USTL FLANDRES ARTOIS — LIFL — U.A 369 du C.N.R.S

LABORATOIRE D'INFO[RMATIQUE FONDAM]ENTALE DE LILLE

N° d'ordre : 475

THÈSE

Nouveau Régime

présentée à

L'UNIVERSITE DES SCIENCES ET TECHNIQUES DE LILLE FLANDRES ARTOIS

pour obtenir le titre de

DOCTEUR en INFORMATIQUE

par

Violaine Thibau

UNE LOGIQUE TRIVALUEE APPLIQUEE A LA PROGRAMMATION LOGIQUE

Soutenue le 8 Janvier 1990 devant la Commission d'examen

M. Nivat	....	Président
G. Ferrand	....	Rapporteur
M. Dauchet	....	Rapporteur
I. Guessarian	....	Examinateur
J. P. Delahaye	....	Examinateur
B. Courcelle	....	Examinateur
G. Comyn	....	Examinateur

UNIVERSITE DES SCIENCES
ET TECHNIQUES DE LILLE
FLANDRES ARTOIS

DOYENS HONORAIRES DE L'ANCIENNE FACULTE DES SCIENCES

M.H. LEFEBVRE, M. PARREAU.

PROFESSEURS HONORAIRES DES ANCIENNES FACULTES DE DROIT ET SCIENCES ECONOMIQUES, DES SCIENCES ET DES LETTRES

MM. ARNOULT, BONTE, BROCHARD, CHAPPELON, CHAUDRON, CORDONNIER, DECUYPER, DEHEUVELS, DEHORS, DION, FAUVEL, FLEURY, GERMAIN, GLACET, GONTIER, KOURGANOFF, LAMOTTE, LASSERRE, LELONG, LHOMME, LIEBAERT, MARTINOT-LAGARDE, MAZET, MICHEL, PEREZ, ROIG, ROSEAU, ROUELLE, SCHILTZ, SAVARD, ZAMANSKI, Mes BEAUJEU, LELONG.

PROFESSEUR EMERITE

M. A. LEBRUN

ANCIENS PRESIDENTS DE L'UNIVERSITE DES SCIENCES ET TECHNIQUES DE LILLE

MM. M. PAREAU, J. LOMBARD, M. MIGEON, J. CORTOIS.

PRESIDENT DE L'UNIVERSITE DES SCIENCES ET TECHNIQUES DE LILLE FLANDRES ARTOIS

M. A. DUBRULLE.

PROFESSEURS - CLASSE EXCEPTIONNELLE

M. CONSTANT Eugène	Electronique
M. FOURET René	Physique du solide
M. GABILLARD Robert	Electronique
M. MONTREUIL Jean	Biochimie
M. PARREAU Michel	Analyse
M. TRIDOT Gabriel	Chimie Appliquée

PROFESSEURS - 1ère CLASSE

M. BACCHUS Pierre	Astronomie
M. BIAYS Pierre	Géographie
M. BILLARD Jean	Physique du Solide
M. BOILLY Bénoni	Biologie
M. BONNELLE Jean-Pierre	Chimie-Physique
M. BOSCQ Denis	Probabilitéss
M. BOUGHON Pierre	Algèbre
M. BOURIQUET Robert	Biologie Végétale
M. BREZINSKI Claude	Analyse Numérique

M. BRIDOUX Michel	Chimie-Physique
M. CELET Paul	Géologie Générale
M. CHAMLEY Hervé	Géotechnique
M. COEURE Gérard	Analyse
M. CORDONNIER Vincent	Informatique
M. DAUCHET Max	Informatique
M. DEBOURSE Jean-Pierre	Gestion des Entreprises
M. DHAINAUT André	Biologie Animale
M. DOUKHAN Jean-Claude	Physique du Solide
M. DYMENT Arthur	Mécanique
M. ESCAIG Bertrand	Physique du Solide
M. FAURE Robert	Mécanique
M. FOCT Jacques	Métallurgie
M. FRONTIER Serge	Ecologie Numérique
M. GRANELLE Jean-Jacques	Sciences Economiques
M. GRUSON Laurent	Algèbre
M. GUILLAUME Jean	Microbiologie
M. HECTOR Joseph	Géométrie
M. LABLACHE-COMBIER Alain	Chimie Organique
M. LACOSTE Louis	Biologie Végétale
M. LAVEINE Jean-Pierre	Paléontologie
M. LEHMANN Daniel	Géométrie
Mme LENOBLE Jacqueline	Physique Atomique et Moléculaire
M. LEROY Jean-Marie	Spectrochimie
M. LHOMME Jean	Chimie Organique Biologique
M. LOMBARD Jacques	Sociologie
M. LOUCHEUX Claude	Chimie Physique
M. LUCQUIN Michel	Chimie Physique
M. MACKE Bruno	Physique Moléculaire et Rayonnements Atmosph.
M. MIGEON Michel	E.U.D.I.L.
M. PAQUET Jacques	Géologie Générale
M. PETIT Francis	Chimie Organique
M. POUZET Pierre	Modélisation - calcul Scientifique
M. PROUVOST Jean	Minéralogie
M. RACZY Ladislas	Electronique
M. SALMER Georges	Electronique
M. SCHAMPS Joel	Spectroscopie Moléculaire
M. SEGUIER Guy	Electrotechnique
M. SIMON Michel	Sociologie
Melle SPIK Geneviève	Biochimie
M. STANKIEWICZ François	Sciences Economiques
M. TILLIEU Jacques	Physique Théorique
M. TOULOTTE Jean-Marc	Automatique
M. VIDAL Pierre	Automatique
M. ZEYTOUNIAN Radyadour	Mécanique

PROFESSEURS - 2ème CLASSE

M. ALLAMANDO Etienne	Composants Electroniques
M. ANDRIES Jean-Claude	Biologie des organismes
M. ANTOINE Philippe	Analyse
M. BART André	Biologie animale
M. BASSERY Louis	Génie des Procédés et Réactions Chimiques

Mme BATTIAU Yvonne	Géographie
M. BEGUIN Paul	Mécanique
M. BELLET Jean	Physique Atomique et Moléculaire
M. BERTRAND Hugues	Sciences Economiques et Sociales
M. BERZIN Robert	Analyse
M.BKOUCHE Rudolphe	Algèbre
M. BODARD Marcel	Biologie Végétale
M. BOIS Pierre	Mécanique
M. BOISSIER Daniel	Génie Civil
M. BOIVIN Jean-Claude	Spectroscopie
M. BOUQUELET Stéphane	Biologie Appliquée aux enzymes
M. BOUQUIN Henri	Gestion
M. BRASSELET Jean-Paul	Géométrie et Topologie
M. BRUYELLE Pierre	Géographie
M. CAPURON Alfred	Biologie Animale
M. CATTEAU Jean-Pierre	Chimie Organique
M. CAYATTE Jean-Louis	Sciences Economiques
M. CHAPOTON Alain	Electronique
M. CHARET Pierre	Biochimie Structurale
M. CHIVE Maurice	Composants Electroniques Optiques
M. COMYN Gérard	Informatique Théorique
M. COQUERY Jean-Marie	Psychophysiologie
M. CORIAT Benjamin	Sciences Economiques et Sociales
Mme CORSIN Paule	Paléontologie
M. CORTOIS Jean	Physique Nucléaire et Corpusculaire
M. COUTURIER Daniel	Chimie Organique
M. CRAMPON Norbert	Tectolique Géodynamique
M. CROSNIER Yves	Electronique
M. CURGY Jean-Jacques	Biologie
Melle DACHARRY Monique	Géographie
M. DEBRABANT Pierre	Géologie Appliquée
M. DEGAUQUE Pierre	Electronique
M. DEJAEGER Roger	Electrochimie et Cinétique
M. DELAHAYE Jean-Paul	Informatique
M. DELORME Pierre	Physioliogie Animale
M. DELORME Robert	Sciences Economiques
M. DEMUNTER Paul	Sociologie
M. DENEL Jacques	Informatique
M. DE PARIS Jean Claude	Analyse
M. DEPREZ Gilbert	Physique du Solide - Cristallographie
M. DERIEUX Jean-Claude	Microbiologie
Melle DESSAUX Odile	Spectroscopie de la réactivité Chimique
M. DEVRAINNE Pierre	Chimie Minérale
Mme DHAINAUT Nicole	Biologie Animale
M. DHAMELINCOURT Paul	Chimie Physique
M. DORMARD Serge	Sciences Economiques
M. DUBOIS Henri	Spectroscopie Hertzienne
M. DUBRULLE Alain	Spectroscopie Hertzienne
M. DUBUS Jean-Paul	Spectrométrie des Solides
M. DUPONT Christophe	Vie de la firme (I.A.E.)
Mme EVRARD Micheline	Génie des procédés et réactions chimiques
M. FAKIR Sabah	Algèbre
M. FAUQUAMBERGUE Renaud	Composants électroniques

M. FONTAINE Hubert	Dynamique des cristaux
M. FOUQUART Yves	Optique atmosphérique
M. FOURNET Bernard	Biochimie Sturcturale
M. GAMBLIN André	Géographie urbaine, industrielle et démog.
M. GLORIEUX Pierre	Physique moléculaire et rayonnements Atmos.
M. GOBLOT Rémi	Algèbre
M. GOSSELIN Gabriel	Sociologie
M. GOUDMAND Pierre	Chimie Physique
M. GOURIEROUX Christian	Probabilités et Statistiques
M. GREGORY Pierre	I.A.E.
M. GREMY Jean-Paul	Sociologie
M. GREVET Patrice	Sciences Economiques
M. GRIMBLOT Jean	Chimie Organique
M. GUILBAULT Pierre	Physiologie animale
M. HENRY Jean-Pierre	Génie Mécanique
M. HERMAN Maurice	Physique spatiale
M. HOUDART René	Physique atomique
M. JACOB Gérard	Informatique
M. JACOB Pierre	Probabilités et Statistiques
M. Jean Raymond	Biologie des populations végétales
M. JOFFRE Patrick	Vie de la firme (I.A.E.)
M. JOURNEL Gérard	Spectroscopie hertzienne
M. KREMBEL Jean	Biochimie
M. LANGRAND Claude	Probabilités et statistiques
M. LATTEUX Michel	Informatique
Mme LECLERCQ Ginette	Catalyse
M. LEFEBVRE Jacques	Physique
M. LEFEBVRE Christian	Pétrologie
Melle LEGRAND Denise	Algèbre
Melle LEGRAND Solange	Algèbre
M. LEGRAND Pierre	Chimie
Mme LEHMANN Josiane	Analyse
M. LEMAIRE Jean	Spectroscopie hertzienne
M. LE MAROIS Henri	Vie de la firme (I.A.E.)
M. LEROY Yves	Composants électroniques
M. LESENNE Jacques	Systèmes électroniques
M. LHENAFF René	Géographie
M. LOCQUENEUX Robert	Physique théorique
M. LOSFELD Joseph	Informatique
M. LOUAGE Francis	Electronique
M. MAHIEU Jean-Marie	Optique-Physique atomique
M. MAIZIERES Christian	Automatique
M. MAURISSON Patrick	Sciences Economiques et Sociales
M. MESMACQUE Gérard	Génie Mécanique
M. MESSELYN Jean	Physique atomique et moléculaire
M. MONTEL Marc	Physique du solide
M. MORCELLET Michel	Chimie Organique
M. MORTREUX André	Chimie Organique
Mme MOUNIER Yvonne	Physiologie des structures contractiles
Mme MOUYART-TASSIN Annie Françoise	Informatique
M. NICOLE Jacques	Spectrochimie
M. NOTELET Françis	Systèmes électroniques
M. PARSY Fernand	Mécanique

M. PECQUE Marcel	Chimie organique
M. PERROT Pierre	Chimie appliquée
M. STEEN Jean-Pierre	Informatique

Je suis très reconnaissante au Professeur Maurice Nivat de l'honneur qu'il me fait de présider cette thèse.

Je tiens à remercier:

- le Professeur Jean Paul Delahaye, qui a été mon directeur de recherche, m'a apporté son soutien constant dans l'élaboration de ce travail et qui m'a toujours guidée de ses indications fructueuses.

- le Professeur Max Dauchet qui a accepté d'être rapporteur de ma thèse. C'est lui qui m'a accueillie à l'Université de Lille et a toujours veillé au bon déroulement de mon travail de recherche.

- le Professeur Gérard Ferrand qui a lui aussi accepté d'être rapporteur de ma thèse et dont les compétences dans le domaine de la programmation logique ainsi que ses remarques au cours d'une discussion à Orléans m'ont été très profitables.

- le Professeur Irène Guessarian qui s'est interessée à mon travail, m'a aidée de ses suggestions et a accepté aussi de faire un rapport malgré son peu de temps disponible.

Je remercie aussi tout particulièrement le Professeur Bruno Courcelle qui a accepté de faire partie du jury de cette thèse, ainsi que le Professeur Gérard Comyn qui vient de Munich pour y assister.

Sur un plan personnel, je voudrais remercier tout particulièrement Ludovic et Isabelle Faille, grâce à qui mon séjour hebdomadaire à Lille est toujours une fête. Ils m'ont chaleureusement acceuillie toutes les semaines, sans jamais se lasser. Je remercie enfin tous mes amis dont l'affection, la chaleur et les encouragements m'ont permis de mener à bien ce travail.

A mes parents.

TABLE DES MATIERES

CHAPITRE 1: PRESENTATION

1.1 INTRODUCTION

La négation en programmation logique pose de nombreux problèmes et de multiples tentatives ont été faites ces dernières années pour les résoudre. Ces tentatives peuvent être classées en fonction de la plus ou moins grande fidélité à la négation réellement utilisée dans Prolog: la négation par l'échec (Clark [7], Lloyd [22], Shepherdson [34], [35], Van Gelder [44]). Malheureusement il ne semble pas possible de rester proche de Prolog tout en ayant une sémantique axiomatique simple. Il n'est pas pas certain de pouvoir réellement trouver une sémantique axiomatique correcte et complète qui rende compte de la négation par l'échec. Une autre négation que la négation par l'échec, en apparence très simple est aussi utilisée dans les systèmes experts fonctionnant en chaînage avant et qui ne correspond pas à la négation par l'échec.

La théorie que nous proposons est assez générale pour s'appliquer à la fois à la Programmation Logique et à cette négation utilisée dans les systèmes experts. Elle ne rend compte de la négation par l'échec que partiellement, mais a l'avantage de proposer une théorie algébrique et logique assez simple ayant un grand nombre de bonnes propriétés, semblables à celles de la théorie de Prolog quand on se limite aux clauses de Horn.

Le 'secret' de ce parallèle et des bonnes propriétés du modèle que nous introduisons réside dans l'utilisation d'une certaine logique à trois valeurs de vérité et l'introduction d'un connecteur d'implication approprié. Le résultat obtenu est une méthode pour gérer la négation qui coïncide dans de nombreux cas avec celle de Prolog et qui est très simple. Finalement à condition d'accepter notre logique trivaluée, une lecture purement axiomatique des programmes est possible. On satisfait ainsi le but initial de la programmation logique: programmer en écrivant des axiomes.

Parmi les critiques qu'on peut faire à notre modèle il y celle déjà évoquée que nous ne rendons pas entièrement compte du "vrai Prolog", mais cela nous paraît inévitable si l'on veut conserver une sémantique axiomatique. Plus grave serait la critique que la logique trivaluée n'est pas la logique usuelle et que l'utiliser est donc contraire à l'intuition. Il est vrai que dans certaines situations, les déductions faites en logique trivaluée sont trop faibles (on obtient trop peu de conséquences logiques d'un programme) mais bien souvent la logique trivaluée est naturelle dans les situations où l'information est incomplète – elle est d'ailleurs depuis longtemps utilisée pour cela en Intelligence Artificielle (Turner [39]).

Le but de ce travail n'a pas été seulement les applications directes de cette logique trivaluée à la programmation logique. En effet, le domaine d'application de cette logique trivaluée que nous avons été amenés à introduire, semble plus large: en particulier des applications aux langages de règles pour systèmes experts sont en cours de développement (Delahaye-Mathieu [8 bis]). Par un souci à la fois de généralisation et de rigueur, notre étude s'est donc voulue plus étendue, et nous avons cherché à développer le plus complètement possible tous les aspects de cette logique trivaluée.

En particulier au chapitre 2, nous étudions les problèmes sémantiques (notion d'interprétation, modèles, conséquence, etc.) dans un cadre général sans se limiter aux interprétations de Herbrand et en traitant les formules générales du langage sans se limiter aux règles de programmes.

Au chapitre 3, toujours en vue d'autres applications, nous avons étudié le problème de l'expressivité des différents connecteurs en logique trivaluée.

L'étude de la complétude a été menée au chapitre 4 afin d'étudier le plus rigoureusement possible cette logique dont l'intérêt est manifeste. Le fait que ces théorèmes de complétude puissent s'établir sans obstacles majeurs justifie indirectement le choix du connecteur d'implication (choix justifié par ailleurs par d'autres arguments).

Les chapitres 5 et 6 concernent l'application directe de cette logique à la programmation logique.
Le chapitre 5 concerne la sémantique des programmes avec négation. Il évoque en particulier la propriété d'intersection des modèles et associe à chaque programme un opérateur conséquence dont le plus petit point fixe coïncide avec le plus petit modèle de ce programme.
Le chapitre 6 se justifie par le fait que l'union de deux interprétations trivaluées n'étant plus nécessairement une interprétation trivaluée, les programmes considérés n'admettent plus de plus grand modèle mais une série de modèles maximaux dont l'intersection présente un intérêt: c'est le modèle optimal qui contient plus d'informations que le plus petit modèle.

1.2 POURQUOI AVOIR CHOISI LA LOGIQUE TRIVALUEE ET INTRODUIT UN NOUVEAU CONNECTEUR D'IMPLICATION?

Dès que l'on introduit la négation dans le corps d'une règle d'un programme défini sans négation, on obtient en logique classique, avec le connecteur d'implication habituel, une trop grande expressivité.

En effet, la formule:

$$A \leftarrow A_1, ..., A_k, \neg B_1, ..., \neg B_q$$

est logiquement équivalente à la formule:

$$\neg A_1 \vee ... \vee \neg A_k \vee B_1 \vee ... \vee B_q \vee A$$

que l'on peut écrire suivant la notation clausale:

$$A, B_1, ..., B_q \leftarrow A_1, ..., A_k$$

En fait, dès que l'on introduit la négation dans les programmes définis sans négation, on n'obtient plus les clauses définies de programme, au sens de Lloyd [22] ou encore des règles mais nimporte quelle clause:

$$A_1, ..., A_k \leftarrow B_1, ..., B_n.$$

On peut alors se demander pourquoi conserver l'écriture d'un programme sous forme de règles et ne pas utiliser simplement l'écriture clausale.

On peut répondre de plusieurs manières à cette question, suivant que l'on veut ou non être fidèle à la négation par l'échec.

Conserver un système à base de règles permet de généraliser la résolution SLD basée sur la notion de règles, à la résolution SLDNF.

En effet, dès que l'on introduit la négation dans un programme, on perd la justesse et la complétude de la résolution SLD (équivalence entre réponse calculée et réponse correcte (Lloyd [22])), même si celui-ci ne comporte que des règles positives (i.e. dont la conclusion est un atome positif).

Par exemple, si A, B, C sont des atomes du langage L, et Pr est le programme suivant:

$$B \leftarrow D, A$$

$$B \leftarrow D, \neg A$$
$$D \leftarrow$$

La substitution Identité est une réponse correcte pour Pr $\cup$ {B} car $\forall$B est une conséquence logique de Pr. Or Pr $\cup$ {B} n'a pas de réfutation SLD.

Cependant, cette généralisation est loin d'être évidente.
Tout d'abord, elle fait appel à la notion de complété de Clark qui donne un double sens aux règles et impose des contraintes d'intégrité sur l'égalité en l'obligeant à être interprétée par l'égalité syntaxique dans tout modèle de P.
Ensuite, le résultat de complétude de la résolution SLDNF ne s'obtient que pour un certain type de programmes que sont les programmes hiérarchiques (Lloyd [22]). Ceci revient plus ou moins à supposer qu'il n'y a pas de récursion et est par conséquent très restrictif.

L'autre réponse que nous avons choisie ici, est de donner une expressivité différente aux formules $A \leftarrow A_1, ..., A_k, \neg B_1, ..., \neg B_q$, de sorte que l'écriture $A \leftarrow A_1, ..., A_k, \neg B_1, ..., \neg B_q$ ne soit plus logiquement équivalente à $\neg A_1 \vee ... \vee \neg A_k \vee B_1 \vee ... \vee B_q \vee A$ et qu'une règle de programme ne soit plus logiquement équivalente à nimporte quelle clause.

Pour ce faire, on aurait pu conserver une logique à deux valeurs de vérité et changer simplement la table du connecteur d'implication. Mais les choix sont limités et vite éliminés si l'on veut conserver une logique cohérente et en particulier complète.

Nous avons donc introduit une troisième valeur de vérité et donné une nouvelle interprétation du connecteur d'implication $\Rightarrow$, noté $\rightarrow$ afin que la formule $P \rightarrow Q$ ne soit plus logiquement équivalente à $\neg P \vee Q$.

