}{\Rightarrow} t_{2a}$ implique $\exists t_{1p}$ $t_1 \overset{a}{\Rightarrow} t_{1p}$ et $t_{1p} \overset{W}{\approx}_k t_{2a}$. Ceci implique le résultat. □

Proposition 3

$\forall t_1, t_2 \in CT_\Omega[\Sigma]$ tels que $\Omega - p(t_i) > k$ pour i=1,2 on a

$t_1 \ll t_2$ implique $t_1 \overset{W}{\approx}_k t_2$.

Preuve:

$t_1 \ll t_2$ est équivalent à $prol(t_1) \subseteq prol(t_2)$. Par la proposition 1 on obtient $t_1 \overset{W}{\approx}_k t_2$ $\forall t_i' \in prol(t_i)$, ce qui implique par la proposition 2 $t_1 \overset{W}{\approx}_k t_2$. □

Proposition 4

$\forall t_1, t_2 \in CT_\Omega[\Sigma]$. $t_1 \ll t_2$ implique

$t_2 \xrightarrow{a} t_{2a}$ implique $\exists t_{1a}$ $t_1 \xrightarrow{a} t_{1a}$ et $t_{1a} \ll t_{2a}$.

Preuve: Soient $t_1, t_2 \in CT_\Omega[\Sigma]$. $t_1 \ll t_2$ et $t_2 \xrightarrow{a} t_{2a}$.

- Supposons $\nexists t_{1a}$ $t_1 \xrightarrow{a} t_{1a}$. Par les propriétés 4 et 5 ceci implique qu'il existe un $t_1' \in prol(t_1)$ tel que $t_1' \notin prol(t_2)$ ce qui contredit $t_1 \ll t_2$.
- On a donc $\exists t_{1a}$ $t_1 \xrightarrow{a} t_{1a}$. Supposons que pour tout t_{1a} $t_1 \xrightarrow{a} t_{1a}$ implique $t_{1a} \not\ll t_{2a}$. ce qui est équivalent à $\exists t_{1a}' \in prol(t_{1a})$ tel que $t_{1a}' \notin prol(t_{2a})$. Par les propriétés 4 et 5 on obtient alors $\exists t_1' \in prol(t_1)$ tel que $t_1' \notin prol(t_2)$. Ceci contredit le fait que $t_1 \ll t_2$. □

Proposition 5

$\forall t_1, t_2 \in CT_\Omega[\Sigma]$ tels que $\Omega\text{-}p(t_1) > k$ et $l(t_2) < k$ on a

$\forall t_1' \in prol(t_1)$ $\exists t_2' \in prol(t_2)$ $t_1' \mathrel{\mathcal{R}_k} t_2'$ implique $t_1 \ll t_2$. (1)

Preuve: On fait une preuve par induction sur $k \in \mathbb{N}$.

- Pour $k=0$ rien est à prouver parce que $l(t_2) < 0$ n'est pas possible.
- Supposons que (1) est satisfait pour un $k \in \mathbb{N}$.
- Soient $\forall t_1, t_2 \in CT_\Omega[\Sigma]$ tels que $\Omega\text{-}p(t_1) > k+1$ et $l(t_2) < k+1$ et $\forall t_1' \in prol(t_1)$ $\exists t_2' \in prol(t_2)$ $t_1' \mathrel{\mathcal{R}_{k+1}} t_2'$. On prouve $t_1 \ll t_2$.
  t_1 est de la forme générale $t_1 \Omega \sum_{i \in I} a_i t_{1i}$ où $t_{1i} \in CT_\Omega[\Sigma]$ et $\Omega\text{-}p(t_{1i}) > k$ pour $i \in I$.
- Si $t_2 \Omega \Omega$. alors $t_1 \ll t_2$.
- Sinon t_2 est de la forme générale $t_2 \Omega \sum_{j \in J} b_j t_{2j}$ où $t_{2j} \in CT_\Omega[\Sigma]$ et $l(t_{2j}) < k$ pour $j \in J$.
  Soient $t_1' \in prol(t_1)$ tel que $t_1' \mathrel{\mathcal{R}_{k+1}} t_2'$. Par les propriétés 4 et 5 t_1' et t_2' sont de la forme générale
  $t_1' \mathrel{\mathcal{S}} \sum_{i \in I} \sum_{j \in I_i} a_i\, t_{1ij}$ où $t_{1ij} \in prol(t_{1i})$ et $I_i \neq \emptyset$ (a)
- $t_2' \mathrel{\mathcal{S}} \sum_{i \in J} \sum_{j \in J_i} b_i\, t_{2ij}$ où $t_{2ij} \in prol(t_{2i})$ et $J_i \neq \emptyset$ (b).

  $t_1' \mathrel{\mathcal{R}_{k+1}} t_2'$ implique $\forall i,j\ \exists n,m$ $a_i = b_n$ et $t_{1ij} \mathrel{\mathcal{R}_k} t_{2nm}$ ce qui implique par la proposition 1 que $\forall t_{1i}' \in prol(t_{1i})$ $t_{1i}' \mathrel{\mathcal{R}_k} t_{2nm}$. ce qui implique par

l'hypothèse d'induction $t_{1i} \ll t_{2n}$. On a donc trouvé que

$t_1' \overset{a}{\Rightarrow} t_1''$ implique $\exists n\ t_1'' \in prol(t_{2n})$ et $b_n = a$ (2).

Par un raisonnement symétrique, on trouve également

$t_2' \overset{a}{\Rightarrow} t_2''$ implique $\exists i\ t_2'' \in prol(t_{1i})$ et $a_i = a$ (3).

Par les propriétés 4 et 5 on déduit de (2) et (3) que $t_1' \in prol(t_2)$.

Du fait que t_1' est un élément quelconque de $prol(t_1)$, on obtient $t_1 \ll t_2$. □

3.4 Complétion $CT_\Omega^\infty[\Sigma]$ de $CT_\Omega[\Sigma]$

Nous définissons une extension de $CT_\Omega[\Sigma]$ en y ajoutant les limites de suites décroissantes de termes de $CT_\Omega[\Sigma]$ qui seront utilisées pour caractériser les solutions d'équations récursives sur $CT_\Omega[\Sigma]$ et finalement pour caractériser les termes "infinis" de $CT[\Sigma,X]$.

- Soit $\mathcal{S}$ l'ensemble des suites $\{t_i\}_{i\in N}$ sur $CT_\Omega[\Sigma]$, telles que
  - $\forall i \in N\ t_{i+1} \ll t_i$, autrement dit la séquence $\{t_i\}$ est décroissante.
  - $\forall k \in N\ \exists l(k)\ \forall i > l(k)\ \Omega\text{-}p(t_i) > k$, c'est-à-dire les Ω figurant dans les t_i se trouvent en profondeur de plus en plus grande.
- Soit $B = \{ \bigcap_{i\in N} t_i \mid \{t_i\} \in \mathcal{S} \}$ un ensemble de constantes, mais où $\bigcap_{i\in N}$ sera interprété comme une opération de borne inférieure.

Considérons l'algèbre de termes $T_\Omega^\infty[\Sigma]$ sur la signature $\Sigma \cup \{\Omega\} \cup B$.

Comme en 3.2 on représente par $\mathfrak{Q}$ la congruence induite par les axiomes (A1) à (A7) sur $T_\Omega^\infty[\Sigma]$. On définit l'ensemble des termes contrôlables $CT_\Omega^\infty[\Sigma]$ de $T_\Omega^\infty[\Sigma]$ par

$$CT_\Omega^\infty[\Sigma] = \{ t \in T_\Omega^\infty[\Sigma] \mid \exists t'\ \mathfrak{Q}\, t \text{ et } t' \in T_\Omega^\infty[\Sigma - \{\tau\}] \}.$$

Pour tout $a \in A$ une relation $\overset{a}{\rightarrow} \subseteq T_\Omega^\infty[\Sigma] \times T_\Omega^\infty[\Sigma]$ est définie exactement comme $\overset{a}{\rightarrow} \subseteq T_\Omega[\Sigma] \times T_\Omega[\Sigma]$.

Pour tout $a \in A-\{\tau\}$ une relation $\stackrel{a}{\Rightarrow} \subseteq CT^{\infty}_{\Omega}[\Sigma] \times CT^{\infty}_{\Omega}[\Sigma]$ est définie comme la plus petite relation satisfaisant

- pour $t,t' \in CT_{\Omega}[\Sigma]$ $t \stackrel{a}{\rightarrow} (\stackrel{\tau}{\rightarrow})^* t'$ implique $t \stackrel{a}{\Rightarrow} t'$.
- $\Omega t_j \stackrel{a}{\Rightarrow} \Omega t_j'$ ssi $(\forall i \in N\ t_i \neq \Omega$ implique $t_i \stackrel{a}{\Rightarrow} t_i')$ et $\{t_j\}.\{t_j'\} \in S$

On définit $\Omega-p(\Omega t_j)=\infty$.

Une séquence de relations $\approx_k \subseteq CT^{\infty}_{\Omega}[\Sigma] \times CT^{\infty}_{\Omega}[\Sigma]$ et $\approx = \bigcap_{k \in N} \approx_k$ sont définies exactement comme sur $CT_{\Omega}[\Sigma]$.

Remarques:

- Si $\{t_j\} \in S$ est une séquence qui devient stationnaire à partir d'un certain rang k, par exemple $t_0.t_1....t_k.t_k.....$ alors $t_k \in CT[\Sigma]$ et $\Omega t_j \approx t_k$.
- Pour un terme Ωt_j on a $\Omega-p(\Omega t_j)=\infty$, parce que, soit la séquence $\{t_j\}$ est stationnaire, soit les occurrences de Ω se trouvent dans une profondeur de plus en plus grande, c'est-à-dire, il ne peut pas exister une séquence $s \in (A-\{\tau\})^*$ telle qu'il existe un indice n tel que $\forall i>n$ $t_j \stackrel{s}{\Rightarrow} \Omega$, ce qui serait nécessaire pour $\Omega t_j \stackrel{s}{\Rightarrow} \Omega$.
- On voit que la relation $\approx$ sur $CT^{\infty}_{\Omega}[\Sigma]$ n'est définie que pour les termes t tels que $\Omega-p(t)=\infty$, c'est-à-dire les termes construits à partir de Nil et des éléments de B.
- La restriction des relations $\approx$ et $\approx_k$ sur $CT_{\Omega}[\Sigma]$ sont les relations $\approx$ et $\approx_k$ sur $CT_{\Omega}[\Sigma]$ définies en 3.3 ce qui justifie que l'on ne les distingue pas par leurs noms. On a également que la restriction de $\approx$ sur $CT[\Sigma]$ est l'équivalence faible, qui est égale à la restriction de $\mathcal{S}$ sur $CT[\Sigma]$.