Plusieurs travaux ont été menés dans cette voie et nous donnons dans le paragraphe suivant, sans prétendre à l'exhaustivité, une idée des diverses recherches dans ce domaine.

1.3 SITUATION DU TRAVAIL

- L'un des précurseurs de l'utilisation de cette logique à trois valeurs de vérité pour la programmation logique, qui consiste en fait à ne pas attribuer une valeur de vérité à chaque atome, fut **Melvin Fitting** [9], [10], [11], [12], [13], [14], [15].

Dans son article, *A Kripke Kleene semantics for logic programs* [9], Fitting introduit une troisième valeur de vérité I (Indéterminé). Il s'intéresse aux programmes ne comportant que des règles positives, de la forme:

$$A \leftarrow B_1, ..., B_n$$

où A est un atome et les B_i sont des formules contenant les symboles ($\wedge$, $\vee$, $\neg$).

Il définit de la même façon que nous le ferons ici, les valeurs de vérité de:

$\neg F$, $F \wedge G$, $F \vee G$, $\exists x\ F$, $\forall x\ G$.

Par contre il ne définit pas la table d'un nouveau connecteur d'implication, mais un connecteur d'équivalence $\cong$ défini par:

$X \cong Y$ est Vrai si et seulement si X et Y ont la même valeur de vérité, Faux sinon.
Son connecteur $\cong$ correspondra à notre opérateur d'équivalence forte, $\leftrightarrow$.

Par modèle d'un programme, il entend modèle de Herbrand de son complété dans lequel l'égalité est interprétée par l'égalité syntaxique.

Le complété d'un programme est obtenu en remplaçant chaque règle:

$$p(t_1, ..., t_n) \leftarrow B_1, ..., B_m$$

par:

$$p(x_1, ..., x_n) \leftarrow \exists y_1 ... \exists y_k\ (x_1 = t_1 \wedge ... \wedge x_n = t_n \wedge B_1 \wedge ... \wedge B_m)$$

où $y_1, ..., y_k$ sont les variables de $t_1, ..., t_n$ et de $B_1, ..., B_m$.

Si D_i désigne la formule $\exists y_1 ... \exists y_k\ (x_1 = t_1 \wedge ... \wedge x_n = t_n \wedge B_1 \wedge ... \wedge B_m)$, le complété de P s'obtient en rassemblant toutes les règles comprenant le même symbole de prédicat p en tête de règle en la règle, de la façon suivante:

$$p(x_1, ..., x_n) \cong D_1 \vee ... \vee D_p.$$

Fitting s'intéresse ensuite à l'interprétation sémantique des programmes logiques en tant que points fixes d'opérateurs. Son ordre sur l'ensemble IHT(L) des interprétations de Herbrand trivaluées est l'inclusion, qui correspond à l'ordre $I \leq F$, $I \leq V$ sur {V, F, I}. A chaque interprétation de Herbrand trivaluée, on associe l'application vv_i définie de l'ensemble des atomes clos du langage dans l'ensemble $\{V, F, I\}$.

L'opérateur introduit F_{Pr} est défini par son image sur une interprétation de Herbrand trivaluée:

$F_{Pr}(i) = \{ato/$ il existe une instance close $ato \leftarrow B_1, ..., B_m$ de Pr telle que $vv_i(B_1, ..., B_m) = V\}$

$\cup \{\neg ato/$ pour toute instance close $ato \leftarrow B_1, ..., B_m$ de P on ait: $vv_i(B_1, ..., B_m) = F\}$.

Cet opérateur est monotone et les points fixes de F_{Pr} sont exactement les modèles du complété de Pr.

Puisque une union de modèles partiels n'est pas nécessairement une interprétation trivaluée, l'ensemble des interprétations trivaluées ne constitue plus un treillis complet mais seulement un semi treillis complet: l'intersection de deux interprétations trivaluées en est une et une union d'interprétations trivaluées deux à deux consistantes est une interprétation trivaluée.

L'opérateur F_{Pr} possède un plus petit point fixe qui est le plus petit modèle de comp(P) mais par contre, il n'admet pas de plus grand point fixe, mais une série de points fixes **maximaux**.

Il y alors deux façons de définir la **sémantique** d'un programme logique:

- Le pppf de F_{Pr}.
- Le point fixe optimal de F_{Pr} qui est le plus grand point fixe inférieur à l'intersection de tous les points fixes maximaux ou encore le plus grand point fixe consistant.

L'idée étant que le point fixe optimal peut être plus intéressant que le plus petit point fixe puisqu'il contient plus d'informations.

Notre théorie se rapproche de celle de Fitting mais nous donnons une interprétation du connecteur $\rightarrow$ et nous définissons la notion de modèle de Herbrand de Pr sans passer par son complété. Nous retrouvons d'ailleurs son connecteur $\cong$ et son opérateur F_{Pr} à partir de notre connecteur $\rightarrow$ et notre opérateur T_{Pr} en complétant notre programme. Tout d'abord la formule:

$X \cong Y$

sera logiquement équivalente à la formule:

$$(X \rightarrow Y) \wedge (Y \rightarrow X) \wedge (\neg X \rightarrow \neg Y) \wedge (\neg Y \rightarrow \neg X)$$

où $\rightarrow$ est le nouveau connecteur d'implication introduit. Nous définirons donc deux sortes de complétion et nous démontrerons que

$$F_{Pr} = T_{Comp(\neg,\ Pr)}$$

où Comp($\neg$, Pr) désigne le $\neg$-complété de Pr. Les modèles de notre programme seront exactement les post-points fixes de notre opérateur T_{Pr} alors que pour Fitting, les modèles du complété de Pr (au sens défini ci-dessus) sont les points fixes de F_{Pr}. Les modèles de Fitting seront donc en particulier des modèles de notre programme. Le plus petit modèle du complété d'un programme au sens de Fitting coïncidera avec le plus petit modèle de Comp($\neg$, Pr) et celui de Comp($\rightarrow$, Comp($\neg$, Pr)) mais pas nécessairement avec le plus petit modèle de notre programme, ce dernier étant le plus petit point fixe de notre opérateur T_{Pr} et coïncidant avec le plus petit modèle de Comp($\rightarrow$, Pr).

- **J.L Lassez** [21] introduit aussi la logique à trois valeurs de vérité. Les programmes considérés ne comportent que des clauses de Horn: $A \leftarrow A_1, \ldots, A_n$.

Son connecteur d'implication est celui de Lukasiewicz, et possède la table de vérité suivante:

$A \rightarrow B$ n'est pas Vrai si A vaut V et B vaut I ($A \rightarrow B$ vaut I), si A vaut V et B vaut F ($A \rightarrow B$ vaut F), si A vaut I et B vaut F ($A \rightarrow B$ vaut I).
$A \rightarrow B$ est Vrai dans tous les autres cas.

La formule $P \rightarrow Q$ n'est pas logiquement équivalente à: $\neg P \vee Q$, par exemple, si P et Q ont la valeur de vérité I.

Son ordre sur {V, F, I} correspond à l'ordre d'inclusion sur l'ensemble des interprétations de Herbrand trivaluées. ($I \leq F, I \leq V$)

L'opérateur τ associé à un programme logique P est le suivant: (une interprétation de Herbrand est considérée comme une application de l'ensemble des atomes clos dans {V, F, I})

Si il existe une instance close d'une règle de P ayant A en tête de règle, $A \leftarrow B_1, \ldots, B_n$, telle que $f(B_1 \wedge \ldots \wedge B_n) = V$, $\tau(f)(A) = V$.

Si pour toute instance close des règles de P ayant A en tête de règle, A $\leftarrow$ $B_1, ..., B_n$, $f(B_1 \wedge ... \wedge B_n) = F$, $\tau(f)(A) = f(A)$.

Si pour toute instance close des règles de P ayant A en tête de règle, A $\leftarrow$ $B_1, ..., B_n$, $f(B_1 \wedge ... \wedge B_n) \neq V$, et il existe une instance close d'une clause de P ayant A en tête de règle, A $\leftarrow$ $B_1, ..., B_n$, telle que $f(B_1 \wedge ... \wedge B_n) = I$, $\tau(f)(A) = \inf(V, f(A))$.

Pour le cas où A ne figure pas en tête d'une règle de P, $\tau(f)(A) = f(A)$.

Dans l'article [21] sont démontrées les propositions suivantes:

- Il y a coïncidence entre les points fixes de τ et les modèles de P.

$$\tau(f) = f \Leftrightarrow f \textit{ est un modèle de } P.$$

- Pour cet opérateur la notion de point fixe optimal ne présente pas d'intérêt puisque le point fixe optimal de τ et son plus petit point fixe coïncident.
Cela provient essentiellement du fait que l'interprétation où tous les atomes clos ont la valeur de vérité Vrai est un modèle de nimporte quel programme comportant des clauses de Horn. Il s'ensuit que ni le plus petit modèle ni le modèle optimal d'un programme (tous deux inclus dans l'interprétation maximale constamment égale à Vrai), ne prennent la valeur de vérité Faux. Les informations négatives résultant du programme complété, ne sont pas représentées dans cette sémantique qui ne semble donc pas convenir à la programmation logique avec négation.

- **Mycroft** [30] s'intéresse aussi à la logique trivaluée. Cependant les clauses qu'il considère sont des clauses définies de programme, son but étant de montrer qu'une logique à plusieurs valeurs de vérité est mieux adaptée à la résolution SLD.
Son opérateur est le même que celui de Van Emden et Kowalski [43]. Les tables de vérité des connecteurs $\wedge$, et $\vee$ sont les mêmes que celles utilisées ici. Il n'introduit pas de nouveau connecteur d'implication.
Sa partie sur la logique trivaluée est essentiellement contenue dans celle de Fitting.

- **Kunen** [19] utilise la logique trivaluée uniquement pour des programmes ayant des règles positives. Son travail est proche de celui de Fitting car il considère le complété d'un programme et son opérateur n'est autre que celui de Fitting.
Selon lui, la sémantique d'un programme logique P n'est pas le plus petit point fixe de l'opérateur T mais $T \uparrow \omega$, qui n'est pas toujours un modèle de P.

D'après nous, une bonne sémantique doit s'exprimer en termes d'axiomes et de modèles.

- Enfin, **Przymusinski** [31], a jeté une vision nouvelle sur la logique à trois valeurs de vérité.

Dans son article, *Non monotonic formalisms and logic programming*, il introduit trois valeurs de vérité et un ordre sur $\{V, F, I\}$, $F \leq I \leq V$ qui ne correspond pas à l'ordre d'inclusion sur l'ensemble des interprétations trivaluées.

D'autre part une interprétation de Herbrand I est la donnée de deux ensembles d'atomes clos: $\mathcal{V}$ et $\mathcal{F}$, sans préciser que $\mathcal{V} \cap \mathcal{F} = \emptyset$, ce qui permet à un atome clos d'avoir deux valeurs de vérité.

Dans ce contexte, la **négation n'est pas monotone**, au sens où:
si f et g sont deux interprétations telles que: pour tout atome clos, $f(A) \leq g(A)$, on n'a pas: $f(\neg A) \leq g(\neg A)$, par exemple si $f(A) = F$, et $g(A) = V$.

Son connecteur d'implication, bien qu'il ne prenne jamais la valeur I, n'est pas le même que celui introduit ici.
En effet, il est défini par: $vv_i(A \to B) = V$ si $vv_i(B) \geq vv_i(A)$, $vv_i(A \to B) = F$ sinon.
On a donc si A prend la valeur I et B la valeur F, $vv_i(A \to B) = F$ alors que dans notre cas, pour l'opérateur $\to$ que nous définissons ici $vv_i(A \to B) = V$.
Même, si l'ordre sur $\{V, F, I\}$ était $I \leq F, I \leq V$ (celui que nous choisissons ici), si $vv_i(A) = F$ et $vv_i(B) = I$, $vv_i(A \to B) = F$, alors que dans notre cas, pour le connecteur $\to$ que nous définissons ici $vv_i(A \to B) = V$.

Les programmes logiques considérés sont des programmes définis ayant des règles positives ($A \leftarrow L_1, ..., L_n$, A est un atome et les L_i sont des littéraux).

L'opérateur associé à chaque programme se construit à deux niveaux:
Pour deux ensembles d'atomes clos $\mathcal{V}$ et $\mathcal{F}$, on définit, étant donnée une interprétation I, $T_I(\mathcal{V}) = \{A /$ il existe une instance close $A \leftarrow L_1, ..., L_n$ de P telle que $\forall i \leq n$, L_i soit Vrai dans I ou $L_i \in \mathcal{V}\}$
$F_I(\mathcal{F}) = \{A /$ pour toute instance close $A \leftarrow L_1, ..., L_n$ de P il existe $i \leq n$ tel que L_i soit Faux dans I ou $L_i \in \mathcal{F}\}$
Pour une interprétation donnée $I = <\mathcal{V}, \mathcal{F}>$, on définit T_I et F_I par:
$T_I\uparrow 0 = \emptyset$, $T_I\uparrow n{+}1 = T_I(T_I\uparrow n)$ et $T_I = \cup_{n \in \mathbb{N}} T_I\uparrow n$, et

$F_I\downarrow 0 = B_P$ la base de Herbrand de P,
$F_I\downarrow n+1 = F_I(F_I\downarrow n)$ et $F_I = \cap_{n \in \mathbb{N}} F_I\downarrow n$.
On est alors en mesure de définir l'opérateur

$$\mathcal{I}: I = <\mathcal{V}, \mathcal{F}> \rightarrow I \cup <T_I, F_I>$$

Cet opérateur est monotone et son plus petit point fixe est le plus petit modèle trivalué de P.
Cependant, les modèles obtenus avec son connecteur d'implication sont des modèles dans notre théorie.

- D'autre part, **Manna et Shamir** [28], ont développé une théorie du point fixe optimal pour une fonctionnelle monotone définie sur un ensemble de fonctions partielles. Cette théorie nous intéressera puisque les interprétations de Herbrand trivaluées que nous considérons peuvent être considérées comme des fonctions partielles de l'ensemble des atomes de Herbrand dans l'ensemble {V, F} (non définies là où elles prennent la valeur Indéterminé).

A côté de ces travaux, un autre développement de la logique trivaluée et de son application à la programmation logique semblait possible.

1.4 COMMENT CONSERVER LES PROPRIETES OBTENUES POUR LES PROGRAMMES SANS NEGATION

En logique classique, pour étudier les programmes sans négation, on fait appel à la notion de modèle de Herbrand.

Une interprétation de Herbrand bivaluée est un sous ensemble de l'ensemble des atomes de Herbrand et à l'ordre $F \leq V$ sur {V, F} correspond l'ordre d'inclusion sur les interprétations de Herbrand. Si l'on note her(L) l'ensemble des atomes clos du langage, l'ensemble des interprétations de Herbrand bivaluées est donc l'ensemble des parties de her(L), $2^{her(L)}$. L'ordre induit par $F \leq V$ sur {V, F} rend la bijection ϕ: $2^{her(L)} \rightarrow$ {Fonctions de her(L) dans {V, F}}, qui à chaque interprétation i associe vv_i - où $vv_i(ato)$ désigne la valeur de vérité d'un atome clos ato du langage relativement à i - monotone si l'ordre sur $2^{her(L)}$ est l'inclusion et l'ordre sur {Fonctions de her(L) dans {V, F}} est l'ordre induit par $F \leq$ V.

Si $2^{her(L)}$ est l'ensemble des interprétations bivaluées de Herbrand, la première propriété importante obtenue est une propriété de **monotonie:**

Si l'on désigne par $\mathcal{F}(L)$ l'ensemble des formules du langage L ne comportant que les symboles $\exists$, $\forall$, $\wedge$, $\vee$.

L'application ϕ: $2^{her(L)} \rightarrow$ {applications de $\mathcal{F}(L)$ dans {V, F}},
$i \rightarrow vv_i/_{\mathcal{F}(L)}$, est monotone.

Cela signifie que: si f est une formule du langage L comportant les symboles $\exists$, $\forall$, $\wedge$, $\vee$, si i et j sont deux éléments de IHB(L) tels que $i \subseteq j$, alors on a: $vv_i(f) \leq vv_j(f)$ (*). L'expression $vv_i(f)$ désigne la valeur de vérité d'une formule f relativement à i. Si f n'est pas close,
$vv_i(f) = vv_i(\forall f)$.

Une formule f vérifiant la propriété (*) est dite monotone.

Si on ne restreint pas l'application vv_i aux formules ne comportant que les symboles $\exists$, $\forall$, $\wedge$, $\vee$, on perd la propriété de monotonie:

Dès que l'on introduit le symbole $\neg$:
si f est la formule: $\neg p(x)$, la base de Herbrand B_L est l'ensemble: {p(a)}, i l'interprétation égale à $\emptyset$, et j l'interprétation égale à {p(a)}, on a: $vv_i(f) = vv_i(\forall x(\neg p(x)) = V$ alors que $vv_j(f) = F$.

De même, dès que l'on introduit le symbole $\Rightarrow$:
si l'on considère la formule $p(x) \Rightarrow q(x)$, et i l'interprétation égale à l'ensemble vide (dans laquelle tout atome clos a la valeur Faux) et j l'interprétation {p(a)}, $vv_i(\forall x(p(x) \Rightarrow q(x))) = V$ alors que $vv_j(f) = F$.
Par ailleurs, le connecteur d'implication considéré comme application de $\{V, F\}^2$ dans {V, F} est non monotone: $\Rightarrow(F, F) = V$ et $\Rightarrow(V, F) = F$.

Le fait que l'application $i \rightarrow vv_i/_{\mathcal{F}(L)}$, soit monotone, a de nombreuses conséquences sur l'étude des programmes Pr comportant des règles de la forme:

$$\text{ato} \Leftarrow \text{for}$$

où for est un élément de $\mathcal{F}(L)$, c'est-à-dire ne comporte que les symboles $\exists$, $\forall$, $\wedge$, $\vee$ et où $\Rightarrow$ désigne le connecteur d'implication habituel, c'est-à-dire que la formule $P \Rightarrow Q$ a la même valeur de vérité que la formule $\neg P \vee Q$.

- Tout d'abord elle entraîne que l'intersection de modèles de Herbrand d'un tel programme est encore un modèle de ce programme.

- Elle entraîne aussi la monotonie de l'opérateur conséquence B_{Pr} associé à un tel programme et introduit pour la première fois dans [43].

Si B_{Pr} est l'opérateur:

B_{Pr}: IHB(L) $\rightarrow$ IHB(L),
 i $\rightarrow$ {ato/ il existe une instance ato $\Leftarrow$ for de Pr, vv_i(for) = V},

cet opérateur étant **monotone**, admet un **plus petit point fixe** qui coïncide avec le plus petit modèle de Herbrand de Pr (intersection des modèles de Herbrand de Pr): en effet, les modèles de Pr sont les post-points fixes de B_{Pr} et B_{Pr} étant monotone, on a: $B_{Pr}\uparrow\alpha \subseteq$ i, pour tout i modèle de Pr.

Dans le cas d'un programme défini, B_{Pr} est continu et
pppf(B_{Pr}) = $B_{Pr}\uparrow\omega$.

Comme B_{Pr} est monotone, et que les interprétations de Herbrand bivaluées forment un treillis complet, B_{Pr} admet aussi un plus grand point fixe.
Il n'y a donc pas ici de points fixes maximaux ni de point fixe optimal comme nous l'obtiendrons dans le cas des programmes avec négation.

Si l'on introduit la négation dans les programmes logiques, c'est-à-dire si l'on veut maintenant étudier les programmes comportant des règles de la forme:

$$\text{lit} \Leftarrow \text{for}$$

où lit est un littéral et for une formule comportant les symboles $\exists$, $\forall$, $\wedge$, $\vee$, $\neg$, $\Rightarrow$, $\Leftrightarrow$, ($P \Leftrightarrow Q$ a la même valeur de vérité que $(P \Rightarrow Q)\wedge(Q \Rightarrow P)$) on perd la propriété d'intersection des modèles même pour les programmes n'ayant que des règles positives. Par exemple:

Si p, q, r sont des symboles de prédicats, f un symbole de fonction, et Pr le programme suivant:

$$q(x) \Leftarrow p(x), r(x)$$
$$q(f(x)) \Leftarrow p(x), \neg r(x)$$
$$p(x) \Leftarrow$$

Pour obtenir la base de Herbrand du programme, on rajoute une constante arbitraire a, l'univers de Herbrand (l'ensemble de tous les termes clos), est alors $\{f^n(a), n \geq 0\}$.

Tout modèle de Pr comprend, pour un n donné:

soit $p(f^n(a)), r(f^n(a)), q(f^n(a))$;
soit $p(f^n(a)), q(f^{n+1}(a))$;

Dans tout modèle de Pr, on a $q(f^n(a))$ ou $q(f^{n+1}(a))$;

Si l'on prend pour M_1, le modèle contenant pour $k \leq n$, $p(f^k(a)), r(f^k(a)), q(f^k(a))$; $p(f^{n+1}(a)), q(f^{n+2}(a))$; pour $k \geq n+2$, $p(f^k(a)), r(f^k(a)), q(f^k(a))$; pour M_2, le modèle contenant pour $k \leq n-1$, $p(f^k(a)), r(f^k(a)), q(f^k(a))$; $p(f^n(a)), q(f^{n+1}(a))$; $p(f^{n+1}(a)), q(f^{n+2}(a))$; pour $k \geq n+2$, $p(f^k(a))$, $r(f^k(a)), q(f^k(a))$;

$M_1 \cap M_2$ ne contient ni $q(f^n(a))$ ni $q(f^{n+1}(a))$; ce n'est donc pas un modèle de Pr.

Cela provient du fait qu'une formule comprenant les symboles $\exists, \forall, \wedge, \vee, \neg, \Rightarrow, \Leftrightarrow$, n'est plus monotone au sens où: elle ne vérifie plus nécessairement: $i \subseteq j$ implique $vv_i(for) \leq vv_j(for)$.

Pour étudier les programmes avec négation, nous avons voulu conserver des propriétés analogues à celles de la logique classique obtenues pour des programmes sans négation et en particulier une propriété de monotonie pour le corps des règles des programmes étudiés.

Pour cela, nous introduisons une troisième valeur de vérité I. Nous définissons une interprétation de Herbrand trivaluée comme un sous ensemble de Her(L) (l'ensemble des litteraux clos) vérifiant:

$$ato \in i \text{ implique que } \neg ato \notin i$$

On ne s'autorise donc pas, comme Przymusinski à donner deux valeurs de vérité à un atome.
Pour faciliter le parallèle avec le cas bivalué, nous considérons une interprétation de Herbrand bivaluée comme un cas particulier d'interprétation trivaluée, en écrivant tous ses éléments au lieu de n'écrire que les atomes positifs.

Une interprétation bivaluée est une interprétation trivaluée telle que:

$$ato \in i \text{ ou } \neg ato \in i$$

Nous utiliserons l'application pos: $\mathbb{HB}(L) \rightarrow 2^{her(L)}$
$i \rightarrow pos(i) = \{ato \in her(L) \cap i\}$

her(L) désignant l'ensemble des atomes clos de L, et $2^{her(L)}$ l'ensemble de ses parties, pour nous ramener à l'écriture habituelle.

Les interprétations de Herbrand trivaluées sont des interprétations **partielles** dans le sens où tout atome clos ne possède pas nécessairement une valeur de vérité (V ou F).

Un atome clos ne possède qu'une valeur de vérité contrairement à Przymusinski.

Nous avons choisi sur l'ensemble $\{V, F, I\}$ l'ordre $I \leq F, I \leq V$ qui correspond à l'ordre naturel d'inclusion sur l'ensemble des interprétations de Herbrand trivaluées et qui n'est pas l'ordre choisi par Przymusinski sur $\{V, F, I\}$. Cet ordre rend la bijection ϕ: $\mathbb{HT}(L) \rightarrow$ {Fonctions de her(L) dans $\{V, F, I\}$}, qui à chaque interprétation i associe vv_i, où $vv_i(ato)$ désigne la valeur de vérité d'un atome clos du langage relativement à i, monotone

Cet ordre choisi sur $\{V, F, I\}$ nous permettra d'**étendre** la propriété de **monotonie** aux formules constituant le corps des programmes étudiés, c'est-à-dire aux formules comportant les symboles $\exists, \forall, \wedge, \vee, \neg, \Rightarrow, \Leftrightarrow$ où $\Rightarrow$ est le connecteur d'implication de Kleene et $\Leftrightarrow$ le connecteur d'équivalence qui lui est associé.

Si i et j sont deux interprétations de Herbrand trivaluées, et for une formule comportant les symboles $\exists, \forall, \wedge, \vee, \neg, \Rightarrow, \Leftrightarrow$ alors:

$$i \subseteq j \text{ implique } vv_i(for) \leq vv_j(for)$$

Si $\mathfrak{For}(L)$ désigne les formules du langage L comportant les symboles $\exists$, $\forall, \wedge, \vee, \neg, \Rightarrow, \Leftrightarrow$, c'est maintenant l'application:

ϕ': $\mathbb{HT}(L) \rightarrow$ {applications de $\mathfrak{For}(L)$ dans $\{V, F, I\}$},
$i \rightarrow vv_{i/\mathfrak{For}(L)}$,

qui est monotone.

L'idée étant, que si for est une formule dans laquelle on ne connait pas la valeur de vérité de certains atomes, la valeur de vérité de cette formule est soit inconnue, soit ne peut changer (passer de V à F ou inversement) si on apporte des informations supplémentaires sur des atomes la composant.

Si l'on conserve le connecteur $\Leftarrow$ dans les règles lit $\Leftarrow$ for de programmes, on n'obtient toujours pas la propriété d'intersection des modèles. C'est pourquoi l'on introduit un nouveau connecteur d'implication noté $\rightarrow$, que l'on appellera implication non monotone comme connecteur central des règles de programmes avec négation. On étudiera alors des programmes dont les règles sont de type lit $\leftarrow$ for.

Ce connecteur d'implication n'est ni celui de Lukaziewicz, ni celui de Kleene, ni celui de Przymusinski. Il est défini de la manière suivante:

$A \rightarrow B$ est Faux si A est Vrai et B $\neq$ Vrai, $A \rightarrow B$ est Vrai sinon.

La formule $P \rightarrow Q$ n'est pas logiquement équivalente à la formule $\neg P \vee Q$, par exemple si P et Q ont la valeur de vérité I.

Restreint à l'ensemble {V, F}, ce connecteur coïncide avec $\Rightarrow$, le connecteur d'implication usuel.

Ce connecteur d'implication n'est pas monotone, dans les deux sens suivants:

- si P a la valeur I dans une interprétation i et la valeur V dans j et Q la valeur I dans i et j, $P \rightarrow Q$ a la valeur V dans i, et la valeur F dans j alors que $i \subseteq j$.
- ou encore, on ne peut pas étendre la propriété de monotonie aux formules contenant ce connecteur d'implication $\rightarrow$.

Puisque la formule $P \rightarrow Q$ n'est pas logiquement équivalente à la formule $\neg Q \rightarrow \neg P$, nous aurons deux connecteurs d'équivalence associés à ce connecteur d'implication $\rightarrow$.

- le connecteur d'équivalence forte $\leftrightarrow$,tel que la formule $A \leftrightarrow B$ soit logiquement équivalente à la formule:

$$(A \rightarrow B) \wedge (B \rightarrow A) \wedge (\neg A \rightarrow \neg B) \wedge (\neg B \rightarrow \neg A)$$

- le connecteur d'équivalence faible $\leftarrow\rightarrow$,tel que la formule $A \leftarrow\rightarrow B$ soit logiquement équivalente à la formule:

$$(A \rightarrow B) \wedge (B \rightarrow A)$$

L'introduction de ce connecteur d'implication et l'extension de la propriété de monotonie aux formules comportant les symboles $\exists$, $\forall$, $\wedge$, $\vee$, $\neg$, $\Rightarrow$, $\Leftrightarrow$, nous permettent d'obtenir la propriété **d'intersection des modèles** pour les programmes consistants utilisant la négation: dans le cas d'un programme **consistant**, une intersection de modèles de Herbrand de ce programme est encore un modèle de ce programme.

Bien que l'on ne fasse pas de restrictions sur les programmes étudiés, les programmes ayant des règles positives jouent un rôle privilégié car ils admettent toujours un modèle. Dès que l'on autorise la négation en tête de règle, on peut avoir des programmes inconsistants:

par exemple, le programme Pr suivant:

$$\begin{array}{l} p(x) \leftarrow r(x) \\ \neg p(x) \leftarrow q(x) \\ q(a) \leftarrow \\ r(x) \leftarrow \end{array}$$

Pour tenir compte aussi des programmes inconsistants, nous introduirons un élément Contra (Contradiction) que nous ajouterons à l'ensemble IHT(L) des interprétations de Herbrand trivaluées.

Avec ce connecteur d'implication, on retrouve le fait que les axiomes qui sont des tautologies en logique classique sont des tautologies en logique trivaluée.

Par exemple, $A \rightarrow A$ est une tautologie alors qu'elle ne l'est pas avec le connecteur $\Rightarrow$ habituel de Kleene au lieu de notre connecteur d'implication non monotone $\rightarrow$.

On obtiendra de bonnes propriétés de complétude pour cette logique comme nous le verrons au chapitre 4. Cela permettra aussi d'affirmer que l'ensemble des tautologies obtenues dans chaque cas est récursivement énumérable.

Avec ce connecteur d'implication et les connecteurs $\neg$ et $\wedge$, on retrouve le connecteur d'équivalence de Fitting $\cong$:

$A \cong B$ est logiquement équivalent à:

$$(A \rightarrow B)\wedge(B \rightarrow A)\wedge(\neg A \rightarrow \neg B)\wedge(\neg B \rightarrow \neg A) \text{ (E)}$$

Le connecteur de Fitting correspond ici à notre connecteur d'équivalence forte $\leftrightarrow$.

Nous généralisons ensuite l'opérateur "conséquence" introduit par Kowalski et Van Emden [43], en tenant compte des programmes inconsistants:

Si Contra est l'élément que l'on rajoute à l'ensemble IHT(L) des interprétations de Herbrand trivaluées pour le cas des programmes inconsistants, tel que $\forall i \in$ IHT(L), $i \subseteq$ Contra, l'opérateur introduit ici est le suivant:

T_{Pr}: IHT(L) $\cup$ {Contra} $\rightarrow$ IHT(L) $\cup$ {Contra}

$T_{Pr}(i)$ = {lit/ il existe une instance close lit $\leftarrow$ for de Pr telle que vv_i(for) = V}

si cet ensemble ne contient pas un atome et sa négation,

$T_{Pr}(i)$ = Contra sinon.

L'extension de la propriété de monotonie aux formules constituant le corps des règles des programmes étudiés, c'est-à-dire comportant les symboles $\exists$, $\forall$, $\wedge$, $\vee$, $\neg$, $\Rightarrow$, $\Leftrightarrow$, nous permet d'obtenir la monotonie de cet opérateur qui admet par conséquent un plus petit point fixe.

Dans le cas d'un programme défini, T_{Pr} est continu et pppf(T_{Pr}) = $T_{Pr}\uparrow\omega$.

De plus, on démontre l'équivalence suivante:

Pr est consistant $\Leftrightarrow$ pppf(T_{Pr}) $\neq$ Contra

Dans le cas d'un programme consistant, ses modèles sont les post-points fixes de T_{Pr}, et son plus petit modèle ou encore l'intersection de tous les modèles de Herbrand coïncide avec le plus petit point fixe de T_{Pr} (toujours en utilisant le fait que T_{Pr} est monotone, on montre que $T_{Pr}\uparrow\alpha \subseteq i$, pour tout i modèle de Pr et pour tout ordinal α).

A partir de cet opérateur, nous pouvons retrouver celui de **Fitting** [9] en complétant notre programme.

Il y a deux sortes de complétion: la **$\neg$-complétion** et la **$\rightarrow$-complétion**.
La complétion d'un programme au sens de Fitting s'obtient en faisant tout d'abord la $\neg$-complétion de Pr et ensuite la $\rightarrow$-complétion du $\neg$-complété de Pr à cause de l'équivalence logique: $A \cong B$ est logiquement équivalent à:

$$(A \rightarrow B)\wedge(B \rightarrow A)\wedge(\neg A \rightarrow \neg B)\wedge(\neg B \rightarrow \neg A)$$

On obtient, si Comp($\neg$, Pr) désigne le $\neg$-complété de Pr,

$$T_{Comp(\neg,\, Pr)} = F_{Pr}$$

D'autre part quand Pr est un programme sans négation (ayant des règles ato $\leftarrow$ for où ato est un atome et for est une formule comportant les symboles $\exists$, $\forall$, $\wedge$, $\vee$), nous retrouvons l'opérateur de Kowalski et Van Emden [43], avec l'application pos:

$$B_{Pr} = pos^{-1} o T_{Pr} o pos$$

L'ensemble des interprétations de Herbrand trivaluées est en bijection avec l'ensemble des fonctions de her(L) (l'ensemble des atomes clos de L) dans {V, F, I} ou encore des fonctions partielles de her(L) dans {V, F} prenant la valeur I là où elles ne sont pas définies. Par conséquent la théorie de Manna et Shamir [28], pour une fonctionnelle monotone définie sur un ensemble de fonctions partielles, s'adapte à notre opérateur T_{Pr}. Les interprétations de Herbrand trivaluées ne forment plus un treillis complet, puisque l'union de deux interprétations de Herbrand n'est plus nécessairement une interprétation de Herbrand. Il n'y a donc plus de plus grand point fixe de T_{Pr}, mais une série de points fixes maximaux. Le plus grand point fixe de T_{Pr}, inclus dans tous ces points fixes maximaux est le point fixe optimal de T_{Pr}, opt(T_{Pr}) (Manna et Shamir, [27], [28]).

De même il n'y a pas de plus grand modèle de Pr (de plus grand post-point fixe de T_{Pr}) mais plusieurs **modèles maximaux**. L'intersection de tous ces modèles maximaux est encore un modèle de Pr et constitue le modèle optimal opt(Pr).

Ce modèle optimal définit une autre sémantique pour les programmes logiques avec négation que celle du plus petit point fixe. Il peut être plus intéressant parce qu'il contient plus d'informations puisque pppf(T_{Pr}) $\subseteq$ opt(Pr).

On introduit alors la notion d'interprétations **consistantes** (justifiée parce que les interprétations de Herbrand trivaluées sont des interprétations partielles).
L'ensemble des modèles consistants d'un programme Pr admet un plus grand élément qui coïncide avec le modèle optimal de Pr.

1.5 PRESENTATION GENERALE

Comme nous l'avons exposé dans l'introduction, nous ne nous limitons toutefois pas aux applications directes de cette logique trivaluée à la programmation logique.

- Le chapitre 2 introduit le formalisme utilisé sans se limiter aux interprétations de Herbrand ni aux règles de programmes.

Nous nous plaçons successivement dans le cadre du calcul propositionnel et du calcul des prédicats afin de reprendre successivement ces deux formalismes au chapitre 4 pour établir des théorèmes de complétude sur cette logique.

Dans le cadre du calcul des prédicats, on reprend le formalisme utilisé dans Yasuhara [45], en particulier la notion d'interprétation.
Les deux définitions importantes de ce chapitre sont les suivantes:

La valeur de vérité d'une formule est définie par la valeur de vérité de sa clôture universelle. Il ne suffit plus de dire qu'une formule est vraie si et seulement si sa clôture universelle est vraie puisque cette formule peut prendre la valeur de vérité Indéterminé.

Deux formules A et B sont **logiquement équivalentes** si et seulement si la formule $A \leftrightarrow B$ est une tautologie; ce qui est plus fort que de dire que leurs clôtures universelles ont la même valeur de vérité dans nimporte quelle interprétation sauf si A et B sont des formules closes. C'est la même chose en logique classique avec le connecteur d'équivalence usuel $\Leftrightarrow$.

Dans ce chapitre, nous étudions les relations entre les modèles trivalués forts, faibles et bivalués.

Dans une dernière partie, nous établissons un théorème de Herbrand trivalué:

Un ensemble de clauses a un modèle de Herbrand si et seulement si il admet un modèle de Herbrand trivalué.
La démonstration de ce théorème étant basée sur le fait qu'à une interprétation trivaluée i correspond une interprétation de Herbrand H, dans laquelle la valeur de vérité de tout atome clos $p(t_1, ..., t_n)$ est donnée par:

$$vv_H(p(t_1, ..., t_n)) = vv_i(p(t_1, ..., t_n)).$$

- Le troisième chapitre est un chapitre essentiellement combinatoire, qui étudie l'expressivité des différents connecteurs.

Nous démontrons tout d'abord un théorème général pour exprimer nimporte quelle fonction de $\{V, F, I\}^n$ dans $\{V, F, I\}$ à l'aide d'un système minimal de connecteurs $(\neg, \rightarrow, I, \wedge)$.

Puis un second théorème caractérise les fonctions f de $\{V, F, I\}^n$ dans $\{V, F, I\}$, monotones, telles que $f(\{V, F\}^n) \subseteq \{V, F\}$ et $f(I, ..., I) = I$ par le fait qu'elles ne s'expriment qu'avec des $\neg$ et des $\vee$.
La démonstration est constructive et donne un algorithme pour construire la fonction ne s'exprimant qu'avec des $\neg$ et des $\vee$ à partir de celle de départ vérifiant les trois propriétés énoncées.
L'idée de la démonstration est la suivante:
à partir de f, on construit une fonction $\mathcal{f}$ qui vérifie les trois propriétés suivantes:

1) $\mathcal{f}$ ne s'exprime qu'avec des $\neg$, des $\wedge$ et des $\vee$.
2) $\mathcal{f}/\{V, F\}^n = f/\{V, F\}^n$
3) $\mathcal{f}$ est monotone sur $\{V, F, I\}^n$ (résulte de 1)) et $\mathcal{f}$ est maximale sur $\{V, F, I\}^n$

Le fait que $\mathcal{f}$ soit maximale permet ensuite d'obtenir f à partir de $\mathcal{f}$ en envisageant quatre cas:

- Cas 1) $f = \mathcal{f}$

- Cas 2) $\exists(a_1, ..., a_n) \in \{V, F, I\}^n / f(a_1, ..., a_n) = I$, et $\mathcal{f}(a_1, ..., a_n) = V$
  $\forall(b_1, ..., b_n) \in \{V, F, I\}^n / \mathcal{f}(b_1, ..., b_n) = F$ implique $f(b_1, ..., b_n) = F$

- Cas 3) $\exists(a_1, ..., a_n) \in \{V, F, I\}^n / f(a_1, ..., a_n) = I$, et $\mathcal{f}(a_1, ..., a_n) = F$
  $\forall(b_1, ..., b_n) \in \{V, F, I\}^n / \mathcal{f}(b_1, ..., b_n) = V$ implique $f(b_1, ..., b_n) = V$

- Cas 4) $\exists(a_1, ..., a_n) \in \{V, F, I\}^n / f(a_1, ..., a_n) = I$, et $\mathcal{f}(a_1, ..., a_n) = F$
  $\exists(b_1, ..., b_n) \in \{V, F, I\}^n / f(b_1, ..., b_n) = I$ et $\mathcal{f}(b_1, ..., b_n) = V$

Ce sont les seuls cas à envisager car $\mathcal{f}$ est maximale.

Pour les n-uplets $(a_1, ..., a_n) \in \{V, F, I\}^n / f(a_1, ..., a_n) = I$, et $\mathcal{f}(a_1, ..., a_n) = V$, on construit des fonctions $V(a_1, ..., a_n)$ avec les connecteurs $\neg$ et $\vee$, telles que:

$(c_1, ..., c_n) \leq (a_1, ..., a_n)$ implique $V(a_1, ..., a_n)(c_1, ..., c_n) = I$
$\text{non}((c_1, ..., c_n) \leq (a_1, ..., a_n))$ implique $V(a_1, ..., a_n)(c_1, ..., c_n) = V$.

Pour les n-uplets $(b_1, ..., b_n) \in \{V, F, I\}^n / f(b_1, ..., b_n) = I$, et $\mathcal{f}(b_1, ..., b_n) = F$, on construit des fonctions $F(b_1, ..., b_n)$ avec les connecteurs $\neg$ et $\wedge$, telles que:

$(c_1, ..., c_n) \leq (b_1, ..., b_n)$ implique $F(b_1, ..., b_n)(c_1, ..., c_n) = I$
$\text{non}((c_1, ..., c_n) \leq (b_1, ..., b_n))$ implique $F(b_1, ..., b_n)(c_1, ..., c_n) = F$.

On construit ensuite f à partir de f, $V(a_1, ..., a_n)$, $F(b_1, ..., b_n)$ dans chacun des cas.

- Dans le chapitre 4, on étudie la **complétude** de cette logique.

C'est-à-dire une équivalence entre une propriété syntaxique et une propriété sémantique:

$$\vdash A \Leftrightarrow \models_T A$$

L'expression $\vdash A$ signifie que A est une formule obtenue par déduction à partir d'un système formel comportant des axiomes et une ou plusieurs règles d'inférence.

Nous obtenons deux théorèmes de complétude en calcul propositionnel:

pour le cas où A est une formule comprenant les connecteurs $\rightarrow$ et $\neg$, et le cas où A est une formule comprenant les connecteurs $\rightarrow$, $\wedge$ et $\neg$.

Nous obtenons un théorème de complétude en calcul des prédicats:

pour le cas où A est une formule comprenant les connecteurs $\rightarrow$ et $\neg$, et le quantificateur $\exists$.

En calcul propositionnel, on introduit dans le cas où A comprend les symboles $\rightarrow$ et $\neg$ (resp. les symboles $\rightarrow$, $\wedge$ et $\neg$) la formule:

$$\text{Ind}_1(A) = \neg((A \rightarrow \neg A) \rightarrow \neg(\neg A \rightarrow A))$$

(resp. la formule:

$$\text{Ind}_2(A) = (\neg A \rightarrow A) \wedge (A \rightarrow \neg A)).$$

Ces deux formules ont la valeur Vrai si A a la valeur I, Faux sinon. L'introduction de ces deux formules paraît naturelle comme l'introduction des formules Fx(A) (Fx(A) prend la valeur Vrai si et seulement si A a la valeur Faux, Faux sinon) et Vr(A) (Vr(A) prend la valeur Vrai si et seulement si A a la valeur Vrai, Faux sinon) en logique classique.