Proposition 6

Pour $\{t_j\} \in S$ $\forall k \in N$ $\exists l(k)$ $\forall n>l(k)$ $\Omega t_j \approx_k t_n$.

Preuve: On fait une preuve par induction sur k.

- Pour une séquence $\{t_j\} \in S$ $\exists n$ $\forall i>n$ $\Omega-p(t_j)>0$ ce qui implique $\forall i>n$ $\Omega t_j \approx_0 t_j$.

- On sait que pour une séquence $\{t_j\}$ donnée $\exists i(k)\ \forall l \geq l(k)\ \Omega\text{-}p(t_l) > k$ $\forall k \in \mathbf{N}$.

  Supposons que pour un k on a pour toute séquence $\{t_j\}$ $\cap t_l \overset{w}{\approx}_k t_n$ $\forall n \geq l(k)$.

- Soit $\{t_j\} \in \mathcal{S}$ et $n := l(k+1)$.

  $\cap t_j \overset{a}{\Rightarrow} \cap t_j'$ implique $\forall j \geq n\ t_j \overset{a}{\Rightarrow} t_j'$ et $\Omega\text{-}p(t_j') \geq k$ par la définition de $\overset{a}{\Rightarrow}$. Ceci implique par l'hypothèse d'induction $\cap t_j' \overset{w}{\approx}_k t_j'$ $\forall j \geq n$. On a donc

  (1) $\cap t_j \overset{a}{\Rightarrow} \cap t_j'$ implique $\forall j \geq n\ t_j \overset{a}{\Rightarrow} t_j'$ et $\cap t_j' \overset{w}{\approx}_k t_j$.

  $\forall j \geq n\ t_j \overset{a}{\Rightarrow} t_j'$ implique $\exists t_{j+1}'\ t_{j+1} \overset{a}{\Rightarrow} t_{j+1}'$ et $t_{j+1}' \ll t_j'$. On peut donc trouver $\{t_j'\} \in \mathcal{S}$ tel que $t_j \overset{a}{\Rightarrow} t_j'$ $\forall l \geq j$. Ceci implique $\cap t_j \overset{a}{\Rightarrow} \cap t_j'$ par la définition de $\overset{a}{\Rightarrow}$. Par l'hypothèse d'induction on a $\cap t_j' \overset{w}{\approx}_k t_l$ $\forall l \geq j$, donc

  (2) $\forall j \geq n\ t_j \overset{a}{\Rightarrow} t_j'$ implique $\cap t_j \overset{a}{\Rightarrow} \cap t_j'$ et $\cap t_j' \overset{w}{\approx}_k t_j'$.

  De (1) et (2) on déduit $\cap t_j \overset{w}{\approx}_{k+1} t_j$ $\forall j \geq n$. □

On étend la relation $\ll$ sur $CT^{\infty}_{\Omega}[\Sigma]$ de la manière suivante.

Définition 5

Pour $\{t_j\}, \{t_j'\} \in \mathcal{S}$ on définit

$\cap t_j \ll \cap t_j'$ ssi $\forall m \in \mathbf{N}\ \exists n\ t_n \ll t_m'$.

On note $\Omega = \ll \cap \ll^{-1}$.

Proposition 7

Pour $\{t_j\}, \{t_j'\} \in \mathcal{S}$ on a

$\cap t_j \ll \cap t_j'$ implique $\cap t_j \overset{w}{\approx} \cap t_j'$.

Preuve: De la proposition 6 on déduit que pour tout k il existe l(k) tel que $\forall n \geq l(k)$ $\cap t_j \overset{w}{\approx}_k t_n$ et $\cap t_j' \overset{w}{\approx}_k t_n'$ (1).
$\cap t_j \ll \cap t_j'$ implique $\forall m\ \exists n(m)\ t_n \ll t_m'$. On peut choisir n et k de telle sorte que $n \geq l(k)$ et $\forall j \geq n\ \Omega\text{-}p(t_j) > k$ et $\Omega\text{-}p(t_j') > k$. Par les propositions 2 et 3 on obtient $\forall j \geq n\ t_j \overset{w}{\approx}_k t_j'$. Finallement, on obtient par (1) $\cap t_j \overset{w}{\approx}_k \cap t_j'$. Parce que k peut être choisi de plus en plus grand on a
$\forall k \in \mathbf{N}\ \cap t_j \overset{w}{\approx}_k \cap t_j'$, donc $\cap t_j \overset{w}{\approx} \cap t_j'$. □

Proposition 8

Pour $\{t_j\}.\{t_j'\} \in S$ on a

$\cap t_j \not\asymp \cap t_j'$ implique $\cap t_j \ll \cap t_j'$.

Preuve: $\cap t_j \not\asymp \cap t_j'$ est équivalent à $\forall k \in N\ \cap t_j \not\asymp_k t_j'$. De la définition de $\not\asymp_k$ et de $\stackrel{a}{\Rightarrow}$ on déduit $\forall k\ \exists n(k)\ \forall n > n(k)\ \Omega\text{-}p(t_n) > k$ et $\Omega\text{-}p(t_n') > k$ ce qui implique $t_n \not\asymp_k t_m\ \forall n,m > n(k)$.
On doit prouver $\forall m \in N\ \exists n(m)\ t_{n(m)} \ll t_m'$.
Soit $m \in N$ et k tel que $l(t_m') < k$.
On a $t_{n(k)} \not\asymp_k t'_{n(k)}$ et $t_{n(k)} \ll t_m$. Ceci implique par la proposition 1 $\forall t \in prol(t_{n(k)})$. $\forall t' \in prol(t_{n(k)}')\ t \not\asymp_k t'$ et $t' \in prol(t_m)$. Parce que $\Omega\text{-}p(t_{n(k)}) > k$ et $l(t_m') < k$ on obtient par la proposition 5 $t_{n(k)} \ll t_m$. □

Remarque: Pour les termes de $CT[\Sigma] \cup B$ les relations Ω, $\ll$ et $\not\asymp$ coïncident, comme on déduit des propositions 7 et 8.

La notion de prolongement peut être étendue sur $CT_\Omega^\infty[\Sigma]$ de la manière suivante.

Définition 6

- Pour $\{t_j\} \in S$ $prol(\cap t_j) = \{t' \mid t' \not\asymp \cap t_j\}$
- Pour $t \in CT_\Omega^\infty[\Sigma] - B$ $prol(t) = \bigcup_{t' \ll t} prol(t')$.

Propriété 8

Sur $CT_\Omega^\infty[\Sigma]$ $t' \ll t$ ssi $prol(t') \subseteq prol(t)$. □

Propriété 9

- $t_1 \not\asymp t_2$ implique $at_1 \not\asymp at_2\ \forall a \in A - \{\tau\}$.
- $t_1 \not\asymp t_2$ implique $t_1 + t_3 \not\asymp t_2 + t_3\ \forall t_3 \in CT_\Omega^\infty[\Sigma]$ tel que $\Omega\text{-}p(t_3) = \infty$. □

Proposition 9

Pour $\{t_i\} \in S$ $a(\cap t_i) \overset{W}{\approx} \cap(at_i)$.

Preuve: On a $a\cap t_i \overset{a}{\Rightarrow} \cap t_i$ et $\forall i \in N$ $at_i \overset{a}{\Rightarrow} t_i$. Par la propriété 7, on obtient $\{t_i\} \in S$ implique $\{at_i\} \in S$. Donc $\cap at_i \overset{a}{\Rightarrow} \cap t_i$. Ceci implique $a\cap t_i \overset{W}{\approx} \cap at_i$. □

Proposition 10

Pour $\{t_i\}.\{t_i'\} \in S$ $\cap(t_i+t_i') \overset{W}{\approx} \cap t_i+\cap t_i'$.

Preuve: Sans perte de généralité on peut supposer $t_i.t_i' \neq \Omega$, donc $\cap(t_i+t_i') \overset{a}{\Rightarrow} t''$ implique $\forall i \in N$ $t_i+t_i' \overset{a}{\Rightarrow} t_i''$ et $t''=\cap t_i''$. Ceci implique $\forall i \in N$ $t_i \overset{a}{\Rightarrow} t_i''$ ou $t_i' \overset{a}{\Rightarrow} t_i''$. Sans perte de généralité on peut supposer que l'on considère une infinité de dérivations à partir des t_i, c'est-à-dire qu'il existe un ensemble infini d'indices I, tel que $\forall i \in I$ $t_i \overset{a}{\Rightarrow} t_i''$ et $\forall i \in N-I$ $t_i' \overset{a}{\Rightarrow} t_i''$.

Apparemment $\{t_i''\}_{i \in I} \in S$ (par une transposition d'indices) et on a $\underset{i \in N}{\cap} t_i'' \ll \underset{i \in I}{\cap} t_i''$, ce qui implique par la proposition 7

$\underset{i \in N}{\cap} t_i'' \overset{W}{\approx} \underset{i \in I}{\cap} t_i''$ (1).