On introduit aussi une interprétation v telle que:

$A^v = A$ si $vv_v(A) = V$
$A^v = \neg A$ si $vv_v(A) = F$
$A^v = Ind_1(A)$ (resp. $Ind_2(A)$) si $vv_v(A) = I$.

C'est l'introduction de ces deux formules $Ind_1(A)$ et $Ind_2(A)$ qui diffère de la démonstration en logique classique.

Pour démontrer que:

$\vdash A \Rightarrow \models_T A$, on raisonne par induction sur la longueur de la preuve dans $\vdash A$ en vérifiant que chaque axiome est une tautologie et en utilisant le fait que la règle d'inférence utilisée est le modus ponens.

Pour démontrer que:

$\models_T A \Rightarrow \vdash A$, on commence par montrer que si A est une formule comprenant les atomes $p_1, ..., p_n$, on a: $p_1^v, ..., p_n^v \vdash A^v$, par induction sur A.
On envisage les cas: A = p atome, $A = \neg B$, $A = B \rightarrow C$ et dans le deuxième cas $A = B \wedge C$.

On en déduit alors, comme A est une tautologie: $p_1^v, ..., p_n^v \vdash A$, ce qui permet alors de démontrer par récurrence descendante sur k, que $p_1^v, ..., p_k^v \vdash A$, pour $1 \leq k \leq n$. Dans chaque cas l'axiome:

$(p \rightarrow A) \rightarrow ((\neg p \rightarrow A) \rightarrow ((Ind_1(p)$ (resp. $Ind_2(p)) \rightarrow A) \rightarrow A)$, permet de conclure $\vdash A$.

En calcul des prédicats, la démonstration adapte la démonstration du théorème de Gödel, dont on trouve une version dans Yasuhara [45].

La partie $\vdash A \Rightarrow \models_T A$ du théorème ne pose pas de problèmes et se démontre par induction sur la longueur de la preuve dans $\vdash A$ en vérifiant que chaque axiome est une tautologie et en utilisant le fait que les règles d'inférence utilisées sont le modus ponens et la généralisation universelle.

Pour démontrer que $\models_T A \Rightarrow \vdash A$, on introduit la formule:
$Ind_1(A) = \neg((A \rightarrow \neg A) \rightarrow \neg(\neg A \rightarrow A))$, qui prend la valeur V si et seulement si A a la valeur I, F sinon.

On montre que si A est une formule close, $\models_T A \Rightarrow \vdash A$, en montrant la contraposée.

On généralise ensuite cela à n'importe quelle formule $A(x_1, ..., x_n)$ ayant $x_1, ..., x_n$ pour variables libres:

$\models_T A(x_1, ..., x_n)$ implique $\models_T \forall x_1 ... \forall x_n A(x_1, ..., x_n)$ (par définition d'une tautologie)
$\models_T \forall x_1 ... \forall x_n A(x_1, ..., x_n)$ implique $\vdash \forall x_1 ... \forall x_n A(x_1, ..., x_n)$ (car $\forall x_1 ... \forall x_n A(x_1, ..., x_n)$ est une formule close)
$\vdash \forall x_1 ... \forall x_n A(x_1, ..., x_n)$ implique $\vdash A(y_1, ..., y_n)$ par un axiome et le théorème de déduction
$\vdash A(y_1, ..., y_n)$ implique $\vdash A(x_1, ..., x_n)$ en renommant les variables de la preuve.

Pour montrer que si A est une formule close non prouvable (on n'a pas: $\vdash$ A) alors A n'est pas une tautologie, on définit les notions de théorie (ensemble clos par déduction, contenant des axiomes donnés au départ) complète et consistante:

Une théorie $\mathcal{T}$ est **complète** si pour toute formule close A,

$$A \in \mathcal{T}, \neg A \in \mathcal{T} \text{ ou } Ind_I(A) \in \mathcal{T}.$$

Une théorie $\mathcal{T}$ est **consistante** si pour toute formule A,

on n'a pas $A \in \mathcal{T}$ et $\neg A \in \mathcal{T}$
ou $Ind_I(A) \in \mathcal{T}$ et $\neg A \in \mathcal{T}$
ou $Ind_I(A) \in \mathcal{T}$ et $A \in \mathcal{T}$.

Ces deux définitions généralisent les définitions classiques en y ajoutant la formule $Ind_I(A)$.

On montre que si A est une formule close non prouvable, (on n'a pas: $\vdash$ A), alors:

$\mathcal{T} \cup \{\neg A\}$ ou $\mathcal{T} \cup \{Ind_I(A)\}$ est une théorie consistante.

Si on démontre qu'une théorie consistante a un modèle, cela entraînera que A n'est pas une tautologie puisque $\neg A$ ou $Ind_I(A)$ auront un modèle, ce qui signifie que A aura la valeur F ou la valeur I dans ce modèle.

La partie essentielle du théorème consiste à démontrer qu'une théorie consistante a un modèle.

Pour cela, on rajoute des constantes au langage $\{s_1, ..., s_n, ...\}$, puis on énumère les formules closes de L commençant par un quantificateur existentiel: $\exists x_{i_j} B_j(x_{i_j})$ et l'on rajoute à la nouvelle théorie obtenue, les formules $\exists x_{i_j} B_j(x_{i_j}) \rightarrow B_j(s_{i_j})$ et en plus du cas classique les formules: $Ind_1(\exists x_{i_j} B_j(x_{i_j})) \rightarrow Ind_1(B_j(s_{i_j}))$.

On construit ensuite une extension complète et consistante $\mathcal{T}^{**}$ de cette nouvelle théorie et on montre qu'elle admet un modèle de Herbrand défini par:

$$vv_H(p(t_1, \ldots, t_n)) = V \text{ si et ssi } \vdash_{\mathcal{T}^{**}} p(t_1, \ldots, t_n)$$
$$vv_H(p(t_1, \ldots, t_n)) = F \text{ si et ssi } \vdash_{\mathcal{T}^{**}} \neg p(t_1, \ldots, t_n)$$
$$vv_H(p(t_1, \ldots, t_n)) = I \text{ si et ssi } \vdash_{\mathcal{T}^{**}} Ind_1(p(t_1, \ldots, t_n))$$

Comme $\mathcal{T} \subseteq \mathcal{T}^{**}$, $\mathcal{T}$ a un modèle.

Ces théorèmes de complétude permettent d'ailleurs d'affirmer aussi que l'ensemble des tautologies obtenues dans chacun des cas est récursivement énumérable.

- Les chapitres 5 et 6 constituent l'application de cette logique à la programmation logique. Comme nous les avons introduits dans le paragraphe précédent, nous les présentons rapidement dans ce paragraphe.

- Dans le chapitre 5, on démontre la propriété d'intersection des modèles, on introduit l'opérateur "conséquence" associé aux programmes et on fait le lien avec l'opérateur introduit par Fitting pour le cas des programmes ayant des règles positives et l'opérateur de Van Emden et Kowalski pour le cas des programmes sans négation.

- Dans le chapitre 6, on étudie spécifiquement les modèles optimaux dont l'existence provient de cette logique à trois valeurs de vérité.
Ces modèles sont intéressants parce qu'ils contiennent plus d'informations que le pppf de l'opérateur mais le problème résidera dans leur calculabilité.

CHAPITRE 2: FORMALISME

Dans un souci à la fois de rigueur et de généralisation, nous donnons ici un cadre général d'étude pour la logique trivaluée sans se limiter aux interprétations de Herbrand qui seront le cadre d'étude des programmes avec négation ni aux formules qui sont des règles de programmes.

Pour cela, il faut distinguer le calcul propositionnel du calcul des prédicats.

Dans un premier paragraphe, nous exposons le formalisme du calcul propositionnel sans variables et sans quantificateurs. Ensuite, dans un second paragraphe on introduit le formalisme du calcul des prédicats, en utilisant la notion d'interprétation définie dans Yasuhara [45] généralisée à la notion d'interprétation trivaluée et en définissant la valeur de vérité d'une formule par la valeur de vérité de sa clôture universelle. Enfin dans l'avant-dernier paragraphe, on introduit un cas particulier d'interprétation trivaluée: les interprétations de Herbrand trivaluées qui joueront un rôle privilégié dans les programmes logiques avec négation.

Dans le dernier paragraphe, on établit l'équivalence entre la consistance au sens de Herbrand et la consistance, pour un ensemble de clauses (en fait des formules universellement quantifiées).

2.1 CALCUL PROPOSITIONNEL

2.1.1 Syntaxe.

Définition 2.1.1

Soit $\mathcal{P}$ un ensemble fini ou infiniment dénombrable de variables propositionnelles (atomes).

Soit $\mathcal{A} = \{(,)\}$ l'ensemble des symboles de ponctuation.

Soit $\mathbf{C} = \{\neg, \wedge, \vee, \leftrightarrow, \Leftrightarrow, \Rightarrow, \leftarrow\rightarrow, \rightarrow\}$ l'ensemble des connecteurs.

L'ensemble $\mathcal{F}(\mathcal{P})$, dont les atomes sont dans $\mathcal{P}$, est défini inductivement comme suit:

$\mathcal{P} \subset \mathcal{F}(\mathcal{P})$.

$A \in \mathcal{F}(\mathcal{P})$ implique $\neg A \in \mathcal{F}(\mathcal{P})$.

$A, B \in \mathcal{F}(\mathcal{P})$ implique $(A \gamma B) \in \mathcal{F}(\mathcal{P})$ si $\gamma \in \{\wedge, \vee, \leftrightarrow, \Leftrightarrow, \Rightarrow, \leftarrow\rightarrow, \rightarrow\}$.

Remarques: $\mathcal{F}(\mathcal{P})$ peut se définir par un procédé constructif:

$F_0 = \mathcal{P}$

$F_{n+1} = F_n \cup \{\neg A / A \in F_n\} \cup \{(A \gamma B) / A, B \in F_n$ et $\gamma \in \{\wedge, \vee, \leftrightarrow, \Leftrightarrow, \Rightarrow, \leftarrow\rightarrow, \rightarrow\}\}$.

$\mathcal{F}(\mathcal{P}) = \cup_{n \in \mathcal{N}} F_n$.

$\mathcal{N}$ désigne l'ensemble des entiers naturels.

$\rightarrow$ désigne le connecteur d'implication non monotone introduit ici et $\leftrightarrow$ le connecteur d'équivalence forte et $\leftarrow\rightarrow$ le connecteur d'équivalence faible qui lui sont associés.

$\Rightarrow$ désigne le connecteur d'implication de Kleene habituel et $\Leftrightarrow$, le connecteur d'équivalence qui lui est associé.

Définition 2.1.2 Le plus petit entier n tel que $A \in F_n$ s'appelle la complexité de A.

Ce n'est pas toujours le nombre de connecteurs de A. Si p, q, r, s sont des atomes, $(p \rightarrow q) \wedge (\neg r \vee s) \in F_3$ alors qu'elle possède 4 connecteurs.

Remarque: On définit maintenant l'opération de substitution d'une formule à un atome dans une formule donnée:

Définition 2.1.3 Si $A \in \mathcal{F}(\mathcal{P})$, $p \in \mathcal{P}$ et $s \in \mathcal{F}(\mathcal{P})$, $A_p{}^s$, la formule obtenue en substituant dans A les occurrences de p par s se définit inductivement de la manière suivante:

- Si $A \in F_0$ et $A = p$, alors $A_p{}^s = s$; si $A = q \neq p$, alors $A_p{}^s = q$.
- Si $A \in F_n$, $n > 0$, et $A = \neg B$, alors $A_p{}^s = \neg B_p{}^s$;

si $A = (B \gamma C)$, alors $A_p{}^s = (B_p{}^s \gamma C_p{}^s)$.

On vérifie immédiatement que $A_p{}^s$ est un élément de $\mathcal{F}(\mathcal{P})$.

Nous allons maintenant définir la valeur de vérité d'une formule en calcul propositionnel en logique trivaluée.

Définitions 2.1.4 Une interprétation trivaluée i est un sous ensemble de $\mathcal{P} \cup \neg\mathcal{P}$ tel que:

$$p \in i \text{ implique } \neg p \notin i.$$

Une interprétation bivaluée i est une interprétation trivaluée telle que:

$$p \in i \text{ ou } \neg p \in i.$$

La valeur de vérité d'une formule relativement à i se définit de la manière suivante:
C'est une application vv_i de $\mathcal{F}(\mathcal{P})$ dans $\{V, F, I\}$ définie par:

-Si $A \in F_0$, et $A = p$, alors si $p \in i$, $vv_i(A) =$ Vrai (V), si $\neg p \in i$, $vv_i(A) =$ Faux (F), $vv_i(A) =$ Indéterminé (I) sinon.

N.B. On emploiera indifféremment V (resp. F, I) ou Vrai (resp. Faux, Indéterminé)

-Si $A \in F_n$, $n > 0$, la valeur de vérité de A est donnée par les tables de vérité suivantes:

$\vee$	V	F	I
V	V	V	V
F	V	F	I
I	V	I	I

$\wedge$	V	F	I
V	V	F	I
F	F	F	F
I	I	F	I

$\Rightarrow$	V	F	I
V	V	F	I
F	V	V	V
I	V	I	I

$\Leftrightarrow$	V	F	I
V	V	F	I
F	F	V	I
I	I	I	I

$\leftrightarrow$	V	F	I
V	V	F	F
F	F	V	F
I	F	F	V

$\leftarrow\rightarrow$	V	F	I
V	V	F	F
F	F	V	V
I	F	V	V

	$\neg$
V	F
F	V
I	I

$\rightarrow$	V	F	I
V	V	F	F
F	V	V	V
I	V	V	V

Remarque: On voit sur ces tables de vérité que:

$A \leftrightarrow B$ a la même valeur de vérité que

$(A \rightarrow B) \wedge (B \rightarrow A) \wedge (\neg A \rightarrow \neg B) \wedge (\neg B \rightarrow \neg A)$

$A \leftarrow\rightarrow B$ a la même valeur de vérité que

$(A \rightarrow B) \wedge (B \rightarrow A)$

$A \Leftrightarrow B$ a la même valeur de vérité que

$(A \Rightarrow B) \wedge (B \Rightarrow A)$

La valeur de vérité de A se définit en effet inductivement de la manière suivante:

> Si $A = \neg B$, $vv_i(A) = \neg vv_i(B)$
> et si $A = (B \gamma C)$, $\gamma \in \{\wedge, \vee, \leftrightarrow, \Leftrightarrow, \Rightarrow, \leftarrow\rightarrow, \rightarrow\}$,
> $vv_i(A) = \gamma(vv_i(B), vv_i(C))$. ($\gamma$ étant considéré comme une application de $\{V, F, I\}^2$ dans $\{V, F, I\}$ dont la table est donnée ci-dessus).

Ces tables de vérité serviront aussi en calcul des prédicats pour définir la valeur de vérité d'une formule relativement à une application sur l'ensemble des variables.

N.B. Par abus de langage, si $vv_i(A) = V$ (resp. F, resp. I), on dira que A est vraie (resp. fausse, resp. indéterminée) dans i.

2.1.2 Modèles et conséquences.

Définitions 2.1.5 Une interprétation trivaluée (resp. bivaluée) i est un modèle fort d'un sous-ensemble $\mathcal{T}$ de $\mathcal{F}(\mathcal{P})$ si $\forall A \in \mathcal{T}$, $vv_i(A) = V$.
$\mathcal{T}$ est alors fortement consistant au sens trivalué (resp. au sens bivalué).

Une interprétation trivaluée i est un modèle faible d'un sous-ensemble $\mathcal{T}$ de $\mathcal{F}(\mathcal{P})$ si $\forall A \in \mathcal{T}$, $vv_i(A) \neq F$.
$\mathcal{T}$ est alors faiblement consistant au sens trivalué.

Pour les interprétations bivaluées, les notions de modèle faible et fort coïncident:

Proposition 2.1.1 *Si i est une interprétation bivaluée, alors pour chaque formule for du langage L,*

$$vv_i(for) \in \{V, F\}.$$

Démonstration Par induction sur la complexité de for, en utilisant le fait que tous les connecteurs utilisés dans le langage laissent stable l'ensemble $\{V, F\}$.
Si for = ato, puisque i est bivaluée, alors $vv_i(for) \in \{V, F\}$.♦

Définition 2.1.6 Si Ax et Bx sont deux sous-ensembles de $\mathcal{F}(\mathcal{P})$, Bx est conséquence trivaluée forte de Ax, si et seulement si tout modèle trivalué fort de Ax est un modèle trivalué fort de Bx.
On le note:

$$Ax \models_T Bx.$$

On définit de même les notions de conséquences trivaluées faibles ($Ax \models_t Bx$) et de conséquence bivaluée ($Ax \models_B Bx$).

Définition 2.1.7 Si Ax et Bx sont deux ensembles de formules de $\mathfrak{F}(\mathfrak{P})$,

$Ax =\!\mid_T \models Bx$ (resp. $Ax =\!\mid_t \models Bx$) si et seulement si Ax et Bx ont les mêmes modèles trivalués forts (resp. faibles).

On définit $Ax =\!\mid_B \models Bx$ de la même manière.

Relations entre ces notions.

Un modèle trivalué fort de Ax est un modèle trivalué faible; par conséquent, si Ax est fortement consistant au sens trivalué, alors il est faiblement consistant au sens trivalué.

Un modèle bivalué de Ax est un modèle trivalué fort de Ax; par conséquent, si Ax est consistant au sens bivalué, alors il est fortement consistant au sens trivalué.

Remarques: - ∅ n'est pas toujours un modèle trivalué faible d'une formule. Ce n'est pas, par exemple un modèle trivalué faible de $(A \rightarrow B) \rightarrow A$, si A et B sont deux atomes.

- Un ensemble de formules ne possède pas nécessairement de modèle trivalué faible: par exemple, si A et B sont deux atomes, la formule: $((A \wedge \neg A) \rightarrow B) \rightarrow (B \wedge \neg B)$ n'a pas de modèle trivalué. Dans n'importe quelle interprétation trivaluée, $A \wedge \neg A$ est soit Indéterminé, soit Faux, et $(A \wedge \neg A) \rightarrow B$ est toujours Vrai tandis que $B \wedge \neg B$ n'est jamais Vrai.

- Aucune autre implication n'est vraie:

- Si A est un atome et $Ax = \{A, \neg A\}$, alors ∅ est un modèle trivalué faible de Ax sans en être un modèle trivalué fort. Ax est consistant au sens trivalué faible sans l'être au sens trivalué fort.

- Même un modèle trivalué fort maximal (au sens de l'inclusion) de Ax n'est pas nécessairement un modèle bivalué (alors que les interprétations trivaluées maximales sont les interprétations bivaluées).

Par exemple, soit Ax l'ensemble $\{A \rightarrow B, A \rightarrow \neg B, \neg A \rightarrow \neg B, \neg A \rightarrow B\}$ où A et B sont deux atomes. Ax est fortement consistant au sens trivalué sans l'être au sens bivalué. Les ensembles $\varnothing$, $\{B\}$ et $\{\neg B\}$ sont des modèles trivalués forts de Ax, les deux derniers étant des modèles maximaux sans être des modèles bivalués.

Cependant, on a la proposition suivante:

Proposition 2.1.2 *Si Ax est un ensemble de formules de $\mathcal{F}(\mathcal{P})$, une interprétation trivaluée i est un modèle trivalué fort de $Ax' = Ax \cup \{ato \vee \neg ato / \ ato \in \mathcal{P}\}$ si et seulement si i est une interprétation bivaluée et i est un modèle bivalué de Ax.*

Démonstration Si i est un modèle trivalué fort de $\{ato \vee \neg ato / \ ato \in \mathcal{P}\}$ alors $\forall ato \in \mathcal{P}$, $ato \in i$ ou $\neg ato \in i$, et i est une interprétation bivaluée. Réciproquement, si i est bivaluée, alors $vv_i(ato \vee \neg ato) =$ Vrai, pour tout ato de $\mathcal{P}$. ♦

- Si Ax $\models_T$ Bx, alors Ax $\models_B$ Bx.

- Si Ax $\models_t$ Bx, alors Ax $\models_B$ Bx.

Aucune autre implication n'est vraie:

- Si Ax $\models_B$ Bx, on n'a pas Ax $\models_T$ Bx.

Par exemple, prenons pour Ax l'ensemble vide et pour Bx l'ensemble $\{A \vee \neg A\}$ où A est un atome. Toute interprétation bivaluée est un modèle de Ax et de Bx alors que $\varnothing$ est un modèle trivalué fort de Ax et non de Bx.

- Si Ax $\models_B$ Bx, on n'a pas Ax $\models_t$ Bx.

Par exemple, prenons pour Ax l'ensemble $\{A, \neg A\}$ et pour Bx l'ensemble $\{B\}$ où A et B sont deux atomes. Ax ne possède pas de modèle bivalué tandis que $\{\neg B\}$ est un modèle trivalué faible de Ax et non de Bx.