La séquence $\{t_i\}$ étant décroissante et $t_i \overset{a}{\Rightarrow} t_i''$ pour $i \in I$ on peut définir une séquence décroissante $\{t_i^*\}$ telle que $t_i^*=t_i''$ pour $i \in I$ et $t_i \overset{a}{\Rightarrow} t_i^*$ $\forall i \in N$. Ceci implique $\cap t_i \overset{a}{\Rightarrow} \cap t_i^*$ et également $\cap t_i^* \ll \underset{i \in I}{\cap} t_i''$. Par la proposition 7 et (1) on obtient $\cap t_i^* \overset{W}{\approx} \underset{i \in N}{\cap} t_i''$. Par conséquent toute a-dérivation de $\cap(t_i+t_i')$ est une a-dérivation de $\cap t_i+\cap t_i'$.

Evidemment, toute a-dérivation de $\cap t_i+\cap t_i'$ est une a-dérivation de $\cap(t_i+t_i')$. Ceci implique $\cap(t_i+t_i') \overset{W}{\approx} \cap t_i+\cap t_i'$. □

Les propositions 9 et 10 sont des propriétés de continuité des opérateurs $a \in A$ et $+$.

3.5 Solution d'équations récursives dans $CT^{\infty}_{\Omega}[\Sigma]$

L'objectif de ce paragraphe est de montrer que les équations récursives en $CT_{\Omega}[\Sigma]$ peuvent être interprétés comme des éléments de $CT^{\infty}_{\Omega}[\Sigma]$. Pour cela, on montre qu'un système d'équations de la forme

$$(S) \quad x_i = f_i(x_1,\ldots x_n),\ i=1..n$$

où les f_i sont des fonctionnelles sur un ensembles de variables construites à partir de Nil, + et $a\in A$, a une solution unique dans $(CT^{\infty}_{\Omega}[\Sigma]/\Omega)^n$.

Soit pour $i=1..n$ $z_i = \bigcap_{k\in N} z_{i,k}$ où $z_{i,0}=\Omega$ et $z_{i,k+1}=f_i(z_{1,k},\ldots z_{n,k})$ pour $i=1..n$.

Démontrons d'abord que $(z_i)_{1..n}$ est une solution du système (S).

On a pour $i=1..n$:

$f_i(\bigcap_{k\in N} z_{1,k},\ldots \bigcap_{k\in N} z_{n,k}) \ \Omega \ \bigcap_{k\in N} f_i(z_{1,k},\ldots z_{n,k})$ par les propositions 9 et 10.

$\bigcap_{k\in N} f_i(z_{1,k},\ldots z_{n,k}) \ \Omega \ \bigcap_{k\in N} z_{i,k+1} \ \Omega \ \bigcap_{k\geq 1} z_{i,k}$. La fonctionnelle f_i étant bien gardée, on a $\bigcap_{k\geq 1} z_{i,k} \ \Omega \ \bigcap_{k\in N} z_{i,k}$.

Evidemment, $(z_i = \bigcap_{k\in N} z_{i,k})_{1..n}$ est la plus grande solution de (S) dans $(CT^{\infty}_{\Omega}[\Sigma]/\Omega)^n$.

De plus cette solution est unique. Si $(z_i')_{1..n}$ est une solution, alors on a $z_i' \ll z_i$ pour $i=1..n$. Par la proposition 7 on obtient $z_i' \preceq z_i$ et par une remarque précédente $z_i' \ \Omega z_i$. On a donc la proposition suivante.

Proposition 11

Le système d'équations $(x_i=f_i(x_1,\ldots x_n))_{1..n}$ où les f_i sont des fonctionnelles bien gardées, a dans $(CT^{\infty}_{\Omega}[\Sigma]/\Omega)^n$ la solution unique $(z_i = \bigcap_{k\in N} f_i^k(\Omega,\ldots\Omega))_{1..n}$.

3.6 Interprétation de $CT[\Sigma,X]/\mathcal{E}$ dans $CT^{\infty}_{\Omega}[\Sigma]/\Omega$

L'objectif de ce paragraphe est de montrer que les termes de $CT[\Sigma,X]/\mathcal{E}$ peuvent être interprétés comme des termes de $CT^{\infty}_{\Omega}[\Sigma]/\Omega$.
Pour cela, on définit une fonction h: $CT[\Sigma,X] \rightarrow CT^{\infty}_{\Omega}[\Sigma]$ qui préserve les relations $\mathcal{E}$ et Ω, c'est-à-dire

$\forall t_1,t_2 \in CT[\Sigma,X]$ $t_1 \mathcal{E} t_2$ ssi $h(t_1)\Omega h(t_2)$.

Ceci permet d'interpréter les termes récursifs de $CT[\Sigma,X]$ comme des termes "infinis" de $CT^{\infty}_{\Omega}[\Sigma]$.

Les termes de $CT[\Sigma,X]/\mathcal{E}$ sans occurrence de rec sont également des termes de $CT^{\infty}_{\Omega}[\Sigma]/\Omega$ étant donné que $\mathcal{E}$ et Ω coïncident sur ces termes. Le problème de l'interprétation se pose donc pour les termes avec occurrences de rec dont la forme la plus générale est

$$t = \mathrm{rec}x_1.f_1(x_1, \mathrm{rec}x_2.f_2(x_1,x_2,\mathrm{rec}x_3.f_3(\ ),\ldots\mathrm{rec}x_n.f_n(\ )\ ),\quad (1)$$
$$\mathrm{rec}x_3.f_3(x_1,\mathrm{rec}x_2.f_2(\ ),x_3,\ldots\mathrm{rec}x_n.f_n(\ )\ ),$$
$$\mathrm{rec}x_n.f_n(x_1,\mathrm{rec}x_2.f_2(\ ),\ldots x_n)\ ).$$

où on peut supposer sans perte de généralité que les f_i sont des fonctionnelles construites à partir de Nil, + et $a \in A$.

Définition 7

On définit une fonction h: $CT[\Sigma,X] \rightarrow CT^{\infty}_{\Omega}[\Sigma]$ par

- h(Nil) = Nil
- Pour les termes récursifs t de la forme (1) on pose $h(t) = z_1$, où z_1 solution de $\{z_i = f_i(z_1,\ldots z_n)\}_{1..n}$
- $h(t_1+t_2) = h(t_1)+h(t_2)$
- h(at) = ah(t).

Proposition 12

Pour tout $t \in CT[\Sigma,X]$ $h(t) \mathcal{S} t$.

Preuve: Rappelons que sur $CT[\Sigma,X]$ $\mathcal{S}$ est égale à $\mathcal{W}$, l'équivalence faible. On prouve la proposition par induction sur la structure des termes de $CT[\Sigma,X]$.

- $h(Nil) \mathcal{S} Nil$.
- Si t_i est un terme récursif de la forme (1), c'est-à-dire
  $t_i = rec x_i.f_i(x_1,t_2,\ldots t_n)$, où
  $t_i(x_1,\ldots x_n) = rec x_i.f_i(x_1,t_2,\ldots x_i,\ldots)$ pour $i=2..n$, on a
  $t_i \mathcal{S} t_i[t_i(x_1,\ldots x_n)/x_i] \mathcal{W}_0 z_{1,1}$ de 3.5 parce que t_i est bien gardé, c'est-à-dire $\Omega-p(z_{1,1})=\Omega-p(t_i(\Omega,\ldots\Omega))>0$.
  Posons $t_{1,0}= t_i$ et $t_{1,k+1}= t_{1,k}[t_i(x_1,\ldots x_n)/x_i]$
  On obtient:
  $t_i \mathcal{S} t_{1,2} \mathcal{W}_1 z_{1,2}$ (parce que $\Omega-p(z_{1,2})>1$)
  et par substitution successive
  $t_i \mathcal{S} t_{1,k+1} \mathcal{W}_k z_{1,k+1}$, ce qui implique
  $t_i \mathcal{W}_k z_{1,k+1}$ $\forall k \in N$ et par la proposition 6 on a
  $z_{1,k+1} \mathcal{W}_k \Omega z_{1,j}$, ce qui implique par la définition de $\mathcal{W}$
  $t_i \mathcal{W} \Omega z_{1,j} = h(t_i)$.
- Si $h(t_i) \mathcal{S} t_i$ pour $i=1,2$ alors on a $h(t_1+t_2) = h(t_1)+h(t_2) \mathcal{S} t_1+t_2$.
- si $h(t) \mathcal{S} t$, alors on a $h(at) = ah(t) \mathcal{S} at$. □

Proposition 13

Pour $t_1,t_2 \in CT[\Sigma,X]$ on a
$t_1 \mathcal{S} t_2$ ssi $h(t_1) \mathcal{S} h(t_2)$.

Preuve: Par la proposition précédente on a $h(t_1) \mathcal{W} t_1$ et $h(t_2) \mathcal{W} t_2$. On obtient le résultat par le fait que $\mathcal{S}=\mathcal{W}$ sur $CT[\Sigma,X]$. □

IV.4 Caractérisation de la congruence observationnelle sur les termes infinis de CCS

4.1 Caractérisation modale de $CT_\Omega[\Sigma]/^\Omega$

Le but de ce paragraphe est de définir une fonction
$\| \ \| : CT_\Omega[\Sigma] \rightarrow FF_A$ telle que

$t' \models \|t\|$ ssi $t' \in prol(t)$ $\forall t' \in CT_\Omega[\Sigma]$.

ce qui implique

$t \,^\Omega t'$ ssi $\|t\| = \|t'\|$.

On définit la fonction $\| \ \|$ par :

- $\|Nil\| = [\bot]$
- $\|\Omega\| = T$
- $\|at\| = \langle a \wedge E \| t \| \rangle [a \wedge E \| t \|]$ pour $a \in A - \{\tau\}$
- $\|t_1 + t_2\| = A_1 \wedge A_2 \wedge [\|\hat{t}_1\| \vee \|\hat{t}_2\|]$ où
  $\|t_i\| = A_i \wedge [\|\hat{t}_i\|]$ où
  A_i une conjonction de termes $\langle f \rangle$ ou $A_i = T$.

On définit les opérateurs ⊛ et ⊕ sur l'image de $\| \ \|$ par

- $⊛ \|t\| = \|at\|$
- $\|t_1\| ⊕ \|t_2\| = \|t_1 + t_2\|$.

F_Ω est défini comme l'ensemble des formules générées à partir de $[\bot]$ et T par application de ⊛ et ⊕, c'est-à-dire F_Ω est l'image de $\| \ \|$.

Propriété 1

Pour $f_1, f_1', f_2, f_2' \in F_\Omega$ on a:

$f_1 \supset f_1'$ implique $⊛ f_1 \supset ⊛ f_1'$

$f_1 \supset f_1'$ et $f_2 \supset f_2'$ implique $f_1 ⊕ f_2 \supset f_1' ⊕ f_2'$ □

Lemme 1

Soit f une formule de F_Ω représentant une union de classes de $\mathcal{S}$. Alors, pour tout $t' \in T[\Sigma]$ tel que $\exists t'' \in CT[\Sigma]$ $t' \mathcal{S}_\tau t''$ ou $t' \mathcal{S} t''$ on a

$$t' \models E(f) \text{ ssi } t' \models \bigvee_{|t| \supset f} E|t|.$$

Preuve: Du fait que E est monotone on a $\bigvee_{|t| \supset f} E|t| \supset E(f)$.
Si $t' \models E(f)$, alors $\exists t''$ $t' \xrightarrow{\Sigma} t''$ et $t'' \models f$, c'est-à-dire $t'' \models \bigvee_{|t| \supset f} |t|$ (1). Donc $t' \mathcal{S} t''$ ou $t' \mathcal{S}_\tau t''$ (cf. [MI] chap.9). Etant donné (1) $\exists |t|$ ($|t| \supset f$ et $t'' \models |t|$). Donc $t' \models E|t|$. Alors, on déduit $t' \models \bigvee_{|t| \supset f} E|t|$. □

Proposition 1

Soit f une formule de F_Ω qui représente une union de classes de $\mathcal{S}$, c'est-à-dire $f = \bigvee_{|t| \supset f} |t|$. Alors

$$\textcircled{@} f = \bigvee_{F \in H} \bigwedge_{|t| \in F} at|$$ où $H = \{\{|t_1|, \ldots, |t_n|\} \mid |t_j| \supset f \text{ pour } i = 1..n\}$

Preuve: Etant donné $f = \bigvee_{|t| \supset f} |t|$ est une union de classes de $\mathcal{S}$ de termes appartenant à $CT[\Sigma]$, pour tout t satisfaisant f existe un terme congru à t sans occurrence de τ. Ceci implique $|\hat{t}| \setminus a = \bot$ et par conséquent $|at| = \langle a \wedge E|t| \rangle [a \wedge E|t|] = \textcircled{@} |t|$ et également $|at + at'| = |at| \oplus |at'|$. Donc $|t| \supset f$ implique $|at| \supset \textcircled{@} f$ et également $|t| \supset f$ et $|t'| \supset f$ implique $|at + at'| \supset \textcircled{@} f \oplus \textcircled{@} f$. Du fait que $\textcircled{@} f \oplus \textcircled{@} f = \textcircled{@} f$, on obtient finalement

$$\bigvee_{F \in H} \bigwedge_{|t| \in F} at| \supset f.$$

Pour l'inverse on montre $t' \models \textcircled{@} f$ implique $t' \models |\sum_{i \in I} at_i|$, où I est fini et $|t_i| \supset f$. $t' \models \textcircled{@} f$ est équivalent à $t' \models \langle a \wedge E(f) \rangle [a \wedge E(f)]$ ce qui implique $t' = \sum_{i \in I} at_i$ où I est fini et $t_i \models E(f)$. $t' \in CT[\Sigma]$ implique que pour tout $i \in I$ $t_i \in CT[\Sigma]$ ou $t_i \mathcal{S}_\tau t_i'$ pour un $t_i' \in CT[\Sigma]$. Par le lemme 1, on déduit que $t_i \models E(f)$ implique $t_i \models E|t_i'|$ pour un $|t_i'|$ tel que $|t_i'| \supset f$. Ceci implique $at_i \models |at_i'|$ et par conséquent

$$\sum_{i \in I} at_i \models |\sum_{i \in I} at_i'|.$$ □

Proposition 2

Soient f,f' des formules de F_Ω représentant des unions de classes, c'est-à-dire $f = \bigvee_{|t| \supset f} |t|$ et $f' = \bigvee_{|t| \supset f'} |t|$. Alors, on a

$$f \oplus f' = \bigvee_{|t| \supset f,\, |t'| \supset f'} |t+t'|.$$

Preuve: Remarquons que pour $t,t' \in CT[\Sigma]$ $|t| \oplus |t'| \equiv |t+t'|$. En effet, dans ce cas pour tous les sous-termes $a_i t_i$ on a
$|a_i t_i| = \langle a_i \wedge E|t_i| \rangle [a_i \wedge E|t_i|]$ ce qui implique
$[|a_i \hat{t}_i|] \supset [|a_i \hat{t}_i|]$ ssi $[|a_i \hat{t}_i|] \equiv [|a_i \hat{t}_i|]$.
Ceci implique que pour une somme $|t+t'|$ de termes de $CT[\Sigma]$ le prédicat c(at,bt') est toujours faux ce qui implique $|t+t'| \equiv |t| \oplus |t'|$.
De la monotonie de $\oplus$ on déduit $|t| \supset f$ et $|t'| \supset f'$ implique $|t| \oplus |t'| \supset f \oplus f'$, équivalent à $|t+t'| \supset f \oplus f'$. On a donc $\bigvee_{|t| \supset f,\, |t'| \supset f'} |t+t'| \supset f \oplus f'$.

Dans le sens inverse on montre $t'' \models f \oplus f'$ implique $t'' \models |t+t'|$ pour t,t' tels que $|t| \supset f$ et $|t'| \supset f'$.

Si $f = [\bot]$ rien est à prouver.

Si $f = T \equiv \bigvee_{t \in CT[\Sigma]} |t|$, $f' = A \wedge [\hat{f}]$ alors $f \oplus f' = A \equiv \bigvee_{|t'| \supset f, t \in CT[\Sigma]} |t+t'|$.

Sinon $f = \bigwedge_{i \in I} \langle x_i \rangle [\hat{f}]$, $f' = \bigwedge_{i \in J} \langle y_i \rangle [\hat{f}']$ où x_i, y_i sont de la forme $\langle a \wedge E(f'') \rangle$.
Ceci implique $f \oplus f' = \bigwedge_{i \in I} \langle x_i \rangle \bigwedge_{i \in J} \langle y_i \rangle [\hat{f} \vee \hat{f}']$. $t'' \models f \oplus f'$ implique que t'' est de la forme $t'' = \sum_{i \in K} a_i t_i$ tel que

- $\forall i \in I \; \exists j \in K \; a_j t_j \models \langle x_i \rangle$ (α) et $a_j t_j \models [\hat{f}]$ (α')
- $\forall i \in J \; \exists j \in K \; a_j t_j \models \langle y_i \rangle$ (β) et $a_j t_j \models [\hat{f}']$ (β')
- $\forall k \in K \; a_j t_j \models [\hat{f}]$ (α') ou $a_j t_j \models [\hat{f}']$ (β')

Soit K',K'' les sous-ensembles d'indices de K pour lesquels (α') respectivement (β') sont vérifiés. On pose,
$t = \sum_{i \in K'} a_i t_i$ et $t' = \sum_{i \in K''} a_i t_i$. Du fait que $K = K' \cup K''$ on a $t'' \not\sim t+t'$, donc $|t''| \equiv |t+t'|$ et par $t \models f$ et $t' \models f'$ on obtient le résultat. □

Théorème 1

Pour tout terme $t \in CT_\Omega[\Sigma]$ on a,

(C) $t' \models \|t\|$ ssi $t' \in prol(t)$.

Preuve: On fait une induction sur la structure de F_Ω.

- $t' \models T$ ssi $t' \in prol(\Omega) = CT[\Sigma]$ et
- $t' \models [\bot]$ ssi $t' \in prol(Nil) = \{t' \mathcal{S} Nil\}$.
- Soit $\|t\|$ tel que $t' \models \|t\|$ ssi $t' \in prol(t)$. Alors $t' \models @\|t\|$ ssi $\exists t_j$ $t' \mathcal{S} \Sigma at_j$ et $t_j \models \|t\|$ par la proposition 1. Par l'hypothèse d'induction $t_j \models \|t\|$ ssi $t_j \in prol(t)$ pour tout i. ssi $t' = \Sigma at_j \in prol(at)$ par la propriété 4 de 3.2. □
- Soit $\|t_1\|.\|t_2\|$ tels que $t_j' \models \|t_j\|$ ssi $t_j' \in prol(t_j)$ pour $i=1.2$. Alors $t' \models \|t_1\| \oplus \|t_2\|$ ssi $t' \mathcal{S} t_1' + t_2'$ où $t_j' \models \|t_j\|$ par la proposition 2. Par l'hypothèse d'induction $t_j' \models \|t_j\|$ ssi $t_j' \in prol(t_j)$ pour $i=1.2$. et $t_j' \in prol(t_j)$ pour $i=1.2$. ssi $t' \in prol(t_1 + t_2)$ par la propriété 5 de 3.2. □

4.2 Proposition d'une logique pour la preuve des processus contrôlables de CCS.

4.2.1 Définition du langage de base F_B

Les formules de F_Ω caractérisent des unions de classes de $\mathcal{S}$ particulières de $CT[\Sigma]$, celles qui sont des prolongements de termes de $CT_\Omega[\Sigma]$. On veut enrichir le langage F_Ω pour obtenir une logique, interprétée dans LTAR, en étendant les opérateurs $\textcircled{a}$ et $\oplus$ sur des disjonctions et conjonctions de formules de F_Ω.

On considère le langage de formules F_B défini par

- $\bot, T, [\bot] \in F_B$
- Si $f, f' \in F_B$ alors $f \vee f', f \wedge f' \in F_B$
- Si $f \in F_B$, b un sous-ensemble de A, i.e. un élément du treillis $B=(2^A, \cup, \cap, \emptyset, A)$ alors $\textcircled{b} f \in F_B$.
- Si $f, f' \in F_B$ alors $f \oplus f' \in F_B$.