- Si Ax $\models_T$ Bx, on n'a pas Ax $\models_t$ Bx.

Par exemple, prenons pour Ax l'ensemble $\{A, A \rightarrow B\}$ et pour Bx l'ensemble $\{B\}$, où A et B sont deux atomes. Le seul modèle trivalué fort de Ax est $\{A, B\}$; c'est un modèle trivalué fort de Bx alors que $\{\neg B\}$ est un modèle trivalué faible de Ax et non de Bx.

- Si Ax $\models_t$ Bx, on n'a pas Ax $\models_T$ Bx.

Par exemple, prenons pour Ax l'ensemble $\{A \leftrightarrow \neg A\}$ et pour Bx l'ensemble $\{A\}$, où A est un atome. Le seul modèle trivalué faible de Ax est $\emptyset$ qui est un modèle trivalué faible de Bx. Le seul modèle trivalué fort de Ax est $\emptyset$ qui n'est pas un modèle trivalué fort de Bx.

Dans la suite de ce paragraphe, on ne considèrera que des modèles trivalués forts et des modèles bivalués.

2.1.3 Tautologies et équivalences trivaluées.

Définitions 2.1.7 Une formule A de $\mathcal{F}(\mathcal{P})$ est une tautologie trivaluée (resp. bivaluée) si A est vraie dans toute interprétation trivaluée (resp. bivaluée) i. On le note $\models_T A$ (resp. $\models_B A$).

Remarque: On n'a pas: $\models_T (A \Rightarrow A)$.
Par contre, on a bien: $\models_T (A \rightarrow A)$.
Cette propriété présente un des intérêts du nouveau symbole d'implication

Lemme 2.1.1 *Si A et s sont des formules et p est une variable propositionnelle, et si A est une tautologie alors $A_p{}^s$ est une tautologie.*

Démonstration Il faut montrer que $vv_i(A_p{}^s)$ = Vrai, pour toute interprétation i. Or $vv_i(A_p{}^s) = vv_{i'}(A)$, avec i' définie par $vv_{i'}(q) = vv_i(q)$ si q est une variable propositionnelle différente de p, $vv_{i'}(p) = vv_i(s)$ (par induction sur A). ♦

Lemme 2.1.1 *On a l'équivalence suivante:*

$\models_T A \leftrightarrow B$ *équivaut à :*

Pour toute interprétation trivaluée i, $vv_i(A) = vv_i(B)$.

Remarque: On a l'équivalent en bivalué:

$\models_B A \Leftrightarrow B$ *équivaut à :*

Pour toute interprétation bivaluée i, $vv_i(A) = vv_i(B)$.

Définition 2.1.8 Deux formules A et B sont logiquement équivalentes si elles vérifient l'une des conditions du lemme 2.1.1.

On le note: $A \equiv_T B$.

On définit de même: $A \equiv_B B$.

Voici des exemples d'équivalences trivaluées:

Exemples 2.1.1

- $A \Rightarrow B \equiv_T \neg B \Rightarrow \neg A$
- $A \Leftrightarrow B \equiv_T (A \Rightarrow B) \wedge (B \Rightarrow A)$
- $A \Rightarrow B \equiv_T \neg A \vee B$
- $A \vee (B \wedge C) \equiv_T (A \vee B) \wedge (A \vee C)$
- $A \wedge (B \vee C) \equiv_T (A \wedge B) \vee (A \wedge C)$
- $\neg\neg A \equiv_T A$
- $A \vee (B \vee C) \equiv_T (A \vee B) \vee C$
- $A \wedge (B \wedge C) \equiv_T (A \wedge B) \wedge C$
- $\neg(A \vee B) \equiv_T \neg A \wedge \neg B$
- $\neg(A \wedge B) \equiv_T \neg A \vee \neg B$
- $(A \vee B) \rightarrow C \equiv_T (A \rightarrow C) \wedge (B \rightarrow C)$
- $(A \wedge B) \rightarrow C \equiv_T A \rightarrow (B \rightarrow C)$
- $A \rightarrow (B \wedge C) \equiv_T (A \rightarrow B) \wedge (A \rightarrow C)$
- $A \rightarrow (B \vee C) \equiv_T (A \rightarrow B) \vee (A \rightarrow C)$

- Par contre, on n'a pas: $A \rightarrow B \equiv_T \neg B \rightarrow \neg A$.

(par exemple, si l'on prend i telle que: $vv_i(B) = I$ et $vv_i(A) = V$).

- Alors que l'on a bien, dans le cas bivalué: $A \Rightarrow B \equiv_B \neg B \Rightarrow \neg A$.

- C'est l'une des raisons de l'introduction de deux symboles d'équivalence: $\leftrightarrow$ et $\leftarrow\rightarrow$. L'unique connecteur d'équivalence $\Leftrightarrow$ se scinde en deux.

En effet:

- $A \leftrightarrow B \equiv_T (A \rightarrow B) \wedge (B \rightarrow A) \wedge (\neg A \rightarrow \neg B) \wedge (\neg B \rightarrow \neg A)$.

- $A \leftarrow\rightarrow B \equiv_T (A \rightarrow B) \wedge (B \rightarrow A)$.

Les trois propositions suivantes établissent des liens entre les notions de conséquences et de tautologies, celles de conséquence et de consistance, et enfin d'équivalence et de tautologie.

Proposition 2.1.3 *Soient $A_1, A_2, \ldots, A_n, B_1, B_2, \ldots, B_m$ des formules de $\mathcal{F}(\mathcal{P})$.*

Toutes les assertions suivantes sont équivalentes:

(1) $\{A_1, A_2, ..., A_n\} \models_T \{B_1, B_2, ..., B_m\}$
(2) $\{A_1 \wedge A_2 \wedge ... \wedge A_n\} \models_T \{B_1, B_2, ..., B_m\}$
(3) $\{A_1, A_2, ..., A_n\} \models_T \{B_1 \wedge B_2 \wedge ... \wedge B_m\}$
(4) $\{A_1 \wedge A_2 \wedge ... \wedge A_n\} \models_T \{B_1 \wedge B_2 \wedge ... \wedge B_m\}$
(5) $\{A_1, A_2, ..., A_{n-1}\} \models_T \{A_n \rightarrow B_1, A_n \rightarrow B_2, ..., A_n \rightarrow B_m\}$
(6) $\{A_1 \wedge A_2 \wedge ... \wedge A_{n-1}\} \models_T \{A_n \rightarrow (B_1 \wedge B_2 \wedge ... \wedge B_m)\}$
(7) $\models_T (A_1 \wedge A_2 \wedge ... \wedge A_n) \rightarrow (B_1 \wedge B_2 \wedge ... \wedge B_m)$

Démonstration (1)$\Leftrightarrow$(2)$\Leftrightarrow$(3)$\Leftrightarrow$(4) car i est un modèle de $\{A_1, A_2, ..., A_n\}$ si et seulement si i est un modèle de $A_1 \wedge A_2 \wedge ... \wedge A_n$. On a, de plus, si Δ est un sous-ensemble de $\mathcal{F}(\mathcal{P})$, $\Delta \cup \{A\} \models_T \{B\} \Leftrightarrow \Delta \models_T A \rightarrow B$. Donc, (3)$\Leftrightarrow$(6) et (4)$\Leftrightarrow$(7) Enfin, on a:
$A_n \rightarrow (B_1 \wedge B_2 \wedge ... \wedge B_m) \equiv_T A_n \rightarrow B_1 \wedge A_n \rightarrow B_2 \wedge ... \wedge A_n \rightarrow B_m$, donc (6)$\Leftrightarrow$(5). ♦

Remarque: Cette proposition n'est plus vraie si l'on remplace le connecteur $\rightarrow$ par le connecteur $\Rightarrow$. On n'a plus $\Delta \cup \{A\} \models_T \{B\} \Leftrightarrow \Delta \models_T (A \Rightarrow B)$. En effet, $A \Rightarrow A$ n'est plus une tautologie, aussi n'a-t-on plus $\models_T (A \Rightarrow A)$.

Proposition 2.1.4 *Soient At un élément de $\mathcal{P}$ et Ax un ensemble de formules de $\mathcal{F}(\mathcal{P})$.*

On a l'équivalence suivante:

(1) Ax $\cup$ $\{\neg At\}$ n'est pas fortement consistant au sens trivalué si et seulement si
(2) $\neg$At n'appartient à aucun modèle trivalué fort de Ax.

Cependant, on n'a pas, comme dans le cas bivalué:

Ax $\cup$ $\{\neg At\}$ n'est pas fortement consistant au sens trivalué si et seulement si
At est une conséquence logique trivaluée de Ax.

Démonstration (1)$\Leftrightarrow$(2) est clair.

At est une conséquence logique trivaluée de Ax implique que Ax $\cup$ $\{\neg At\}$ n'est pas fortement consistant au sens trivalué.

La réciproque est fausse comme le montre l'exemple suivant:

Soit Ax l'ensemble:

$\{A \rightarrow B,\ A \rightarrow \neg B,\ \neg A \rightarrow B,\ \neg A \rightarrow \neg B\}$; où A et B sont deux atomes.

$Ax \cup \{\neg A\}$ n'est pas fortement consistant au sens trivalué alors que l'on n'a pas $Ax \models_T A$, puisque $\emptyset$ est un modèle trivalué fort de Ax et non de A. ♦

Proposition 2.1.5 *Soient for_1 et for_2 deux formules de $\mathcal{F}(\mathcal{P})$.*

a) $for_1 \equiv_T for_2$ implique $for_1 =\!\mid_T \models for_2$. La réciproque est fausse.
b) $for_1 \equiv_T for_2$ équivaut à $\models_T (for_1 \leftrightarrow for_2)$.
c) $for_1 =\!\mid_T \models for_2$ équivaut à $\models_T (for_1 \leftarrow\!\!\rightarrow for_2)$.
d) $for_1 \equiv_B for_2$ équivaut à $\models_B (for_1 \Leftrightarrow for_2)$ et équivaut à
$for_1 =\!\mid_B \models for_2$.

Démonstration a) L'implication est claire.
Pour montrer que la réciproque est fausse, considérons les deux formules:

$(A \leftrightarrow B) \wedge (A \vee \neg A)$ et

$(A \Leftrightarrow B)$, où A et B sont deux atomes.

On a bien:

$(A \leftrightarrow B) \wedge (A \vee \neg A) =\!\mid_T \models (A \Leftrightarrow B)$.

En effet un modèle de $(A \leftrightarrow B) \wedge (A \vee \neg A)$ est tel que A et B sont soit tous les deux Faux, soit tous les deux Vrai. De même pour un modèle de $(A \Leftrightarrow B)$. Cependant, on n'a pas $(A \leftrightarrow B) \wedge (A \vee \neg A) \equiv_T (A \Leftrightarrow B)$. Il suffit de prendre une interprétation i telle que $vv_i(A) = I$ et $vv_i(B) = F$.

b) Par définition (définition 2.1.8).

c) $\models_T (for_1 \leftarrow\!\!\rightarrow for_2)$ équivaut à $\models_T (for_1 \rightarrow for_2)$ et
$\models_T (for_2 \rightarrow for_1)$,
ce qui équivaut à $for_1 \models_T for_2$ et $for_2 \models_T for_1$,
ce qui équivaut à $for_1 =\!\mid_T \models for_2$.

d) Clair. ♦

Remarque: $for_1 =\!\mid_T \models for_2$ implique $for_1 =\!\mid_B \models for_2$ mais la réciproque est fausse. Par exemple, si A et B sont deux atomes:

$$(A \rightarrow B)\wedge(A \rightarrow \neg B)\wedge(\neg A \rightarrow B)\wedge(\neg A \rightarrow \neg B) =\!\mid_B \models$$
$$(B \rightarrow A)\wedge(B \rightarrow \neg A)\wedge(\neg B \rightarrow A)\wedge(\neg B \rightarrow \neg A),$$

puisque ces deux expressions n'ont aucun modèle bivalué. Alors que {B} est un modèle trivalué fort du premier et non du second.

2.2 CALCUL DES PREDICATS

2.2.1 Syntaxe.

L(*) est un langage logique comprenant:

- Un ensemble fini ou infiniment dénombrable de variables: Var.

- Pour tout n élément de $\mathbb{N}$, un ensemble fini ou infiniment dénombrable F_n de fonctions n-aires. Les fonctions 0-aires constituent l'ensemble des constantes Const.

- Un ensemble $\mathcal{A}$ de symboles de ponctuation: {(,), ,}

- Pour tout n élément de $\mathbb{N}$, un ensemble fini ou infiniment dénombrable P_n de predicats n-aires. P_2 contient éventuellement le prédicat d'égalité.

- Un connecteur unaire: $\neg$.

- Un ensemble de connecteurs binaires: $\wedge$, $\vee$, $\neg$, $\Rightarrow$, $\rightarrow$, $\leftrightarrow$, $\Leftrightarrow$, $\leftarrow\rightarrow$.

- Deux quantificateurs: $\forall$, $\exists$.

Remarque: Nous utilisons la notation L(*) employée dans le formalisme de Yasuhara [45]. Dans la démonstration du dernier théorème de complétude, on étend L(*) à L(*') en lui adjoignant des constantes.

Définitions 2.2.1 L'ensemble des termes TER(L(*)) se définit inductivement de la manière suivante:

$$Var \subseteq TER(L(*));$$

$$Const \subseteq TER(L(*));$$

Si $t_1, t_2, ..., t_n \in TER(L(*))$, $f \in F_n$,
alors $f(t_1, t_2, ..., t_n) \in TER(L(*))$.

L'ensemble ATO(L(*)) des formules atomiques se définit inductivement de la manière suivante:

$$P_0 \subseteq ATO(L(*)).$$

Si $t_1, t_2, ..., t_n \in TER(L(*))$ et $p \in P_n$, alors
$p(t_1, t_2, ..., t_n) \in ATO(L(*))$.

Les éléments de ATO(L(*)) s'appellent des atomes et les négations d'atomes des littéraux.

L'ensemble FOR(L(*)) des formules se définit inductivement de la manière suivante:

$$ATO(L(*)) \subseteq FOR(L(*)).$$

Si $f \in FOR(L(*))$, alors $\neg f \in FOR(L(*))$.

Si $f, g \in FOR(L(*))$, et γ est un connecteur binaire,
alors $(f \gamma g) \in FOR(L(*))$.

Si $f \in FOR(L)$, alors $\exists x\, f, \forall x\, f \in FOR(L(*))$.

On définit de façon usuelle les notions de variable libre, liée, et de formule close et de terme clos,de clôtures universelle et existentielle d'une formule A.

Définitions 2.2.2 Si t et u sont deux termes, et si x est une variable, on définit de manière usuelle $u(x \leftarrow t)$, le terme obtenu en substituant à chaque occurrence de x dans u, le terme t.

Si A est une formule et t un terme, on note $A(x \leftarrow t)$ la formule obtenue en substituant à chaque occurrence libre de x dans A le terme t après avoir renommé les variables liées de A de sorte qu'elles ne figurent pas dans t (t est alors libre pour x dans A, i.e.: si $x_1, x_2, ..., x_n$ sont les variables libres de t, il n'y a pas de x dans la portée d'un quantificateur de x_i dans A).

Définitions 2.2.3 Si A est une formule de FOR(L(*)), on définit la notion de sous formule de A inductivement de la manière suivante:

- A est une sous formule de A.
- Si $A = \neg B$ ou $A = (B \gamma C)$ avec $\gamma \in \{\vee, \wedge, \Rightarrow, \Leftrightarrow, \rightarrow, \leftrightarrow, \leftarrow\rightarrow\}$, B et C sont des sous formules immédiates de A.
- Si $A = \forall x\ B$ ou $A = \exists x\ B$, B est une sous formule immédiate de A.
- Toute sous formule d'une sous formule immédiate de A est une sous formule de A.

Si f et g sont deux formules de FOR(L(*)), on peut alors définir la notion de substitution de la formule f à la formule g dans une formule A de FOR(L(*)), notée $A(f \leftarrow g)$.

- Si f n'est pas une sous formule de A, $A(f \leftarrow g) = A$.
- Si $A = f$, $A(f \leftarrow g) = g$.
- Si $A = \neg B$, $A(f \leftarrow g) = \neg B(f \leftarrow g)$.
- Si $A = (B \gamma C)$ avec $\gamma \in \{\vee, \wedge, \Rightarrow, \Leftrightarrow, \rightarrow, \leftrightarrow, \leftarrow\rightarrow\}$, $A(f \leftarrow g) = (B(f \leftarrow g) \gamma C(f \leftarrow g))$.
- Si $A = \forall x\ B$, $A(f \leftarrow g) = \forall x\ B(f \leftarrow g)$.
- Si $A = \exists x\ B$, $A(f \leftarrow g) = \exists x\ B(f \leftarrow g)$.

Définition 2.2.4 Une interprétation $\mathcal{U}$ pour L(*) est constituée d'un domaine non vide D; de fonctions de D^n dans D; de fonctions de D^n dans {V, F, I}.

Il y a une bijection β fixe entre les symboles de fonctions n-aires et les fonctions de D^n vers D et entre les symboles de prédicats n-aires et les fonctions de D^n dans {V, F, I}.

Il y a une correspondance fixe (pas necessairement une bijection) entre les constantes de L(*) et les éléments de D.

Une assignation ϕ est une application de Var dans D.

A chaque assignation, ϕ: Var $\rightarrow$ D, on associe l'application valeur de vérité relativement à ϕ, vv_ϕ: FOR(L(*)) $\rightarrow$ {V, F, I} de la façon suivante:

On prolonge ϕ à TER(L(*)) inductivement par:

$$\phi(f(t_1, t_2, ..., t_n)) = \beta(f)(\phi(t_1), \phi(t_2), ..., \phi(t_n)).$$

Si A est atomique, de la forme: $p(t_1, t_2, ..., t_n)$,
$$vv\phi(A) = \beta(p)(\phi(t_1), \phi(t_2), ..., \phi(t_n)).$$

Si $A = \neg B$, $vv\phi(A) = \neg vv\phi(B)$ et si $A = (B\ \gamma\ C)$, avec $\gamma \in \{\vee, \wedge, \Rightarrow, \Leftrightarrow, \rightarrow, \leftrightarrow, \leftarrow\rightarrow\}$, $vv\phi(A) = \gamma(vv\phi(B), vv\phi(C))$ en utilisant les tables de vérité données en 2.1.4.

Si $A = \exists x\ f$, $vv\phi(A) = V$ s'il existe $a \in D$, tel que $vv\phi_{(x\leftarrow a)}(A) = V$, $vv\phi(A) = F$ si pour tout $a \in D$, $vv\phi_{(x\leftarrow a)}(A) = F$, $vv\phi(A) = I$ sinon.

Si $A = \forall x\ f$, $vv\phi(A) = V$ si pour tout $a \in D$, $vv\phi_{(x\leftarrow a)}(A) = V$, $vv\phi(A) = F$ s'il existe $a \in D$, tel que $vv\phi_{(x\leftarrow a)}(A) = F$, $vv\phi(A) = I$ sinon.

L'expression $\phi(x\leftarrow a)$ désigne l'assignation ϕ' telle que $\phi'(y) = \phi(y)$ si y est une variable différente de x et $\phi'(x) = a$.

Remarque: Si à chaque symbole de prédicat, on associe une fonction de D^n dans $\{V, F\}$, on a une interprétation bivaluée habituelle.

Lemme 2.2.1 *Soit $\mathcal{U}$ une interprétation sur L(*). Si t est un terme clos, pour toutes assignations ϕ et ψ, on a: $\phi(t) = \psi(t)$.*

Démonstration Par induction sur t. ♦

Lemme 2.2.2 *Soit $\mathcal{U}$ une interprétation sur L(*). Si A est une formule atomique close, pour toutes assignations ϕ et ψ, on a:*
$$vv\phi(A) = vv\psi(A).$$

Démonstration Par induction sur A en utilisant le lemme ci-dessus. ♦

Lemme 2.2.3 *Soit $\mathcal{U}$ une interprétation sur L(*). Soit A une formule à n variables libres $x_1, x_2, ..., x_n$. Si ϕ et ψ sont des assignations telles que:*

$$\forall j = 1,2, ..., n,\ \phi(x_j) = \psi(x_j) \text{ alors:}$$

$$vv\phi(A(x_1, ..., x_n)) = vv\psi(A(x_1, ..., x_n)).$$

Démonstration Par induction sur t terme et ensuite sur A. ♦

Lemme 2.2.4 *Soit $\mathcal{U}$ une interprétation sur L(*).*
Si A est une formule close, pour toutes assignations ϕ et ψ, on a:

$$vv\phi(A) = vv\psi(A).$$

On note $vv_{\mathcal{U}}(A)$ la valeur de vérité de A relativement à $\mathcal{U}$.

Démonstration Par induction sur A. Si A est atomique, lemme 2.2.2. Le seul cas non trivial est $A = \forall x\ B$, la seule variable libre éventuelle de B étant x. Or $vv\phi_{(x\leftarrow a)}(B) = vv\psi_{(x\leftarrow a)}(B)$ d'après le lemme 2.2.3. ♦

On peut maintenant définir la valeur de vérité d'une formule indépendamment d'une assignation, par la valeur de vérité de sa clôture universelle.