La sémantique des opérateurs est définie de la manière suivante:

- L'interprétation de $\bot, T, [\bot], f \vee f', f \wedge f'$ est l'interprétation en LTAR (2.1).
- $t \models \textcircled{b} f$ ssi $\exists t_i \in CT[\Sigma], a_i \in A$ $t \stackrel{\mathcal{S}}{\sim} \Sigma a_i t_i$ et $a_i \in b$ et $t_i \models f$.
- $t \models f \oplus f'$ ssi $\exists t', t'' \in CT[\Sigma]$, $t \stackrel{\mathcal{S}}{\sim} t' + t''$ et $t' \models f$ et $t'' \models f'$.

D'abord nous donnons quelques propriétés concernant $\bot$.

Propriété 2

1) $\textcircled{b} \bot \equiv \bot$

2) $\bot \oplus f \equiv \bot$

3) $\emptyset f \equiv \bot$

Preuve: évidente. □

Ensuite, on montre que les formules de F_B représentent des disjonctions de formules de F_Ω et par conséquent des unions de

classes de $\mathcal{S}$ sur $CT[\Sigma]$.

Propriété 3

1) $f \oplus (f' \vee f'') \equiv f \oplus f' \vee f \oplus f''$.

2) $\textcircled{\scriptsize \cup}(f \vee f') \equiv \textcircled{\scriptsize \cup} f \vee \textcircled{\scriptsize \cup} f' \vee \textcircled{\scriptsize \cup} f \oplus \textcircled{\scriptsize \cup} f'$.

3) $\textcircled{a \cup b} f \equiv \textcircled{a} f \vee \textcircled{b} f \vee \textcircled{a} f \oplus \textcircled{b} f$.

Preuve:

1) $t \models f \oplus (f' \vee f'')$

ssi $]t', t''$ $t \overset{\mathcal{S}}{\sim} t' + t''$ et $t' \models f$ et $t'' \models f' \vee f''$

ssi $]t', t''(t \overset{\mathcal{S}}{\sim} t' + t''$ et $t' \models f$ et $t'' \models f')$ ou $]t', t''(t \overset{\mathcal{S}}{\sim} t' + t''$ et $t' \models f$ et $t'' \models f'')$

ssi $t \models f \oplus f' \vee f \oplus f''$.

2) $t \models \textcircled{\scriptsize \cup}(f \vee f')$

ssi t de la forme $t \overset{\mathcal{S}}{\sim} \sum_{i \in I} a_i t_i$ pour $a_i \in b$ et $t_i \models f \vee f'$ pour $t_i \in CT[\Sigma]$

ssi $t \overset{\mathcal{S}}{\sim} \sum_{i \in I} a_i t_i$ $a_i \in b$ et $(\forall i \in I$ $t_i \models f$ ou $\forall i \in I$ $t_i \models f'$ ou $]I', I'' \neq \emptyset$ $I' \cup I'' = I$
et $\forall i \in I'$ $t_i \models f$ et $\forall i \in I''$ $t_i \models f')$

ssi $t \models \textcircled{\scriptsize \cup} f$ ou $t \models \textcircled{\scriptsize \cup} f'$ ou $\sum_{i \in I'} a_i t_i \models \textcircled{\scriptsize \cup} f$ et $\sum_{i \in I''} a_i t_i \models \textcircled{\scriptsize \cup} f$

ssi $t \models \textcircled{\scriptsize \cup} f \vee \textcircled{\scriptsize \cup} f' \vee \textcircled{\scriptsize \cup} f \oplus \textcircled{\scriptsize \cup} f'$.

3) analogue à la preuve de 2). □

Proposition 3

Pour toute formule f de F_B on a $f \equiv \bot$ ou f est une disjonction de formules de F_Ω.

Preuve: Remarquons que $\bot$ caractérise l'ensemble $\emptyset$. L'application d'un opérateur de F_B à $\bot$ (s'il est unaire) ou à $\bot$ et f (s'il est binaire) donne un résultat $\bot$ ou f (propriété 2), donc $\bot$ ou une disjonction de formules de F_Ω si f est déjà une telle formule.

- $[\bot]$ et $\top$ sont des formules de F_Ω.
- Si f, f' représentent des disjonctions de formules de F_Ω, donc des unions de classes de $\mathcal{S}$ sur $CT[\Sigma]$, alors $f \vee f'$ et $f \wedge f'$ représentent des unions de classes de $\mathcal{S}$.

- Si $f=\bigvee f_i$ pour $f_i \in F_\Omega$, alors

$$\Diamond f = \begin{cases} \Diamond_\emptyset f = \bot & \text{si } b=\emptyset \\ (\Diamond_{\{a_1\}} \cup \ldots \cup \Diamond_{\{a_n\}}) f & \text{si } b=\{a_1,\ldots a_n\} \subseteq A. \end{cases}$$

Dans le deuxième cas on peut à partir des propriétés 3 (2) et (3) décomposer $\Diamond f$ complètement en une disjonction de formules de la forme $\Diamond_{1_1} f \oplus \ldots \oplus \Diamond_{n_n} f$ qui sont des formules de F_Ω.

- Si $f=\bigvee_{i\in I} f_i$, $f'=\bigvee_{i\in I'} f_i'$ où $f_i, f_i' \in F_\Omega$, alors

$f \oplus f' = (\bigvee_{i\in I} f_i) \oplus (\bigvee_{i\in I'} f_i') = \bigvee_{(i,j)\in I\times I'} f_i \oplus f_j'$ d'après la propriété 3 (1), une disjonction de formules de F_Ω. □

Par la suite on montre que $\Diamond f$ et $f \oplus f'$ admettent une représentation par des formules de FF_A, similaire à celle donnée en 4.1 pour les formules de F_Ω.

Pour ceci on introduit un opérateur E tel que pour $f \in F_B$

$t \models E(f)$ ssi $t \in CT[\Sigma]$ et $t \models f$ ou $\exists t' \in CT[\Sigma]$ $t \mathcal{S}_\tau t'$ et $t' \models f$.

Propriété 4

1) $E(f \vee f') \equiv E(f) \vee E(f')$

2) $E(f \wedge f') \equiv E(f) \wedge E(f')$

Preuve: Remarquons que pour $t', t'' \in CT[\Sigma]$ $t \mathcal{S}_\tau t'$ et $t \mathcal{S}_\tau t''$ on a $t' \mathcal{S} t''$ et du fait que les formules $f \in F_\Omega$ représentent des unions de classes de $\mathcal{S}$, on a $t' \models f$ ssi $t'' \models f$.

1) $t \models E(f \vee f')$

ssi $t \in CT[\Sigma]$ et $t \models f$ ou $t \models f'$ ou $\exists t' \in CT[\Sigma]$ $t \mathcal{S}_\tau t'$ et $t' \models f$ ou $t' \models f'$

ssi $t \models E(f)$ ou $t \models E(f')$.

2) $t \models E(f) \wedge E(f')$

ssi ($t \in CT[\Sigma]$ et $t \models f$ ou $\exists t' \in CT[\Sigma]$ $t \mathcal{S}_\tau t'$ et $t' \models f$) et
($t \in CT[\Sigma]$ et $t \models f$ ou $\exists t'' \in CT[\Sigma]$ $t \mathcal{S}_\tau t''$ et $t'' \models f'$)

ssi $t \in CT[\Sigma]$ et $t \models f \wedge f'$ ou $\exists t' \in CT[\Sigma]$ $t \mathcal{S}_\tau t'$ et $t' \models f \wedge f'$ par la remarque précédente

ssi $t \models E(f \wedge f')$. □

Pour une formule $f \in F_B$. $E(f)$ est donc exprimable en termes de formules de LTAR. plus exactement si $f=\forall f_i$ pour $f_i \in F_\Omega$. alors $E(f)=\forall E(f_i)$.

On définit également pour $b \in B$ une formule $b' \in$ LTAR par

- $\emptyset'=\bot$
- $A'=\top$
- $a'=a$ pour $a \in A$
- $(b_1 \cup b_2)'=b_1' \vee b_2'$
- $(b_1 \cap b_2)'=b_1' \wedge b_2'$

Par la suite on ne distingue plus b et b'. Par ces définitions on obtient.

Proposition 4

1) $\oplus f = \langle b \wedge E(f) \rangle [b \wedge E(f)]$.

2) Pour $g= \bigwedge_{i \in I} \langle f_i \rangle [\hat{g}]$ et $g'= \bigwedge_{i \in J} \langle f_i' \rangle [\hat{g}']$ ou $g'=\top$ on a

$$g \oplus g' = \begin{cases} \bigwedge_{i \in I} \langle f_i \wedge \hat{g} \rangle & \text{si } g'=\top \\ \bigwedge_{i \in I} \langle f_i \wedge \hat{g} \rangle \bigwedge_{i \in J} \langle f_i' \wedge \hat{g}' \rangle [\hat{g} \vee \hat{g}'] & \text{sinon} \end{cases}$$

Plus particulièrement, on a

$$\bigoplus_{i \in I} \textcircled{b} f_i = \bigwedge_{i \in I} \langle b_i \wedge E(f_i) \rangle [\bigvee_{i \in I} b_i \wedge E(f_i)]$$

Preuve:

1) $t \models \oplus f$

ssi t a la forme $t=\Sigma a_i t_i'$ $\}t_i \in CT[\Sigma]$ $a_i \in b$ et $(t_i' \mathcal{S} t_i$ ou $t_i' \mathcal{S}_T t_i)$ et $t_i \models f$

ssi $t=\Sigma a_i t_i'$ $a_i \in b$ et $t_i' \models E(f)$

ssi $t \models \langle b \wedge E(f) \rangle [b \wedge E(f)]$.

Evidemment $t \models \langle b \rangle$ ssi $\}t'$ $\}a \in A$ $t \xrightarrow{a} t'$ et $a \in b$.

2) Pour prouver (2) on montre pour $\langle f_i \rangle [f_i'] \in F_\Omega$

$$(2') \quad \bigoplus_{i \in I} \langle f_i \rangle [f_i'] = \bigwedge_{i \in I} \langle f_i \wedge f_i' \rangle [\bigvee_{i \in I} f_i'].$$

d'où on peut obtenir facilement (2) dans le cas où $g' \neq \top$.

Remarquons d'abord que

- $\langle f_i \rangle [f_i'] = \langle f_i \wedge f_i' \rangle [f_i']$ (propriété 1 (T5) de 2.1) et
- $t \mathcal{S} \Sigma t_i$ où $t_i \in CT[\Sigma]$ implique qu'il existe $t_i' \in CT[\Sigma]$ $t \sim \Sigma t_i'$ et

$t_j' \mathcal{S} t_j$ (propriété 2 de 3.1). On sait également que pour toute formule $f \in FF_A$ $t \sim t'$ et $t \models f$ ssi $t' \models f$.

- Pour $g = \langle f \rangle [f']$ et $g' = T$, on obtient

  $t \models g \oplus g'$ ssi $t \mathcal{S} t' + t''$ $t' \models \langle f \wedge f' \rangle [f']$ et $t'' \models T$.

  $t \models \langle f \wedge f' \rangle$ ssi $t \mathcal{S} a t_1 + t_2$ et $a t_1 \models \langle f \wedge f' \rangle$.

  ssi $t \mathcal{S} a t_1 + t_2$ et $a t_1 \models \langle f \wedge f' \rangle [f']$.

  Evidemment, $t_2 \models T$.

  Ceci implique $t \models g \oplus g'$ ssi $t \models \langle f \wedge f' \rangle$.

Maintenant, on montre (2')

$t \models \bigoplus_{i \in I} \langle f_i \rangle [f_i']$

ssi $t \mathcal{S} \sum_{i \in I} t_i'$ et $t_i' \models \langle f_i \wedge f_i' \rangle [f_i']$

ssi $\exists t_i'$ $t \sim \sum_{i \in I} t_i'$ et $t_i' \models \langle f_i \wedge f_i' \rangle [f_i']$ par la remarque précédente et le fait que les $\langle f_i \rangle [f_i']$ représentent des unions de classes de $\mathcal{S}$.

ssi $\exists t_i'$ $t \sim \sum_{i \in I} t_i'$ et $\sum_{i \in I} t_i' \models \bigwedge_{i \in I} \langle f_i \wedge f_i' \rangle [\bigvee_{i \in I} f_i']$ (propriété 2 de 2.2)

ssi $t \models \bigwedge_{i \in I} \langle f_i \wedge f_i' \rangle [\bigvee_{i \in I} f_i']$ par la remarque précédente. □

On peut obtenir facilement les propriétés suivantes.

Propriété 5

1) $\textcircled{a \cap b} f = \textcircled{a} f \wedge \textcircled{b} f$

2) $\textcircled{a}(f \wedge f') = \textcircled{a} f \wedge \textcircled{a} f'$

Preuve:

1) $\textcircled{a \cap b} f = \langle a \wedge b \wedge E(f) \rangle [a \wedge b \wedge E(f)]$

$= \langle a \wedge E(f) \rangle \langle b \wedge E(f) \rangle [a \wedge E(f)] [b \wedge E(f)]$ (propriété 1 (T5) et (T2) de 2.1)

$= \textcircled{a} f \wedge \textcircled{b} f$.

2) $\textcircled{a}(f \wedge f') = \langle b \wedge E(f \wedge f') \rangle [b \wedge E(f \wedge f')]$

$= \langle b \wedge E(f) \wedge b \wedge E(f') \rangle [b \wedge E(f) \wedge b \wedge E(f')]$ (propriété 4 (3))

$= \textcircled{a} f \wedge \textcircled{a} f'$ (comme dans la preuve de (1)). □

Propriété 6

Soient $f.f'.f'' \in F_\Omega$ tels que $g \oplus g = g$ pour $g = f.f'.f''$. Alors
$f \supset f'$ et $f' \supset f''$ implique $f \oplus f' \oplus f'' = f \oplus f''$.

Preuve: Remarquons que $g \oplus g = g$ ssi $\forall t.t' \in CT[\Sigma]$ ($t \models g$ et $t' \models g$ implique $t+t' \models g$). Ceci est le cas ssi g peut être mis sous la forme $g = \Lambda \langle g_i \rangle [\hat{g}]$ ou $g = [\hat{g}]$.
Soient $f.f'.f'' \in F_\Omega$ tels que $g \oplus g = g$ pour $g = f.f'.f''$ et $f \supset f'$ et $f' \supset f''$.
$f \supset f'$ implique $f \oplus f \supset f \oplus f'$ donc $f \supset f \oplus f'$
implique $f \oplus f'' \supset f \oplus f' \oplus f''$.
$f' \supset f''$ implique $f' \oplus f'' \supset f'' \oplus f''$ donc $f' \oplus f'' \supset f''$
implique $f \oplus f' \oplus f'' \supset f \oplus f''$. □

Proposition 5

$$\left(\bigoplus_{i \in I} b_i f_i\right) \wedge \left(\bigoplus_{i \in J} c_i g_i\right) = \bigoplus_{i \in I} \left(b_i f_i \wedge \left(\bigvee_{i \in J} c_i g_i\right)\right) \oplus \bigoplus_{i \in J} \left(c_i g_i \wedge \left(\bigvee_{i \in I} b_i f_i\right)\right)$$

Preuve:

$$\left(\bigoplus_{i \in I} b_i f_i\right) \wedge \left(\bigoplus_{i \in J} c_i g_i\right) =$$

$$\bigwedge_{i \in I} \langle b_i E(f_i) \rangle \bigwedge_{i \in J} \langle c_i E(g_i) \rangle \left[\bigvee_{i \in I} b_i E(f_i) \wedge \bigvee_{i \in J} c_i E(g_i)\right] =$$

$$\bigwedge_{i \in I} \langle b_i E(f_i) \wedge \left(\bigvee_{i \in J} c_i E(g_i)\right) \rangle \bigwedge_{i \in J} \langle c_i E(g_i) \wedge \left(\bigvee_{i \in I} b_i E(f_i)\right) \rangle \left[\bigvee_{(i,j) \in I \times J} b_i c_j E(f_i g_j)\right] =$$

$$\bigwedge_{i \in I} \left(\bigvee_{j \in J} \langle b_i c_j E(f_i g_j) \rangle\right) \wedge \bigwedge_{j \in J} \left(\bigvee_{i \in I} \langle c_j b_i E(g_j f_i) \rangle\right) [\ldots] =$$

$$\bigvee_{k \in J^n} \left(\bigwedge_{i \in I} \langle c_{j_k} b_i E(g_{j_k} f_i) \rangle\right) \wedge \bigvee_{j \in I^m} \left(\bigwedge_{i \in J} \langle c_j b_{i_j} E(g_j f_{i_j}) \rangle\right) [\ldots] =$$

où $n = |I|$ et $m = |J|$

$$\bigvee_{k \in J^n} \bigvee_{j \in I^m} \left(\bigwedge_{i \in I} \langle c_{j_k} b_i E(g_{j_k} f_i) \rangle \wedge \bigwedge_{i \in J} \langle c_i b_{i_j} E(g_j f_{i_j}) \rangle\right) [\ldots] =$$

$$\bigvee_{k \in J^n} \bigvee_{j \in I^m} \left(\bigoplus_{i \in I} (c_{j_k} b_i)(g_{j_k} \wedge f_i) \oplus \bigoplus_{i \in J} (c_i b_{i_j})(g_i \wedge f_{i_j})\right) =$$

$$\bigoplus_{i \in I} \left(b_i f_i \wedge \left(\bigvee_{i \in J} c_i g_i\right)\right) \oplus \bigoplus_{i \in J} \left(c_i g_i \wedge \left(\bigvee_{i \in I} b_i f_i\right)\right) \quad \square$$

Remarquons, que d'après cette proposition le terme

$(\bigoplus_{i\in I}(b_i)f_i)\wedge(\bigoplus_{i\in J}(c_i)g_i)$ peut se transformer dans un terme équivalent où la conjonction est "repoussée" à un niveau inférieur. Par application de cette proposition et les règles

- $\bigoplus f\wedge\bigoplus g \equiv (a\cap b)(f\wedge g)$
- $(\bigvee_{i\in I} f_i)\wedge g \equiv \bigvee_{i\in I}(f_i\wedge g)$
- $f\wedge[\bot] \equiv \begin{cases}[\bot] \text{ si } f\equiv[\bot] \text{ ou } f\equiv T\\ \bot \text{ sinon}\end{cases}$

on peut obtenir à partir d'une formule f donnée une formule f' équivalente sans occurrence de conjonction.

Dans l'exemple suivant, on illustre cette idée pour le cas où $I=J=\{1,2\}$.

Exemple 1

on note $\alpha:=(a_1)f_1$, $\alpha':=(a_2)f_2$ et $\beta:=(b_1)g_1$, $\beta':=(b_2)g_2$. On obtient

$$(\alpha\wedge\alpha')\oplus(\beta\wedge\beta') \equiv \alpha\wedge(\beta\vee\beta')\oplus\alpha'\wedge(\beta\vee\beta') \oplus \beta\wedge(\alpha\vee\alpha')\oplus\beta'\wedge(\alpha\vee\alpha')$$
$$\equiv \alpha\beta\oplus\alpha'\beta' \vee \alpha\beta'\oplus\alpha'\beta \vee \alpha\beta\oplus\alpha'\beta'\oplus\alpha\beta' \vee \alpha\beta\oplus\alpha'\beta'\oplus\alpha'\beta$$
$$\vee \alpha\beta'\oplus\alpha'\beta\oplus\alpha\beta \vee \alpha\beta'\oplus\alpha'\beta\oplus\alpha'\beta' \vee \alpha\beta\oplus\alpha'\beta'\oplus\alpha\beta'\oplus\alpha'\beta$$

Les formules $\alpha\beta$ peuvent être mises sous la forme $\alpha\wedge\beta\equiv(a\cap b)(f\wedge g)$, et $f\wedge g$ peut être mis sous une forme où la proposition est de nouveau appliquable.