Définitions 2.2.5 La valeur de vérité d'une formule A relativement à $\mathcal{U}$ est la valeur de vérité de sa clôture universelle et se note $vv_{\mathcal{U}}(A)$.

<u>On a donc: $vv_{\mathcal{U}}(A) = vv_{\mathcal{U}}(\forall A)$</u>.

Une formule close A est vraie (resp. fausse; resp. indéterminée) dans $\mathcal{U}$ si $vv_{\mathcal{U}}(A) = V$ (resp. F; resp. I). Par conséquent:

Une formule A est **vraie** dans $\mathcal{U}$ si sa clôture universelle est vraie dans $\mathcal{U}$.

Une formule A est **satisfiable** dans $\mathcal{U}$ si sa clôture existentielle est vraie dans $\mathcal{U}$.

N.B. On emploiera indifféremment V (resp. F, I) ou Vrai (resp. Faux, Indéterminé)

Lemme 2.2.5 *Soit A une formule, $\forall A$ sa clôture universelle, $\exists A$ sa clôture existentielle. On a les équivalences suivantes:*

$$vv_{\mathcal{U}}(\forall A) = V \Leftrightarrow \forall\phi,\ vv\phi(A) = V.$$
$$vv_{\mathcal{U}}(\forall A) = F \Leftrightarrow \exists\phi,\ vv\phi(A) = F.$$
Dans les autres cas, $vv_{\mathcal{U}}(\forall A) = I$.

$$vv_{\mathcal{U}}(\exists A) = V \Leftrightarrow \exists\phi,\ vv\phi(A) = V.$$
$$vv_{\mathcal{U}}(\exists A) = F \Leftrightarrow \forall\phi,\ vv\phi(A) = F.$$
Dans les autres cas, $vv_{\mathcal{U}}(\exists A) = I$.

Démonstration Par induction sur le nombre de variables libres de A.

Si $\forall A = \forall x\ A(x)$ (A n'a qu'une variable libre), on utilise le fait que
$vv\phi_{(x\leftarrow\psi(x))}(A) = vv\psi(A)$ (lemme 2.2.3).

$vv_{\mathcal{U}}(\forall A) = V \Leftrightarrow \forall\phi,\ vv_\phi(\forall x\ A(x)) = V \Leftrightarrow \forall\phi,\ \forall d,\ vv_{\phi(x\leftarrow d)}(A(x)) = V$.
Or $\forall\phi,\ \forall d,\ vv_{\phi(x\leftarrow d)}(A(x)) = V$ implique $\forall\psi,\ vv_\psi(A) = V$ en prenant $d = \psi(x)$.
Réciproquement, $\forall\psi,\ vv_\psi(A) = V$ implique $\forall\phi,\ \forall d,\ vv_{\phi(x\leftarrow d)}(A(x)) = V$ en choisissant ψ telle que $d = \psi(x)$.

Si A a n variables libres $x_1, ..., x_n$, $\forall x_n\ A(x_1, ..., x_n)$ a n-1 variables libres et $\forall A$ comme clôture universelle.

Par récurrence,

$$vv_{\mathcal{U}}(\forall A) = V \Leftrightarrow \forall\phi,\ vv_\phi(\forall x_n\ A(x_1, ..., x_n)) = V.$$

On utilise ensuite:

$$\begin{aligned}&\forall\phi,\ vv_\phi(\forall x_n\ A(x_1, ..., x_n)) = V \Leftrightarrow\\ &\forall\phi,\ \forall d,\ vv_{\phi(x_n\leftarrow d)}(A(x_1, ..., x_n)) = V \Leftrightarrow\\ &\forall\psi,\ vv_\psi(A(x_1, ..., x_n)) = V.\end{aligned}$$

En effet dans un sens on utilise le fait que

$$vv_{\psi(x_n\leftarrow\psi(x_n))}(A(x_1, ..., x_n)) = vv_\psi(A(x_1, ..., x_n)),$$

et dans l'autre sens, on considère ψ telle que $\psi(x_i) = \phi(x_i)$ si $i \neq n$ et $\psi(x_n) = d$. ♦

2.2.2 Modèles et conséquences.

Définition 2.2.6 Une interprétation $\mathcal{U}$ est un modèle fort d'un ensemble $\mathcal{T}$ de formules du langage L(*) si chaque A de $\mathcal{T}$ est vraie dans $\mathcal{U}$.

C'est-à-dire si pour chaque A de $\mathcal{T}$ et chaque ϕ: Var $\rightarrow$ D,

$$vv_\phi(A) = \text{Vrai. (lemme 2.2.5).}$$

$\mathcal{T}$ est alors fortement consistant au sens trivalué.

Une interprétation $\mathcal{U}$ est un modèle faible d'un ensemble $\mathcal{T}$ de formules du langage L(*) si chaque A de $\mathcal{T}$ n'est jamais fausse dans $\mathcal{U}$.

C'est-à-dire si pour chaque A de $\mathcal{T}$ et chaque ϕ: Var $\rightarrow$ D,

$$vv_\phi(A) \neq \text{Faux. (lemme 2.2.5)}$$

$\mathcal{T}$ est alors faiblement consistant au sens trivalué.

Pour les interprétations bivaluées, les notions de modèle faible et fort coïncident:

Proposition 2.2.1 *Si $\mathcal{U}$ est une interprétation bivaluée, alors pour chaque formule for du langage L(*),*

$$vv_{\mathcal{U}}(for) \in \{V, F\}.$$

Démonstration Par induction sur la complexité de for, en utilisant le fait que tous les connecteurs utilisés dans le langage laissent stable l'ensemble {V, F}.
Si for $\in$ ATO(L(*)), puisque $\mathcal{U}$ est bivalué, alors $vv_{\mathcal{U}}$(for) $\in$ {V, F}.♦

Définition 2.2.7 Si Ax et Bx sont deux sous-ensembles de FOR(L(*)), Bx est conséquence trivaluée forte de Ax, si et seulement si tout modèle trivalué fort de Ax est un modèle trivalué fort de Bx.
On le note:

$$\text{Ax} \models_T \text{Bx}.$$

On définit de même les notions de conséquences trivaluées faibles (Ax $\models_t$ Bx) et de conséquence bivaluée (Ax $\models_B$ Bx).

Définition 2.2.8 Si Ax et Bx sont deux ensembles de formules de FOR(L(*)),

Ax $=\!\mid_T \models$ Bx (resp. Ax $=\!\mid_t \models$ Bx) si et seulement si Ax et Bx ont les mêmes modèles trivalués forts (resp. faibles).

On définit Ax $=\!\mid_B \models$ Bx de la même manière.

Relations entre ces notions.

Un modèle trivalué fort de Ax est un modèle trivalué faible; par conséquent, si Ax est fortement consistant au sens trivalué, alors il est faiblement consistant au sens trivalué.

Un modèle bivalué de Ax est un modèle trivalué fort de Ax; par conséquent, si Ax est consistant au sens bivalué, alors il est fortement consistant au sens trivalué.

Aucune autre implication n'est vraie:

On peut reprendre les contre exemples donnés en calcul propositionnel en introduisant des formules atomiques avec variables.

Les symboles p et q sont deux symboles de prédicats.

Remarques: - Si l'on interprète les prédicats par les fonctions constantes I de D^n dans {V, F, I} dans $\mathcal{U}$, on n'a pas toujours un modèle trivalué faible d'une formule. Par exemple si on considère la formule:

$$f(x_1, ..., x_n) = (p(x_1, ..., x_n) \rightarrow q(x_1, ..., x_n)) \rightarrow p(x_1, ..., x_n).$$

$\forall\phi$, $vv_\phi(f(x_1, ..., x_n)) = F$, et par conséquent $vv_{\mathcal{U}}(f(x_1, ..., x_n)) = F$.

- Un ensemble de formules ne possède pas nécessairement de modèle trivalué faible: par exemple, la formule:

$$f(x_1, ..., x_n) = ((p(x_1, ..., x_n) \wedge \neg p(x_1, ..., x_n)) \rightarrow q(x_1, ..., x_n)) \\ \rightarrow (q(x_1, ..., x_n) \wedge \neg q(x_1, ..., x_n)).$$

Dans n'importe quelle interprétation $\mathcal{U}$, $vv_{\mathcal{U}}(f(x_1, ..., x_n)) = F$.
En effet $vv_{\mathcal{U}}(f(x_1, ..., x_n)) = F \Leftrightarrow vv_{\mathcal{U}}(\forall f) = F \Leftrightarrow \exists\phi$, $vv_\phi(f) = F$.
Or on a: $\forall\mathcal{U}$, $\forall\phi$, $vv_\phi(f) = F$.
Si ϕ est une valuation telle que $\phi(x_i) = t_i$, on a $\beta(p)(t_1, ..., t_n) = V$, F, ou I.
Donc $((\beta(p)(t_1, ..., t_n) \wedge \neg\beta(p)(t_1, ..., t_n)) \rightarrow \beta(q)(t_1, ..., t_n))$ prend toujours la valeur V, alors que $\beta(q)(t_1, ..., t_n) \wedge \neg\beta(q)(t_1, ..., t_n)$ ne prend jamais la valeur V.

- Aucune autre implication n'est vraie:

- Si on prend pour Ax l'ensemble:

$$\{p(x_1, ..., x_n), \neg p(x_1, ..., x_n)\},$$

il admet un modèle trivalué faible dans lequel p est interprété par la fonction $\beta(p)$ constamment égale à I. Ax est consistant au sens trivalué faible sans l'être au sens trivalué fort.

- L'ensemble Ax:

$$\{p(x_1, ..., x_n) \rightarrow q(x_1, ..., x_n), p(x_1, ..., x_n) \rightarrow \neg q(x_1, ..., x_n), \\ \neg p(x_1, ..., x_n) \rightarrow \neg q(x_1, ..., x_n), \neg p(x_1, ..., x_n) \rightarrow q(x_1, ..., x_n)\}$$

est fortement consistant au sens trivalué sans l'être au sens bivalué.

Dans un modèle trivalué de Ax, p est nécessairement interprété par $\beta(p)$ constamment égale à I et q est interprété par n'importe quelle fonction de D^n dans {V, F, I}

Cependant, on a la proposition suivante:

Proposition 2.2.2 *Si Ax est un ensemble de formules de FOR(L(*)), une interprétation trivaluée* $\mathcal{U}$ *est un modèle trivalué fort de Ax' = Ax* $\cup$ *{ato$\vee\neg$ato/ ato* $\in$ *ATO(L(*))} si et seulement si* $\mathcal{U}$ *est une interprétaion bivaluée modèle de Ax.*

Démonstration Si $\mathcal{U}$ est un modèle trivalué fort de {ato$\vee\neg$ato/ ato $\in$ ATO(L(*))} alors $\forall$ato $\in$ ATO(L(*)), ato = $p(t_1, ..., t_n)$, $vv_{\mathcal{U}}$(ato$\vee\neg$ato) = V implique $\forall\phi$, $vv_\phi(p(t_1, ..., t_n)\vee\neg p(t_1, ..., t_n))$ = V et par conséquent $\beta(p)$ est à valeurs dans {V, F}.
Réciproquement, si $\mathcal{U}$ est une interprétation bivaluée, alors $vv_{\mathcal{U}}$(ato$\vee\neg$ato) = Vrai, pour tout ato de ATO(L(*)). ♦

- Si Ax $\models_T$ Bx, alors Ax $\models_B$ Bx.

- Si Ax $\models_t$ Bx, alors Ax $\models_B$ Bx.

Aucune autre implication n'est vraie:

- Si Ax $\models_B$ Bx, on n'a pas Ax $\models_T$ Bx.

Par exemple, prenons pour Ax l'ensemble vide et pour Bx l'ensemble:

$\{p(x_1, ..., x_n)\vee\neg p(x_1, ..., x_n)\}$.

Toute interprétation bivaluée est un modèle de Ax et de Bx alors qu'une interprétation dans laquelle p est interprété par la fonction constante de D^n dans {V, F, I}, égale à I est un modèle trivalué fort de Ax et non de Bx.

- Si Ax $\models_B$ Bx, on n'a pas Ax $\models_t$ Bx.
Par exemple, prenons pour Ax l'ensemble:

$\{p(x_1, ..., x_n), \neg p(x_1, ..., x_n)\}$

et pour Bx l'ensemble:

$\{q(x_1, ..., x_n)\}$.

Ax ne possède pas de modèle bivalué tandis que l'interprétation dans laquelle q est interprété par la fonction constante égale à Faux et p par la fonction constante égale à I est un modèle trivalué faible de Ax et non de Bx.

- Si Ax $\models_T$ Bx, on n'a pas Ax $\models_t$ Bx.
Par exemple, prenons pour Ax l'ensemble:

$\{p(x_1, ..., x_n), p(x_1, ..., x_n) \rightarrow q(x_1, ..., x_n)\}$

et pour Bx l'ensemble:

$\{q(x_1, ..., x_n)\}$.

Le seul modèle trivalué fort de Ax est l'interprétation dans laquelle p et q sont interprétés par la fonction constante égale à Vrai ; c'est un modèle trivalué fort de Bx alors que l'interprétation dans laquelle q est interprété par la fonction constante égale à Faux et p par la fonction constante égale à I est un modèle trivalué faible de Ax et non de Bx.

- Si Ax $\models_t$ Bx, on n'a pas Ax $\models_T$ Bx.
Par exemple, prenons pour Ax l'ensemble:

$\{p(x_1, ..., x_n) \leftrightarrow \neg p(x_1, ..., x_n)\}$

et pour Bx l'ensemble:

$\{p(x_1, ..., x_n)\}$.

Le seul modèle trivalué faible de Ax est l'interprétation dans laquelle p est interprété par la fonction constante égale à I.
En effet si $\beta(p) \neq I$, $\exists(d_1, ..., d_n) \in D^n$ tel que $\beta(p)(d_1, ..., d_n) \neq I$, si on prend ϕ telle que $\phi(x_i) = d_i$, $vv_\phi(p(x_1, ..., x_n) \leftrightarrow \neg p(x_1, ..., x_n)) = F$.
Donc ce n'est pas un modèle trivalué faible de Ax puisque l'on doit avoir $\forall\phi\ vv_\phi(p(x_1, ..., x_n) \leftrightarrow \neg p(x_1, ..., x_n)) \neq F$.
C'est un modèle trivalué faible de Bx.
Or le seul modèle trivalué fort de Ax est l'interprétation dans laquelle p est interprété par la fonction constante égale à I qui n'est pas un modèle trivalué fort de Bx.

2.2.3 Tautologies et équivalences trivaluées.

Définition 2.2.9 Une formule A est une tautologie si elle est vraie dans toute interprétation du langage.

Une formule A est satisfiable s'il existe une interprétation $\mathcal{U}$ dans laquelle elle soit satisfiable.

On note $\mathcal{V}(*)$ l'ensemble des tautologies.

Si A est une tautologie, on le note:

$$\models_T A.$$

Remarques: - On n'a plus l'équivalence suivante:

$\models_T A \leftrightarrow B$ équivaut à :

Pour toute interprétation $\mathcal{U}$, $vv_{\mathcal{U}}(\forall A) = vv_{\mathcal{U}}(\forall B)$.

En effet, $\models_T A \leftrightarrow B$ équivaut à:
Pour toute interprétation $\mathcal{U}$, $vv_{\mathcal{U}}(\forall(A \leftrightarrow B)) = V \Leftrightarrow$
Pour toute interprétation $\mathcal{U}$, $\forall\phi$, $vv\phi(A \leftrightarrow B) = V \Leftrightarrow$
Pour toute interprétation $\mathcal{U}$, $\forall\phi$, $vv\phi(A) = vv\phi(B)$.

Ceci entraîne bien que pour toute interprétation $\mathcal{U}$, $vv_{\mathcal{U}}(\forall A) = vv_{\mathcal{U}}(\forall B)$, mais la réciproque est fausse.
En effet, supposons que $vv_{\mathcal{U}}(\forall A) = vv_{\mathcal{U}}(\forall B) = F$. $\exists\phi$ telle que $vv\phi(A) = F$ et $\exists\phi'$ telle que $vv\phi'(B) = F$ mais on peut très bien avoir $vv\phi(A) \neq vv\phi'(B)$

La réciproque est vraie uniquement si A et B sont des formules closes.

De même, on n'a plus l'équivalence $\models_B A \Leftrightarrow B$ équivaut à :

Pour toute interprétation bivaluée $\mathcal{U}$, $vv_{\mathcal{U}}(\forall A) = vv_{\mathcal{U}}(\forall B)$.

On définit donc la notion d'équivalence trivaluée (et bivaluée) de la manière suivante, en choisissant la définition la plus forte:

Définitions 2.2.10 Deux formules A et B de FOR(L(*)) sont logiquement équivalentes si elles vérifient

$$\models_T A \leftrightarrow B$$

On le note: $A \equiv_T B$.

On a de même $A \equiv_B B$ si et seulement si $\models_T A \Leftrightarrow B$ par définition.

Lemme 2.2.6 *Si $A \equiv_T B$, alors on a:*

- $\forall A \equiv_T \forall B$
- *A et B ont les mêmes modèles trivalués forts.*
- $\forall x\, A \equiv_T \forall x\, B$
- $\exists x\, A \equiv_T \exists x\, B$
- $\neg A \equiv_T \neg B$

Si $A_1 \equiv_T B_1$ et $A_2 \equiv_T B_2$, alors on a:

- *Si* $\gamma \in \{\vee, \wedge, \Rightarrow, \Leftrightarrow, \rightarrow, \leftrightarrow, \leftarrow\rightarrow\}$, $(A_1\, \gamma A_2) \equiv_T (B_1\, \gamma B_2)$

En plus des exemples donnés en 2.1.1, nous avons les exemples suivants d'équivalences trivaluées en calcul des prédicats.

Exemples 2.2.1

- $\forall x\, (P \wedge Q) \equiv_T (\forall x\, P) \wedge (\forall x\, Q)$.
- $\exists x\, (P \vee Q) \equiv_T (\exists x\, P) \vee (\exists x\, Q)$.
- $\forall x\, (P \rightarrow Q) \equiv_T (\exists x\, P) \rightarrow Q$ si x n'est pas une variable libre de Q.
- $\exists x\, (P \rightarrow Q) \equiv_T (\forall x\, P) \rightarrow Q$ si x n'est pas une variable libre de Q.
- $\neg(\forall x\, P) \equiv_T (\exists x\, \neg P)$.
- $\neg(\exists x\, P) \equiv_T (\forall x\, \neg P)$.

- Par contre, on n'a pas: $A \rightarrow B \equiv_T \neg B \rightarrow \neg A$.

C'est une des raisons de l'introduction de deux symboles d'équivalence: $\leftrightarrow$ et $\leftarrow\rightarrow$.

En effet:

- $A \leftrightarrow B \equiv_T (A \rightarrow B) \wedge (B \rightarrow A) \wedge (\neg A \rightarrow \neg B) \wedge (\neg B \rightarrow \neg A)$.

- $A \leftarrow\rightarrow B \equiv_T (A \rightarrow B) \wedge (B \rightarrow A)$.

Lemme 2.2.7 *Soit A(x) une formule dans laquelle x est une variable libre.*

Soit t un terme ayant $x_1, x_2, ..., x_n$ pour variables libres, libre pour x dans A (il n'y a pas de x dans la portée d'un quantificateur de x_i dans A, i = 1, 2, ..., n).

Si ϕ et ψ sont des assignations telles que $\forall x_j \in Var$, $x_j \neq x$, $\phi(x_j) = \psi(x_j)$ et $\phi(t) = \psi(x)$ alors:

$$vv\phi(A(x \leftarrow t)) = vv\psi(A(x)).$$

Démonstration Par induction sur A.

On commence par montrer que si u est un terme, $\psi(u) = \phi(u(x \leftarrow t))$.

Ce qui permet de montrer le résultat pour A atomique.
Si $A = p(t_1, ..., t_n)$, $A(x \leftarrow t) = p(t_1(x \leftarrow t), ..., t_n(x \leftarrow t))$.
Or $vv\phi(A(x \leftarrow t)) = \beta(p)(\phi(t_1(x \leftarrow t)), ..., \phi(t_n(x \leftarrow t))) = \beta(p)(\psi(t_1), ..., \psi(t_n)) = vv\psi(A(x))$.

Le cas non trivial de l'induction est $A(x) = \forall y\, B(x)$; t est libre pour x dans A donc y n'est pas une variable libre de t et $y \neq x$ car x est une variable libre de A.

$A(x \leftarrow t)) = \forall y\, B(x \leftarrow t)$ car $y \neq x$;

or $\phi(y\leftarrow a)(t) = \phi(t) = \psi(x) = \psi(y\leftarrow a)(x)$ car y n'est pas variable libre de t et $x \neq y$.

On applique alors l'hypothèse de récurrence à B et on a:
$vv\phi_{(y\leftarrow a)}(B(x \leftarrow t)) = vv\psi_{(y\leftarrow a)}(B(x))$.

Donc $vv\phi(A(x \leftarrow t)) = vv\phi(\forall y\, B(x \leftarrow t)) = vv\psi(\forall y\, B(x)) = vv\psi(A(x))$.
♦

Lemme 2.2.8 *Si A(x) est une formule admettant x comme variable libre, et si t est un terme libre pour x dans A, on a:*

Si A(x) est une tautologie, alors $A(x \leftarrow t)$ est une tautologie.

Démonstration Soit ϕ une assignation. Soit ψ telle que: $\forall x_j \in Var$, $x_j \neq x$, $\phi(x_j) = \psi(x_j)$ et $\phi(t) = \psi(x)$ alors $vv\phi(A(x \leftarrow t)) = vv\psi(A(x))$ d'après le lemme ci-dessus. On a donc $vv\phi(A(x \leftarrow t))$ = Vrai puisque A(x) est une tautologie. ♦

Lemme 2.2.9 *Si f et g sont deux formules de For(L(*)), on a:*

$f \equiv_T g$ implique $A \equiv_T A(f \leftarrow g)$.

Démonstration Par induction sur la formule A, en utilisant le lemme 2.2.6.
Soit f n'est pas une sous formule de A et $A(f \leftarrow g) = A$.
Si f est une sous formule de A, on procède inductivement.
- $f = A$, $A \equiv_T A(f \leftarrow g)$ n'est rien d'autre que $f \equiv_T g$.
- Si $f = \forall x\, B$ par exemple, $B \equiv_T B(f \leftarrow g)$ par induction, donc:

$$\forall x\, B \equiv_T \forall x\, B(f \leftarrow g). \blacklozenge$$

- **On n'a plus** comme dans le cas propositionnel: Si A et B sont deux formules de FOR(L(*)),

$$A \models_T B \text{ équivaut à: } \models_T A \rightarrow B,$$

sauf si A et B sont des formules **closes**.

En effet, $\models_T A \rightarrow B$ implique $A \models_T B$. Si $\mathfrak{U}$ est un modèle de A, $\forall \phi$, $vv_\phi(A) = V$, et comme $vv_\phi(A \rightarrow B) = V$, $vv_\phi(B) = V$, donc $\mathfrak{U}$ est un modèle de B.
La réciproque est fausse: si $vv_\phi(A) = V$, on n'a pas forcément $vv_\phi(B) = V$ sauf si $\forall \phi$, $vv_\phi(A) = V$ (c'est pour cela que la réciproque est vraie avec des formules closes).

- **On n'a plus** comme dans le cas propositionnel: Si A et B sont deux formules de FOR(L(*)),

$A \mathrel{=\!\!|}_T \models B$ équivaut à: $\models_T A \leftrightarrow B$, sauf si A et B sont des formules **closes**.

En effet, $\models_T A \leftrightarrow B$ implique $A \mathrel{=\!\!|}_T \models B$. $\models_T A \leftrightarrow B$ équivaut à $\forall \mathfrak{U}$, $\forall \phi$, $[vv_\phi(A) = V \Leftrightarrow vv_\phi(B) = V]$. Si $\mathfrak{U}$ est un modèle de A, $\forall \phi$, $vv_\phi(A) = V$, par conséquent $\forall \phi$, $vv_\phi(B) = V$, donc $\mathfrak{U}$ est un modèle de B.
La réciproque est fausse: si $vv_\phi(A) = V$, on n'a pas forcément $vv_\phi(B) = V$ sauf si $\forall \phi$, $vv_\phi(A) = V$ (c'est pour cela que la réciproque est vraie avec des formules closes).

Les propositions 2.1.3 et 2.1.5 deviennent vraies pour les formules **closes** et on a l'analogue de la proposition 2.1.4.

Les propositions suivantes établissent des liens entre les notions de conséquences et de tautologies, celles de conséquence et de consistance, et enfin d'équivalence et de tautologie.

Proposition 2.2.3 *Soient $A_1, A_2, ..., A_n, B_1, B_2, ..., B_m$ des formules closes de FOR(L(*)).*

Toutes les assertions suivantes sont équivalentes:

(1) $\{A_1, A_2, ..., A_n\} \models_T \{B_1, B_2, ..., B_m\}$
(2) $\{A_1 \wedge A_2 \wedge ... \wedge A_n\} \models_T \{B_1, B_2, ..., B_m\}$
(3) $\{A_1, A_2, ..., A_n\} \models_T \{B_1 \wedge B_2 \wedge ... \wedge B_m\}$
(4) $\{A_1 \wedge A_2 \wedge ... \wedge A_n\} \models_T \{B_1 \wedge B_2 \wedge ... \wedge B_m\}$
(5) $\{A_1, A_2, ..., A_{n-1}\} \models_T \{A_n \rightarrow B_1, A_n \rightarrow B_2, ..., A_n \rightarrow B_m\}$
(6) $\{A_1 \wedge A_2 \wedge ... \wedge A_{n-1}\} \models_T \{A_n \rightarrow (B_1 \wedge B_2 \wedge ... \wedge B_m)\}$
(7) $\models_T (A_1 \wedge A_2 \wedge ... \wedge A_n) \rightarrow (B_1 \wedge B_2 \wedge ... \wedge B_m)$

Démonstration (1)$\Leftrightarrow$(2)$\Leftrightarrow$(3)$\Leftrightarrow$(4) car i est un modèle de $\{A_1, A_2, ..., A_n\}$ si et seulement si i est un modèle de $A_1 \wedge A_2 \wedge ... \wedge A_n$. On a, de plus, si Δ est un sous-ensemble de formules closes de FOR(L(*)), $\Delta \cup \{A\} \models_T \{B\} \Leftrightarrow \Delta \models_T A \rightarrow B$. Donc, (3)$\Leftrightarrow$(6) et (4)$\Leftrightarrow$(7) Enfin, on a:
$A_n \rightarrow (B_1 \wedge B_2 \wedge ... \wedge B_m) \equiv_T A_n \rightarrow B_1 \wedge A_n \rightarrow B_2 \wedge ... \wedge A_n \rightarrow B_m$, donc (6)$\Leftrightarrow$(5). ♦

Remarque: Cette proposition n'est plus vraie si l'on remplace le connecteur $\rightarrow$ par le connecteur $\Rightarrow$. On n'a plus $\Delta \cup \{A\} \models_T \{B\} \Leftrightarrow \Delta \models_T (A \Rightarrow B)$. En effet, $A \Rightarrow A$ n'est plus une tautologie, aussi n'a-t-on plus $\models_T (A \Rightarrow A)$.

Proposition 2.2.4 *Soient $At = p(t_1, ..., t_n)$ un élément de ATO(L(*)) et Ax un ensemble de formules de FOR(L(*)).*

On a l'équivalence suivante:

(1) $Ax \cup \{\neg At\}$ n'est pas fortement consistant au sens trivalué si et seulement si
(2) $\neg At$ n'est vrai dans aucun modèle trivalué fort de Ax.

Cependant, on n'a pas (même si At est un atome clos, alors que c'est vrai en bivalué dans le cas d'un atome clos):

$Ax \cup \{\neg At\}$ n'est pas fortement consistant au sens trivalué si et seulement si
At est une conséquence logique trivaluée de Ax.

Démonstration (1)$\Leftrightarrow$(2) est clair.

At est une conséquence logique trivaluée de Ax implique que Ax $\cup$ {$\neg$At} n'est pas fortement consistant au sens trivalué.
En effet soit $\mathcal{U}$ un modèle de Ax. $\forall\phi$, $vv_\phi(p(t_1, ..., t_n)) = V$ et donc $vv_{\mathcal{U}}(\neg p(t_1, ..., t_n)) = F$

La réciproque est fausse comme le montre l'exemple suivant:

Soit Ax l'ensemble:

$$\{p(x_1, ..., x_n) \rightarrow q(x_1, ..., x_n), p(x_1, ..., x_n) \rightarrow \neg q(x_1, ..., x_n), \neg p(x_1, ..., x_n) \rightarrow q(x_1, ..., x_n), \neg p(x_1, ..., x_n) \rightarrow \neg q(x_1, ..., x_n)\ \};$$

Ax $\cup$ {$\neg p(x_1, ..., x_n)$} n'est pas fortement consistant au sens trivalué alors que l'on n'a pas Ax $\models_T p(t_1, ..., t_n)$, puisque l'interprétation dans laquelle p et q sont interprétés par les fonctions constamment égales à I est un modèle trivalué fort de Ax et non de $p(t_1, ..., t_n)$. ♦

Proposition 2.2.5 *Soient for_1 et for_2 deux formules de FOR(L(*)).*

a) $for_1 \equiv_T for_2$ implique $for_1 =\!|_T\!\models for_2$. La réciproque est fausse.
b) $for_1 \equiv_T for_2$ équivaut à $\models_T (for_1 \leftrightarrow for_2)$.
c) $\models_T (for_1 \leftarrow\!\rightarrow for_2)$ implique $for_1 =\!|_T\!\models for_2$.
Si on suppose de plus for_1 et for_2 closes,
$for_1 =\!|_T\!\models for_2$ implique $\models_T (for_1 \leftarrow\!\rightarrow for_2)$.
d) $for_1 \equiv_B for_2$ équivaut à $\models_B (for_1 \Leftrightarrow for_2)$.
$\models_B (for_1 \Leftrightarrow for_2)$ implique $for_1 =\!|_B\!\models for_2$.
Si on suppose de plus for_1 et for_2 closes,
$for_1 =\!|_B\!\models for_2$ implique $\models_B (for_1 \Leftrightarrow for_2)$.

Démonstration a) L'implication est claire.
Pour montrer que la réciproque est fausse, considérons les deux formules:

$$(p(x_1, ..., x_n) \leftrightarrow q(x_1, ..., x_n)) \wedge (p(x_1, ..., x_n) \vee \neg p(x_1, ..., x_n)) \text{ et}$$

$$(p(x_1, ..., x_n) \Leftrightarrow q(x_1, ..., x_n)).$$

On a bien:

$$(p(x_1, ..., x_n) \leftrightarrow q(x_1, ..., x_n)) \wedge (p(x_1, ..., x_n) \vee \neg p(x_1, ..., x_n)) =\!|_T\!\models (p(x_1, ..., x_n) \Leftrightarrow q(x_1, ..., x_n)).$$

En effet un modèle trivalué fort de:

$$(p(x_1, ..., x_n) \leftrightarrow q(x_1, ..., x_n)) \wedge (p(x_1, ..., x_n) \vee \neg p(x_1, ..., x_n))$$

est tel que p et q soient interprétés par la même fonction à valeurs dans $\{V, F\}$, de même pour un modèle de $(p(x_1, ..., x_n) \Leftrightarrow q(x_1, ..., x_n))$. Cependant, on n'a pas:

$$(p(x_1, ..., x_n) \leftrightarrow q(x_1, ..., x_n)) \wedge (p(x_1, ..., x_n) \vee \neg p(x_1, ..., x_n)) \equiv_T (p(x_1, ..., x_n) \Leftrightarrow q(x_1, ..., x_n)).$$

Il suffit de prendre une interprétation dans laquelle p est interprété la fonction constante = I et q par la fonction constante = F.

b) Par définition de $\equiv_T$.

c) $\models_T (for_1 \leftarrow\rightarrow for_2)$ équivaut à $\models_T (for_1 \rightarrow for_2)$ et
$\models_T (for_2 \rightarrow for_1)$,
ce qui équivaut à $for_1 \models_T for_2$ et $for_2 \models_T for_1$, si for_1 et for_2 sont des formules closes (cf. la remarque ci-dessus), ce qui équivaut à:
$$for_1 =\!\mid_T\,\models for_2.$$

d) Clair.♦

Remarque: $for_1 =\!\mid_T\,\models for_2$ implique $for_1 =\!\mid_B\,\models for_2$ mais la réciproque est fausse. Par exemple:

$$(p(x_1, ..., x_n) \rightarrow q(x_1, ..., x_n)) \wedge (p(x_1, ..., x_n) \rightarrow \neg q(x_1, ..., x_n)) \wedge$$
$$(\neg p(x_1, ..., x_n) \rightarrow q(x_1, ..., x_n)) \wedge (\neg p(x_1, ..., x_n) \rightarrow \neg q(x_1, ..., x_n))$$

$$=\!\mid_B\,\models$$

$$(q(x_1, ..., x_n) \rightarrow p(x_1, ..., x_n)) \wedge (q(x_1, ..., x_n) \rightarrow \neg p(x_1, ..., x_n)) \wedge$$
$$(\neg q(x_1, ..., x_n) \rightarrow p(x_1, ..., x_n)) \wedge (\neg q(x_1, ..., x_n) \rightarrow \neg p(x_1, ..., x_n)),$$

puisque ces deux expressions n'ont aucun modèle bivalué.
Alors que l'interprétation dans laquelle p est interprété par la fonction constante égale à I et q par la fonction constante égale à V est un modèle trivalué fort du premier et non du second.

Nous allons maintenant présenter un cas particulier important d'interprétation auquel nous nous limiterons dans presque toute la suite (sauf dans la démonstration du théorème de complétude en calcul des prédicats):

2.3 LES INTERPRETATIONS DE HERBRAND TRIVALUEES

Ce sont des interprétations, dans lesquelles le domaine D est l'ensemble de tous les termes clos, les constantes et les symboles de fonctions sont interprétés par eux-mêmes, le prédicat égalité est interprété par l'égalité syntaxique et il ne reste qu'à donner une interprétation à chaque prédicat n-aire (une fonction de l'ensemble: $\{\text{termes clos}\}^n$ dans l'ensemble: $\{V, F, I\}$).

2.3.1 Syntaxe.

Définition 2.3.1 La base de Herbrand d'un langage L, her(L) est l'ensemble de tous les atomes clos de L sauf ceux de la forme: $t_1 = t_2$.
On note par Her(L) l'ensemble: her(L) $\cup$ $\neg$her(L).
L'univers de Herbrand Uni(L) est l'ensemble de tous les termes clos de L.

Définition 2.3.2 Une interprétation de Herbrand trivaluée est un sous-ensemble i de Her(L) tel que:

$$\text{ato} \in i \text{ implique } \neg\text{ato} \notin i.$$

Une interprétation de Herbrand bivaluée est une interprétation de Herbrand trivaluée telle que:

$$\forall \text{ato} \in \text{her}(L), \text{ ato} \in i \text{ ou } \neg\text{ato} \in i.$$

On note IHT(L) l'ensemble de toutes les interprétations de Herbrand trivaluées sur L.
On note IHT^∞(L) l'ensemble IHT(L) $\cup$ {Contra}
On note IHB(L) l'ensemble de toutes les interprétations de Herbrand bivaluées sur L.

Définition 2.3.3 On définit les ordres suivants sur IHT^∞(L) et IHB(L):

$$\forall i \in \text{IHT}^\infty(L),\ i \leq \text{Contra}.$$
$$\forall i, j \in \text{IHT}(L),\ i \leq j \Leftrightarrow i \subseteq j$$

$$\forall i, j \in \text{IHB}(L),\ i \leq j \Leftrightarrow \text{pos}(i) \subseteq \text{pos}(j),$$

où pos est l'application définie par:

$$\text{pos: IHB}(L) \longrightarrow \text{her}(L),$$
$$i \longrightarrow \text{pos}(i) = \{\text{ato} \in \text{her}(L) \cap i\}.$$

L'application pos est un isomorphisme de IHB(L) sur her(L).

Habituellement une interprétation de Herbrand bivaluée n'est définie qu'avec sa partie positive. Cette définition là nous permet de considérer une interprétation de Herbrand bivaluée comme un cas particulier d'interprétation trivaluée.

Nous pouvons rapidement donner la valeur de vérité d'une formule close relativement à une interprétation de Herbrand trivaluée (ce qui donne ensuite la valeur de vérité d'une formule quelconque relativement à une assignation ϕ puisque toute assignation ϕ est à valeurs dans UNI(L)).

Définition 2.3.4 Soit $i \in$ IHT(L). On définit vv_i: {Formules closes de L} $\longrightarrow$ {V, F, I} inductivement de la façon suivante:.

- Si $f \in$ her(L), $f = p(t_1, t_2, ..., t_n)$ où p n'est pas le prédicat égalité, alors $vv_i(f)$ = Vrai si et seulement si $f \in i$, Faux si et seulement si $\neg f \in i$, Indéterminé sinon.

- Si $f = (t_1 = t_2)$, alors $vv_i(f)$ = Vrai si et seulement si t_1 et t_2 sont <u>syntaxiquement identiques</u>, Faux sinon.

- Si $f = \neg g$ où g est une formule close du langage L alors $vv_i(f) = \neg vv_i(g)$ où
$\neg$: {V, F, I} $\rightarrow$ {V, F, I} a la table de vérité ci-dessus (2.1.4).

- Si $f = (g \gamma h)$ où g, h sont des formules closes et γ est un connecteur binaire, alors $vv_i(f) = \gamma(vv_i(g), vv_i(h))$ où γ: $\{V, F, I\}^2 \rightarrow \{V, F, I\}$ a la table de vérité ci-dessus(2.1.4) pour chaque connecteur binaire γ.

- Si $f = \forall x\ g$, alors $vv_i(f)$ = Vrai si et seulement si pour tout t élément de UNI(L), $vv_i(g(t))$ = Vrai, $vv_i(f)$ = Faux si et seulement si il existe t appartenant à UNI(L) tel que $vv_i(g(t))$ = Faux, $vv_i(f)$ = Indéterminé sinon.

- Si $f = \exists x\ g$, $vv_i(f)$ = Vrai si et seulement si $\exists t \in$ UNI(L), $vv_i(g(t))$ = Vrai, $vv_i(f)$ = Faux si et seulement si $\forall t \in$ UNI(L), $vv_i(g(t))$ = Faux, Indéterminé sinon.

- Dans ces deux derniers cas, on peut définir $vv_i(f)$ à l'aide des fonctions:
$\wedge$: $\{V, F, I\}^{UNI(L)} \longrightarrow \{V, F, I\}$ et $\vee$: $\{V, F, I\}^{UNI(L)} \longrightarrow \{V, F, I\}$,
avec $\wedge(h) = V$ (resp. $\vee(h) = V$) si et seulement si $\forall t \in$ UNI(L) (resp. $\exists t \in$ UNI(L)), $h(t) = V$,

$\wedge(h) = F$ (resp. $\vee(h) = F$) si et seulement si $\exists t \in UNI(L)$ (resp. $\forall t \in UNI(L)$), $h(t) = F$,

$\wedge(h) = I$ (resp. $\vee(h) = I$) sinon,

où h est l'application définie par: $h: UNI(L) \longrightarrow \{V, F, I\}$, $h(t) = vv_i(g(t))$ ($f = \exists x\ g$ ou $\forall x\ g$).

On peut limiter les notions de modèles, de tautologies et d'équivalences trivaluées aux interprétations de Herbrand.

On obtient alors les définitions suivantes:

2.3.2 Modèles.

Définition 2.3.5 Soit Ax un ensemble de formules closes du langage L. Une interprétation $i \in IHT(L)$ est un modèle de Herbrand trivalué fort (resp. faible) de Ax si et seulement si $\forall f \in Ax$, $vv_i(f) =$ Vrai (resp. $\neq$ Faux).

Si Ax a un modèle de Herbrand trivalué fort (resp. faible), alors Ax est fortement (resp. faiblement) consistant au sens de Herbrand trivalué.

Si $i \in IHB(L)$, alors les deux notions coïncident et l'on définit:

Définition 2.3.6 Une interprétation $i \in IHB(L)$ est un modèle de Herbrand bivalué de Ax si et seulement si $\forall f \in Ax$, $vv_i(f) =$ Vrai.

Si Ax a un modèle de Herbrand bivalué, alors Ax est consistant au sens de Herbrand bivalué.

Relations entre toutes ces notions.

On reprend les mêmes contre-exemples que dans le cas propositionnel, A et B désignant deux atomes clos de ATO(L).

Un modèle de Herbrand trivalué fort de Ax est un modèle de Herbrand trivalué faible; par conséquent, si Ax est fortement consistant au sens de Herbrand trivalué, alors il est faiblement consistant au sens de Herbrand trivalué.

Un modèle de Herbrand bivalué de Ax est un modèle de Herbrand trivalué fort de Ax; par conséquent, si Ax est consistant au sens de Herbrand bivalué, alors il est fortement consistant au sens de Herbrand trivalué.

Remarques: - $\emptyset$ n'est pas toujours un modèle de Herbrand trivalué faible d'une formule. Ce n'est pas, par exemple un modèle de Herbrand trivalué faible de $(A \to B) \to A$

- Un ensemble de formules ne possède pas nécessairement de modèle de Herbrand trivalué faible: par exemple, la formule: $((A \wedge \neg A) \to B) \to (B \wedge \neg B)$. Dans n'importe quel modèle de Herbrand trivalué faible, $A \wedge \neg A$ est soit Indéterminé, soit Faux, et $(A \wedge \neg A) \to B$ est toujours Vrai tandis que $B \wedge \neg B$ n'est jamais Vrai.

- Aucune autre implication n'est vraie:

- Si $Ax = \{A, \neg A\}$, alors $\emptyset$ est un modèle de Herbrand trivalué faible de Ax sans être un modèle de Herbrand trivalué fort. Ax est consistant au sens de Herbrand trivalué faible sans l'être au sens de Herbrand trivalué fort.