Les opérateurs $\bigoplus$ de F_B expriment apparemment l'atteignabilité inévitable sous une condition b par une transition observable. En F_B on peut également définir des opérateurs $\Diamond$ exprimant l'atteignabilité possible sous une condition b par une transition observable. Pour ceci on définit:

$\Diamond f := \bigoplus f\oplus T$.

On a les propriétés suivantes:

Propriétés 7

1) $t \models \Diamond f$ ssi $\exists t'$ $t \overset{b}{\Rightarrow} t'$ et $t' \models f$

2) $\Diamond f \equiv \langle b \wedge E(f) \rangle$

3) $\langle b \cup c \rangle f \equiv \Diamond f \vee \Diamond f$

4) $\Diamond(f \vee f') \equiv \Diamond f \vee \Diamond f'$

5) $\Diamond f \wedge \Diamond f' \equiv \Diamond f \otimes \Diamond f'$

Preuve: (1) et (2) sont des conséquences immédiates de la proposition 4. A partir de (1) et (2) on peut déduire facilement (3),(4) et (5), où pour (5) on utilise également la proposition 4. □

4.2.2 L'extension F_μ de F_B.

Comme nous avons déjà remarqué, les modalités ⓑ et ⟨b⟩ de F_B expriment l'atteignabilité inévitable et possible sous une condition b par <u>une</u> transition observable. Dans ce paragraphe nous étendons F_B par adjonction d'opérateurs de point fixe, ce qui permet d'introduire des modalités exprimant l'atteignabilité inévitable et possible sous une condition b par un nombre quelconque de transitions observables. Cette extension permet également de donner une "traduction" des processus infinis de $CT[\Sigma,X]$.

On définit un <u>langage de formules</u> par :

- $\bot, T, [\bot], x \in X$ sont des formules, où X est un ensemble de variables.
- Si f,f' sont des formules, alors aussi $f \vee f'$ et $f \wedge f'$.
- Si f,f' sont des formules, alors
  - pour $b \in B$ ⓑf est une formule,
  - $f \oplus f'$ est une formule,
  - $\mu x.f$ et $\sigma x.f$ sont des formules.

On représente par F_μ l'ensemble des formules fermées de ce langage.

L'<u>interprétation</u> des opérateurs est définie comme pour F_B et LTAR, mais sur l'ensemble de termes $CT[\Sigma,X]$.
On montre par la suite que les formules de F_μ représentent des unions de classes de $\mathcal{S}$ sur $CT[\Sigma,X]$.

<u>Proposition 6</u>

Toute fonctionnelle F de F_μ est V-continue, c'est-à-dire pour toute séquence $\{f_i\}$ sur F_μ, telle que $f_i \supset f_{i+1}$, on a

$$F\left(\bigvee_{i \in N} f_i\right) = \bigvee_{i \in N} F(f_i).$$

<u>Preuve</u>: Il suffit de montrer que les opérateurs de F_μ sont V-continus. Pour $\vee$ et $\wedge$ ceci est évident. Pour ⓑ et $\oplus$ on montre:

Soient $\{f_i\}$, $\{g_i\}$ des suites croissantes de formules de F_μ, c'est-à-dire $f_i \supset f_{i+1}$ et $g_i \supset g_{i+1}$ $\forall i \in N$.

- On montre $V\circledast f_i \equiv \circledast(Vf_i)$:
  $V\circledast f_i \supset \circledast(Vf_i)$ est une conséquence de la monotonie de $\circledast$.
  Soit $t=\Sigma a_i t_i$, donc $t_i \mathcal{S} t_i'$ ou $t_i \mathcal{S}\tau t_i'$ pour $t_i' \in CT[\Sigma,X]$ et $t \models \circledast(Vf_i)$. Ceci implique que $\forall i$ $a_i \in b$ et $t_i \models E(Vf_i)$. Ceci implique $\forall i$ $\exists j(i)$ $a_i t_i \models \circledast f_j$.
  Soit $n=\max\{j(i)\}$. Du fait que $\{f_i\}$ est croissante, on a $\forall i$ $a_i t_i \models \circledast f_n$, ce qui implique $\Sigma a_i t_i$ ce qui implique $t \models \circledast f_n$.
- On montre $(Vf_i)\oplus(Vg_i) \equiv V(f_i \oplus g_i)$:
  $V(f_i \oplus g_i) \supset (Vf_i)\oplus(Vg_i)$ est une conséquence de la monotonie de $\oplus$.
  Soit $t \in CT[\Sigma,X]$ et $t \models (Vf_i)\oplus(Vg_i)$. Ceci implique $\exists t_1, t_2$ $t \sim t_1 + t_2$ et $t_1 \models Vf_i$ et $t_2 \models Vg_i$, c'est-à-dire $\exists i,j$ $t_1 \models f_i$ et $t_2 \models g_j$. Pour $n=\max\{i,j\}$, on a donc $t_1 \models f_n$ et $t_2 \models g_n$, ce qui implique $t_1+t_2 \models f_n \oplus g_n$, donc $t \models Vf_i \oplus g_i$.
  □

Proposition 7

Toute fonctionnelle F de F_μ est Λ-continue, c'est-à-dire pour toute séquence $\{f_i\}$ sur F_μ, telle que $f_{i+1} \supset f_i$, on a

$$F(\Lambda f_i) \equiv \Lambda F(f_i)$$

Preuve: Comme pour la preuve précédente, il suffit de montrer que $\circledast$ et $\oplus$ sont Λ-continus.
Soient $\{f_i\}$, $\{g_i\}$ des suites décroissantes de formules de F_μ, c'est-à-dire $f_{i+1} \supset f_i$ et $g_{i+1} \supset g_i$ $\forall i \in N$.

- On montre $\circledast(\Lambda f_i) \equiv \Lambda \circledast f_i$
  $\circledast(\Lambda f_i) \supset \Lambda \circledast f_i$ est une conséquence de la monotonie de $\circledast$.
  Soit $t=\Sigma a_i t_i$, donc $t_i \mathcal{S} t_i'$ ou $t_i \mathcal{S}\tau t_i'$ pour $t_i' \in CT[\Sigma,X]$ et $t \models \Lambda \circledast f_i$. Ceci implique $t \models \circledast f_i$ $\forall i \in N$, c.a.d. $\forall i$ $a_i \in b$ et $t_i \models E(f_i)$. On obtient comme dans la preuve précédente $t_i' \models \Lambda f_i$. Ceci implique $\forall i$ $a_i t_i \models \circledast(\Lambda f_i)$, et par conséquent $t \models \circledast(\Lambda f_i)$.
- On montre $(\Lambda f_i)\oplus(\Lambda g_i) \equiv \Lambda(f_i \oplus g_i)$
  $(\Lambda f_i)\oplus(\Lambda g_i) \supset \Lambda(f_i \oplus g_i)$ est une conséquence de la monotonie de $\oplus$.
  Soit $t \in CT[\Sigma,X]$ et $t \models f_i \oplus g_i$ $\forall i \in N$. Parce que t est de degré fini, il

existe un nombre fini de $t_1'.t_2'$ et $t_1".t_2"$ tels que $t \sim t_1'+t_2'$ et $t \sim t_1"+t_2"$ et $(t_1' \not\mathcal{S} t_1"$ ou $t_2' \not\mathcal{S} t_2"$ pour i=1,2). Il existe donc $t_1'.t_2'$ tels que $t \sim t_1'+t_2'$ et $t_1' \models f_i$ et $t_2' \models g_i$ pour un nombre infini d'indices $i \in \mathbf{N}$. Du fait que $\{f_i\}$ et $\{g_i\}$ sont décroissantes, on a que $t_1' \models f_i$ et $t_2' \models g_i$ implique $t_1' \models \bigwedge_{j \leq i} f_j$ et $t_2' \models \bigwedge_{j \leq i} g_j$. Du fait que pour tout i il existe $j > i$ tel que $t_1' \models f_j$ et $t_2' \models g_j$ on obtient $t_1' \models \Lambda f_i$ et $t_2' \models \Lambda g_i$, donc $t \models (\Lambda f_i) \oplus (\Lambda g_i)$. □

A partir des propositions 6 et 7, on obtient facilement:

Proposition 8

a) Si F est une fonctionnelle de F_μ, on peut approcher les points fixes par:
   - $\mu x.F(x) = \bigvee f_i$ où $f_0 = \bot$ et $f_{i+1} = F(f_i)$
   - $\sigma x.F(x) = \Lambda f_i$ où $f_0 = T$ et $f_{i+1} = F(f_i)$

b) Les formules de F_μ représentent des unions de classes de $\mathcal{S}$ sur $CT[\Sigma,X]$.

A partir de la proposition 8, on voit que l'on peut définir une "traduction" pour les termes de la forme recx.t de $CT[\Sigma,X]$. D'après les résultats des paragraphes 3.5, 3.6 et 4.1 pour un t de la forme générale donnée en 3.6, on a $t' \mathcal{S}$ recx.t ssi $t' \models \bigwedge_{i \in N} \| z_{1,i} \|$, où les $z_{1,i}$ sont définis en 3.5. On obtient alors le résultat que $t' \mathcal{S}$ recx.t ssi $t' \models \sigma x.|t|$ pour une fonctionnelle $|t|$ de F_μ, obtenue par la fonction de traduction de formules suivante:

On définit une fonction $|\ |: CT[\Sigma,X] \to F_\mu$ par
   - $|Nil| = [\bot]$
   - $|x| = x$
   - $|at| = @|t|$
   - $|t+t'| = |t| \oplus |t'|$
   - $|recx.t| = \sigma x.|t|$

Des résultats des paragraphes 3.5, 3.6 et 4.1 on déduit facilement:

Proposition 9

$\forall t,t' \in CT[\Sigma,X]$ $t' \mathcal{S} t$ ssi $t' \models |t|$. □

Les opérateurs de point fixe permettent également de définir des modalités exprimant l'atteignabilité sous une condition b par un nombre quelconque de transitions observables. La proposition 8 garantit qu'elles peuvent être approchées par des calculs itératifs. On définit:

$POT(b,f) = \mu x.(f \vee \langle\!\langle b \rangle\!\rangle x)$
$INEV(b,f) = \mu x.(f \vee [\![b]\!] x)$
$ALL(b,f) = \sigma.(f \wedge [\![b]\!] x)$
$SOME(b,f) = \sigma.(f \wedge \langle\!\langle b \rangle\!\rangle x)$

De ces définitions on déduit que

- $t \models POT(b,f)$ ssi $\exists s \in A^* \exists t'$ $t \stackrel{s}{\Rightarrow} t'$ et $t' \models f$ et $s=\epsilon$ ou $s=a_1 \ldots a_n$ ou $a_i \in b$. c.à.d. ssi il est possible d'atteindre un t' satisfaisant f, en maintenant vrai la "condition b" jusqu'à ce que f soit satisfait.
- $t \models INEV(b,f)$ ssi pour tous les chemins maximaux partant de la racine de t, on passe par un t' satisfaisant f, après avoir seulement exécuté des transitions observables appartenant à b, c.à.d. ssi il est inévitable d'atteindre un t' satisfaisant f, en maintenant vrai la condition b jusqu'à ce que f soit satisfait.

Les modalités POT(b,f) et INEV(b,f) correspondent aux modalités POT(g,f) et INEV(g,f) de LTAC. Evidemment, les duaux ALL(b,f) et SOME(b,f) correspondent à ALL(g,f) respectivement SOME(g,f) de LTAC.
Nous avons donc défini une logique permettant de faire des preuves constructives qui contient des modalités de LTAC. Pour prouver $t \models f$, il suffit de prouver $\vdash |t| \supset f$ dans la logique.

IV.5 Conclusion

Le but de ce paragraphe était la proposition d'une logique pour la preuve constructive.

Nous avons commencé par donner une caractérisation modale de la congruence observationnelle sur les termes de $T[\Sigma]$, c.à.d. les termes finis de CCS. Des caractérisations similaires ont été proposées dans [HM],[BR] et [St] mais dans ces travaux on représente la classe de congruence d'un terme t par la conjonction (infinie) $|t| = \bigwedge_{t \models f} f$. De plus, $|t|$ n'est pas donné de façon constructive. Pour notre caractérisation, nous avons par contre défini des opérateurs $\circledast$ et $\oplus$ tels que $|at| = \circledast |t|$ et $|t+t'| = |t| \oplus |t'|$. Par la suite nous avons étendu les opérateurs $\circledast$ et $\oplus$ sur des formules représentant des unions de classes de congruence, plus particulièrements des "prolongements". Cette extension a été définie de telle sorte que $\circledast f$ et $f \oplus f'$ sont les formules les plus "strictes", déductibles de f et f', et représentant des unions de classes de $\mathcal{L}$. Ceci n'a été possible qu'en se restreignant aux processus contrôlables de CCS, c'est-à-dire les processus pour lesquels l'exécution d'une τ-transition mène à un processus observationnellement équivalent.

Pour ces processus nous avons pu définir une logique permettant d'exprimer les modalités de la logique LTAC et de faire des preuves constructives.

Signalons enfin, que l'on peut établir une bijection entre les termes de $CT_{\Omega}[\Sigma]$ et les formules construites à partir de $[\bot]$ et T avec les opérateurs $\circledast$ pour $a \in A - \{\tau\}$ et $\oplus$. On pourrait donc définir les opérateurs v et $\wedge$ directement sur les termes de $CT_{\Omega}[\Sigma]$ pour simplifier l'écriture des formules.

V CONCLUSION

L'objectif de ce travail a été l'étude des logiques temporelles en tant qu'outil de spécification et de preuve de programmes. Nous avons d'abord étudié les logiques du temps arborescent en tant qu'outil de spécification et comparé les logiques du temps arborescent aux logiques du temps linéaire en ce qui concerne leur puissance d'expression.

Nous avons montré que les logiques du temps arborescent, et en particulier la logique LTAC, sont des outils de spécification satisfaisants, en ce qui concerne leur puissance d'expression et leur décidabilité. Pour LTAC nous avons donné une procédure de décision, qui a été programmée dans le système CESAR [Sch] où LTAC est utilisé comme langage de spécification.

Un résultat important est, que les logiques du temps arborescent et les logiques du temps linéaire sont deux familles de logiques non comparables. Chacune des deux permet d'exprimer un certain type d'inévitabilité. Cependant, les logiques du temps linéaire, si on assimile un programme à un ensemble de traces, ne permettent pas d'exprimer la possibilité qu'un événement se produise, et par conséquent des propriétés essentielles du comportement, telles que l'absence de blocage partiel. De plus, ces logiques ne permettent pas de caractériser la structure

interne d'un système. En particulier, les relations d'équivalence qu'elles induisent sur les programmes ne préservent pas le choix non déterministe.

La deuxième partie de ce travail porte sur l'étude des logiques du temps arborescent en tant qu'outil de preuve. Nous avons proposé une logique permettant de faire des preuves constructives pour les processus contrôlables de CCS. Cette logique contient des opérateurs exprimant l'inévitabilité et la possibilité; elle peut être considérée comme une extension de LTAC.

Evidemment, il n'est pas envisageable d'utiliser la logique proposée en tant que telle comme outil de preuve de programmes, ses modèles étant des arbres de synchronisation. Il est nécessaire de l'enrichir en y ajoutant des variables propositionnelles portant sur les valeurs échangées lors d'une communication.

Enfin, l'approche suivie pour la définition de la logique proposée est à notre avis suffisamment générale pour qu'elle puisse être appliquée à d'autres algèbres utilisées comme langages de description de programmes non déterministes.

REFERENCES

[Ab] Abrahamson K. *Modal logic of concurrent non deterministic programs* Semantics of Concurrent Computation Proceedings, 79, LNCS 70.

[AS] Abrial J.R., Schuman S.A. *Non deterministic system verification* LNCS 70.

[Be] Ben-Ari M. *Complexity of proofs and models in programming logic* Thesis of Doctor of Philosophie, Tel Aviv 81.

[BMP1] Ben-Ari M., Manna Z., Pnueli A. *The temporal logic of branching time* ACM 8,81.

[BMP2] Ben-Ari M., Manna Z., Pnueli A. *The temporal logic of branching time* Acta Informatica 20, 83.

[BR] Brooks S.D., Rounds W.C. *Behavioural equivalence relations induced by programming logics* Proceedings 10th ICALP 83, LNCS 154.

[Ca] Mc Call S. *The strong future tense* Notre Dame Journal of formal logic 20(3), 79.

[Ch] Chellas B.F. *Modal logic, an introduction* Cambridge University Press,81.

[CE] Clarke E.M., Emerson E.A. *Design and synthesis of synchronisation skeletons using branching time logic* Harvard University TR-12-81.

[Em] Emerson E.A. *Alternative semantics for temporal logics* TCS Vol26, North Holland.

[EH1] Emerson E.A., Halpern J.Y. *Decision procedure and expressiveness in temporal logic of branching time* 14th annual ACM Symposium on Theory of Computing, 82.

[EH2] Emerson E.A., Halpern J.Y. *"Sometimes" and "not never" revisited: on branching versus linear time* POPL 83.

[GR] Golson W.G., Rounds W.C. *Connections between two theories of concurrence, metric spaces and synchronisation trees* University of Michigan CRL-TR-383.

[Gr] Graf S. *On Lamport's comparison between linear and branching time logic* IMAG RR348, à paraître dans RAIRO-Informatique Théorique

[GS] Graf S., Sifakis J. *A modal characterisation on finite terms of CCS* Rapport IMAG RR402, à paraître dans ICALP 84.

[HC] Hughes G., Cresswell M. *An introduction to modal logic* Methuen, 68.

[He] Hennessy M. *Axiomatising finite delay operators* Edinburgh, CSR-124-82.

[HM] Hennessy M., Milner R. *On observing non determinism and concurrency* 7th ICALP,80 LNCS 85.

[HS] Hennessy M., Stirling C. *The power or future perfect in programming logic,* Dec.83.

[Ko] Kozen D. *Results on the propositional μ-Calculus* ICALP 82.

[La1] Lamport L. *"Sometime" is sometimes "not never"* 7th POPL, 80.

[La2] Lamport L. *What good is temporal logic* IFIP 83.

[Ma] Manna Z. *Verification of sequential Programs: Temporal Axiomatisation* Theoretical Foundations of Programming Methodology,81.

[Mi] Milner R. *A calculus for communicating systems* LNCS 92.

[MP] Manna Z., Pnueli A. *The modal logic of programs* ICALP 79,LNCS 77.

[Pn] Pnueli A. *The temporal logic of concurrent programs* 19th FOCS, 77.

[QS1] Queille J.P., Sifakis J. *Specification and verification of concurrent systems in CESAR* Int.Symposium of Programming LNCS 137.

[QS2] Queille J.P.,Sifakis J. *Fairness and related properties in transition systems - a temporal logic to deal with fairness* Acta Informatica 20, 83.

[Ro] Rogers H. *Theory of recursive functions and effective computability* Mc Graw Hill, 67.

[RU] Rescher N., Urquhart A. *Temporal logic* Springer, 71.

[Sch] Schwartz J.P. *Quasar, une réalisation du système CESAR: description, spécification et analyse des applications réparties* Thèse INP Grenoble, 83.

AUTORISATION DE SOUTENANCE

VU les dispositions de l'article 3 de l'arrêté du 16 avril 1974,

VU le rapport de présentation de Monsieur J. SIFAKIS, Chargé de recherche

Mademoiselle GRAF Susanne

est autorisée à présenter une thèse en soutenance pour l'obtention du titre de DOCTEUR de TROISIEME CYCLE, spécialité "Informatique".

Fait à Grenoble, le 14 février 1984

Le Président de l'I.N.P.-G

D. BLOCH
Président
de l'Institut National Polytechnique
de Grenoble

P.O. Le Vice-Président,