- Même un modèle de Herbrand trivalué fort maximal de Ax n'est pas nécessairement un modèle de Herbrand bivalué (alors que les interprétations de Herbrand trivaluées maximales sont les interprétations de Herbrand bivaluées).

Par exemple, soit Ax l'ensemble $\{A \to B, A \to \neg B, \neg A \to \neg B, \neg A \to B\}$. Ax est fortement consistant au sens de Herbrand trivalué sans l'être au sens bivalué. Les ensembles $\emptyset$, $\{B\}$ et $\{\neg B\}$ sont des modèles de Herbrand trivalués forts de Ax, les deux derniers étant des modèles maximaux sans être des modèles bivalués.

Cependant, on a la proposition suivante:

Proposition 2.3.1 *Une interprétation $i \in IHT(L)$ est un modèle trivalué fort de $Ax' = Ax \cup \{ato \vee \neg ato / \ ato \in her(L)\}$ si et seulement si $i \in IHB(L)$ et i est un modèle bivalué de Ax.*

Démonstration Si $i \in IHT(L)$ est un modèle trivalué fort de $\{ato \vee \neg ato / \ ato \in her(L)\}$ alors $\forall ato \in her(L)$, $ato \in i$ ou $\neg ato \in i$, et $i \in IHB(L)$.
Réciproquement, si $i \in IHB(L)$, alors $vv_i(ato \vee \neg ato) = Vrai$, pour tout ato de her(L).♦

Tautologies et conséquences:

Definition 2.3.7 Soit A une formule de L.
On note par:

$$\models_{TH} A$$

le fait que A soit Vraie dans toute interprétation de Herbrand trivaluée (i.e. sa clôture universelle est Vraie dans toute interprétation de Herbrand trivaluée).

On dit que A est une tautologie de Herbrand trivaluée.

(On a de même les notions de tautologie de Herbrand trivaluée faible: A n'est jamais Faux dans une interprétation de Herbrand trivaluée, ce qui se note: $\models_{tH}$ A et de tautologie de Herbrand bivaluée, ce qui se note: $\models_{BH}$ A).

Lemme 2.3.1 *On a, si A est une formule à n variables libres, $x_1, ..., x_n$,*

$$\models_{TH} A(x_1, ..., x_n) \Leftrightarrow$$
$$\forall i \in IHT(L),\ \forall \phi,\ vv_{\phi}(A(x_1, ..., x_n)) = V \Leftrightarrow$$
$$\forall i \in IHT(L),\ \forall (t_1, ..., t_n) \in UNI(L)^n,\ vv_i(A(t_1, ..., t_n)) = V.$$

Definition 2.3.8 - Soient Ax et Bx deux ensembles de formules closes. Bx est conséquence de Herbrand de Ax, ce qui se note:

$$Ax \models_{TH} Bx$$

si et seulement si tout modèle trivalué fort de Ax est un modèle trivalué fort de Bx.
(De la même façon, on définit Ax $\models_{tH}$ Bx et Ax $\models_{BH}$ Bx).

Remarque: - Si Ax $\models_{TH}$ Bx, alors Ax $\models_{BH}$ Bx.

- Si Ax $\models_{tH}$ Bx, alors Ax $\models_{BH}$ Bx.

Aucune autre implication n'est vraie:

- Si Ax $\models_{BH}$ Bx, on n'a pas Ax $\models_{TH}$ Bx.

Par exemple, prenons pour Ax l'ensemble vide et pour Bx l'ensemble $\{A \vee \neg A\}$. Toute interprétation de Herbrand bivaluée est un modèle de Ax et de Bx alors que $\varnothing$ est un modèle de Herbrand trivalué fort de Ax et non de Bx.

- Si Ax $\models_{BH}$ Bx, on n'a pas Ax $\models_{tH}$ Bx.
Par exemple, prenons pour Ax l'ensemble $\{A, \neg A\}$ et pour Bx l'ensemble $\{B\}$. Ax ne possède pas de modèle de Herbrand bivalué tandis que $\{\neg B\}$ est un modèle de Herbrand trivalué faible de Ax et non de Bx.

- Si Ax $\models_{TH}$ Bx, on n'a pas Ax $\models_{tH}$ Bx.
Par exemple, prenons pour Ax l'ensemble $\{A, A \rightarrow B\}$ et pour Bx l'ensemble $\{B\}$. Le seul modèle de Herbrand trivalué fort de Ax est $\{A, B\}$; c'est un modèle de Herbrand trivalué fort de Bx alors que $\{\neg B\}$ est un modèle de Herbrand trivalué faible de Ax et non de Bx.

- Si Ax $\models_{tH}$ Bx, on n'a pas Ax $\models_{TH}$ Bx.
Par exemple, prenons pour Ax l'ensemble $\{A \leftrightarrow \neg A\}$ et pour Bx l'ensemble $\{B\}$. Le seul modèle de Herbrand trivalué faible de Ax est $\varnothing$ qui est un modèle de Herbrand trivalué faible de Bx. Le seul modèle de Herbrand trivalué fort de Ax est $\varnothing$ qui n'est pas un modèle trivalué fort de Bx.

Définition 2.3.9 Soient A et B deux formules de L.

Si $\models_{TH} A \leftrightarrow B$, on a:

$$A \equiv_{TH} B$$

Lemme 2.3.2 *On a, si A et B sont deux formules ayant leurs variables libres dans l'ensemble* $\{x_1, ..., x_n\}$*:*

$$\begin{aligned} & A(x_1, ..., x_n) \equiv_{TH} B(x_1, ..., x_n) \Leftrightarrow \\ & \forall i \in IHT(L),\ \forall \phi,\ vv_\phi(A(x_1, ..., x_n) \leftrightarrow B(x_1, ..., x_n)) = V \Leftrightarrow \\ & \forall i \in IHT(L),\ \forall (t_1, ..., t_n) \in UNI(L)^n, \\ & \qquad vv_i(A(t_1, ..., t_n)) = vv_i(B(t_1, ..., t_n)). \end{aligned}$$

On va établir dans ce dernier paragraphe un théorème de Herbrand trivalué, c'est-à-dire une relation entre les ensembles consistants au sens de Herbrand et les ensembles consistants.

2.4 UN THEOREME DE HERBRAND TRIVALUE

Définitions 2.4.1 Une règle trivaluée est de la forme:

lit $\leftarrow$ for

où for est une formule ne contenant que les connecteurs $\Rightarrow$, $\Leftrightarrow$, $\neg$, $\vee$, $\wedge$, et les quantificateurs $\exists$, $\forall$.

Une clause trivaluée est de la forme:

$lit_{i_1} \vee ... \vee lit_{i_p} \leftarrow lit_{j_1} \wedge ... \wedge lit_{j_n}$

Une clause définie trivaluée est de la forme:

$lit \leftarrow lit_{j_1} \wedge ... \wedge lit_{j_n}$

On a le théorème suivant:

Théorème 2.4.1

Soit S un ensemble de clauses trivaluées:

$$lit_{i_1} \vee ... \vee lit_{i_p} \leftarrow lit_{j_1} \wedge ... \wedge lit_{j_n}.$$

On a l'équivalence suivante:

S a un modèle trivalué ⇔ S a un modèle de Herbrand.

Démonstration La démonstration est basée sur le fait qu'à une interprétation trivaluée i, on peut associer une interprétation de Herbrand H, définie par:
Si $p(t_1, ..., t_n)$ est un atome clos, $vv_H(p(t_1, ..., t_n)) = vv_i(p(t_1, ..., t_n))$
Soit I un modèle trivalué de S. I est formé d'atomes et de négations d'atomes (avec variables) et ne contient jamais un atome et sa négation.
Soit U_S l'univers de Herbrand associé à S , formé des constantes de S et de tous les termes clos fabriqués à partir des constantes et des symboles de fonctions de S.
Soit B_S la base de Herbrand associée à S , formée de toutes les formules atomiques closes de S.
Soit I' l'ensemble:

$\{p(t_1, ..., t_n) \in B_S /\ p(t_1, ..., t_n)$ soit Vrai relativement à I$\} \cup \{\neg q(t'_1, ..., t'_p) /\ q(t'_1, ..., t'_p) \in B_S$ et $\neg q(t'_1, ..., t'_p)$ soit Vrai relativement à I$\}$.

Montrons que si I est un modèle trivalué de S, alors I' est un modèle de Herbrand trivalué de S.

Soit F une formule de S, de la forme:

$lit_{i_1} \vee ... \vee lit_{i_p} \leftarrow lit_{j_1} \wedge ... \wedge lit_{j_n}.$

ayant pour variables libres $x_1, ..., x_s$.
Soit ϕ une assignation de variables, c'est-à-dire une application de Var dans U_S.

$vv_{\phi}(F) = \text{Vrai} \Leftrightarrow [\forall(d_1, ..., d_s) \in (U_S)^s, (\forall k = 1, ..., n,$ $vv_{\phi(x_1\leftarrow d_1)...(x_s\leftarrow d_s)}(lit_{jk}) = \text{Vrai})$ implique $\exists k \in \{1, ..., p\}$ tel que $vv_{\phi(x_1\leftarrow d_1)...(x_s\leftarrow d_s)}(lit_{ik}) = \text{Vrai}]$.
Si lit_{jk} est de la forme: $lit_{jk}(t_1, ..., t_q)$
$\forall k = 1, ..., n$, $vv_{\phi(x_1\leftarrow d_1)...(x_s\leftarrow d_s)}(lit_{jk}) = \text{Vrai}$ implique $\forall k = 1, ..., n$, $vv_{I'}(lit_{jk}(t'_1, ..., t'_q)) = V$, où $t'_j = t_j\ (x_1\leftarrow d_1, ..., x_s\leftarrow d_s)$.

Avec la définition de I', cela implique que $\forall k = 1, ..., n$,
$vv_I(lit_{jk}(t'_1, ..., t'_q)) = V$, et comme I est un modèle, $\exists k \in \{1, ..., p\}$
tel que $vv_I(lit_{ik}(t'_1, ..., t'_q)) = V$, d'où en réutilisant la définition de I',
$\exists k \in \{1, ..., p\}$ tel que $vv_{I'}(lit_{ik}(t'_1, ..., t'_q)) = V$, ou encore
$\exists k \in \{1, ..., p\}$ tel que $vv_{\phi(x_1\leftarrow d_1)...(x_s\leftarrow d_s)}(lit_{ik}) = \text{Vrai}$,
et I' est un modèle de Herbrand de S. ♦

Remarque: De même que dans le cas bivalué, si on enlève l'hypothèse clause, ce théorème n'est plus vrai.

Prenons par exemple pour S l'ensemble $\{p(a), \exists x\ \neg p(x)\}$.

Soit $\mathcal{U}$ l'interprétation ayant pour domaine $\{\alpha, \beta\}$, dans laquelle on interprète la constante a par α et le prédicat p par l'application π: $\alpha \rightarrow V$, $\beta \rightarrow F$.

$\mathcal{U}$ est un modèle de S qui n'admet pas de modèle de Herbrand (parce que dans un modèle de Herbrand, on ne peut pas ajouter de constantes supplémentaires).

Dans le chapitre suivant, nous étudions l'expressivité des differents connecteurs.

CHAPITRE 3: EXPRESSIVITE DES CONNECTEURS

3.1 INTRODUCTION

Nous étudions ici l'expressivité des différents connecteurs en logique trivaluée.
Ceci peut être utile dans la pratique, pour construire des systèmes experts à base de règles qui contiennent d'autres symboles que le $\neg$, $\wedge$, $\vee$, $\rightarrow$ prévus.

Le formalisme utilisé est celui introduit en 2.1, auquel nous rajoutons des connecteurs supplémentaires pour des facilités d'écriture.

Nous nous limitons ici aux formules du calcul propositionnel utilisant les connecteurs usuels de la logique bivaluée classique $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$, auxquels nous ajoutons les constantes V, F, I, le connecteur $\rightarrow$, et les connecteurs $\leftrightarrow$, $\leftarrow\rightarrow$ et Vr, Fx, Ind qui en découlent.

Nous noterons de la même façon les connecteurs n-aires et les opérateurs de $\{V, F, I\}^n$ dans $\{V, F, I\}$ qui leur sont associés.

Dans une première partie, nous étudierons l'expressivité des connecteurs et nous en déduirons un système générateur de connecteurs pour exprimer une formule générale du langage. A chaque formule du langage, correspondra une formule construite avec les mêmes atomes et des connecteurs de ce système.

Dans une deuxième partie nous étudierons directement les fonctions de $\{V, F, I\}^n$ dans $\{V, F, I\}$ et nous donnerons des systèmes minimaux de connecteurs permettant de les exprimer. Nous obtiendrons donc un résultat plus général que précédemment.

Dans une dernière partie, nous caractérisons les fonctions monotones de $\{V, F, I\}^n$ dans $\{V, F, I\}$ telles que $f(\{V, F\}^n) \subset \{V, F\}$ et $f(I, ..., I) = I$.

Nous montrons qu'une telle fonction peut ne s'écrire qu'avec des $\neg$ et des $\vee$ et nous donnons l'algorithme permettant de trouver la fonction avec des $\neg$ et des $\vee$ qui lui est associée.

3.2 FORMALISME

Le formalisme utilisé est le même qu'en 2.1, sauf que l'on a rajouté les connecteurs 0-aires V, F, I, et les connecteurs unaires Vr, Fx, Ind.

Définition 3.2.1

Soit $\mathcal{P}$ l'ensemble des variables propositionnelles (atomes).

Soit $\mathcal{A} = \{(,)\}$, l'ensemble des symboles de ponctuation

Soit $\mathbf{Const} = \{V, F, I\}$ l'ensemble des constantes.

Soit $\mathcal{C}_1 = \{\neg, \mathrm{Ind}, \mathrm{Fx}, \mathrm{Vr}\}$ l'ensemble des connecteurs unaires.

Soit $\mathcal{C}_2 = \{\wedge, \vee, \Rightarrow, \Leftrightarrow, \rightarrow, \leftrightarrow, \leftarrow\rightarrow\}$ l'ensemble des connecteurs binaires.

L'ensemble $\mathcal{F}(\mathcal{P})$ des formules dont les atomes sont dans $\mathcal{P}$, est défini inductivement comme suit:

$\mathbf{Const} \subset \mathcal{F}(\mathcal{P})$.

$\mathcal{P} \subset \mathcal{F}(\mathcal{P})$.

$A \in \mathcal{F}(\mathcal{P})$ implique $\mathrm{Ind}(A)$, $\mathrm{Vr}(A)$, $\mathrm{Fx}(A)$, $\neg A \in \mathcal{F}(\mathcal{P})$.

$A, B \in \mathcal{F}(\mathcal{P})$ implique $(A\gamma B) \in \mathcal{F}(\mathcal{P})$ si $\gamma \in \{\wedge, \vee, \Rightarrow, \Leftrightarrow, \rightarrow, \leftrightarrow, \leftarrow\rightarrow\}$.

Si $f \in \mathcal{F}(\mathcal{P})$ et a pour atomes $A_1, ..., A_n$, on note f par $f(A_1, ..., A_n)$.

Remarque:

$\mathcal{F}(\mathcal{P})$ peut se définir par un procédé constructif:

$F_0 = \mathcal{P} \cup \mathbf{Const}$.
$F_{n+1} = F_n \cup \{Ind(A), Vr(A), Fx(A), \neg A / A \in F_n\} \cup \{A\gamma B / A, B \in F_n \text{ et } \gamma \in \{\wedge, \vee, \Rightarrow, \Leftrightarrow, \rightarrow, \leftrightarrow, \leftarrow\rightarrow\}\}$.
$\mathcal{F}(\mathcal{P}) = \bigcup_{n \in \mathcal{N}} F_n$.

Définition 3.2.2 La valeur de vérité d'une formule relativement à i se définit de la manière suivante:

C'est une application vv_i de $\mathcal{F}(\mathcal{P})$ dans $\{V, F, I\}$ définie par:

- Si $A \in F_0$, et $A = p$, alors si $p \in i$, $vv_i(A) =$ Vrai (V), si $\neg p \in i$, $vv_i(A) =$ Faux (F), $vv_i(A) =$ Indéterminé (I) sinon.

Si $A \in \mathbf{Const}$, la valeur de vérité de A est indépendante de l'interprétation i:
- $A = I$, $vv_i(A) = I$.
- $A = F$, $vv_i(A) = F$.
- $A = V$, $vv_i(A) = V$.

- Si $A \in F_n$, $n > 0$, la valeur de vérité de A est donnée par les tables de vérité données en 2.1 pour les connecteurs $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow, \rightarrow, \leftrightarrow, \leftarrow\rightarrow$.

Si $A = Ind(B)$, $vv_i(A) = V \Leftrightarrow vv_i(B) = I$,
$vv_i(A) = F$ sinon.
Si $A = Vr(B)$, $vv_i(A) = V \Leftrightarrow vv_i(B) = V$,
$vv_i(A) = F$ sinon.
Si $A = Fx(B)$, $vv_i(A) = V \Leftrightarrow vv_i(B) = F$,
$vv_i(A) = F$ sinon.

Remarque: Le centre d'un connecteur étant sa restriction à $\{V, F\}^n$, $\Rightarrow$ et $\rightarrow$ ont même centre et $\Leftrightarrow$, $\leftrightarrow$, $\leftarrow\rightarrow$ ont même centre.

3.3. EXPRESSIVITE DES CONNECTEURS $\neg, \vee, \mathrm{I}, \rightarrow$

3.3.1 Avec les connecteurs $\neg$, $\vee$ (resp. $\neg$, $\wedge$; resp. $\neg$, $\Rightarrow$) on obtient $\neg$, $\vee$, $\wedge$, $\Rightarrow$, $\Leftrightarrow$. On ne peut pas exprimer les connecteurs I, V, F, Ind, Vr, Fx, $\rightarrow$, $\leftrightarrow$, $\leftarrow\rightarrow$.

a) Avec $\neg$, $\vee$ on obtient $\wedge$, $\Rightarrow$, $\Leftrightarrow$. En effet:

$$A \wedge B \equiv_T \neg(\neg A \vee \neg B)$$

$$A \Rightarrow B \equiv_T \neg A \vee B$$

$$A \Leftrightarrow B \equiv_T (A \Rightarrow B) \wedge (B \Rightarrow A)$$

b) Avec $\neg$, $\wedge$ on obtient $\vee$, $\Rightarrow$, $\Leftrightarrow$. En effet on obtient $\vee$, et on utilise a).

$$A \vee B \equiv_T \neg(\neg A \wedge \neg B)$$

c) Avec $\Rightarrow$, $\neg$ on obbtient $\wedge$, $\vee$, $\Leftrightarrow$. En effet on obtient $\vee$, et on utilise a).

$$A \vee B \equiv_T \neg A \Rightarrow B.$$

d) Avec $\neg$, $\vee$ (resp. $\neg$, $\wedge$; resp. $\neg$, $\Rightarrow$) on ne peut pas exprimer les connecteurs I, V, F, Ind, Vr, Fx, $\rightarrow$, $\leftrightarrow$, $\leftarrow\rightarrow$.

En effet V, F, Ind, Vr, Fx, $\rightarrow$, $\leftrightarrow$, $\leftarrow\rightarrow$ ne prennent jamais la valeur I. Or si f est une fonction de $\{V, F, I\}^n$ dans $\{V, F, I\}$ n'utilisant que $\neg$, $\vee$, on a $f(I, \ldots, I) = I$ (par réccurrence sur le nombre de connecteurs de f).

Si f est une fonction de $\{V, F, I\}^n$ dans $\{V, F, I\}$ n'utilisant que $\neg$, $\vee$, on a $f(V, \ldots, V) = V$ (par réccurrence sur le nombre de connecteurs de f); donc on ne peut obtenir la fonction constante I à partir de $\neg$, $\vee$.

3.3.2 Avec $\rightarrow$ et $\neg$, on peut obtenir Vr, Fx, Ind et $\leftarrow\rightarrow$.

$$Vr(X) \equiv_T \neg(X \rightarrow \neg X)$$

$$Fx(X) \equiv_T \neg(\neg X \rightarrow X)$$

$$Ind(X) \equiv_T \neg((X \rightarrow \neg X) \rightarrow \neg(\neg X \rightarrow X))$$

$$A \leftarrow\rightarrow B \equiv_T \neg((A \rightarrow B) \rightarrow \neg(B \rightarrow A))$$

3.3.3 Un seul des connecteurs Ind, Fx, Vr, permet de retrouver les deux autres avec $\neg$, $\vee$ (resp. $\neg$, $\wedge$; resp. $\neg$, $\Rightarrow$).

$$Vr(X) \equiv_T X\wedge\neg Ind(X)$$

$$Vr(X) \equiv_T Fx(\neg X)$$

$$Fx(X) \equiv_T Vr(\neg X)$$

$$Fx(X) \equiv_T \neg X\wedge\neg Ind(X)$$

$$Ind(X) \equiv_T \neg Vr(X\vee\neg X) \equiv_T \neg(Vr(X)\vee Vr(\neg X))$$

$$Ind(X) \equiv_T \neg Fx(X\wedge\neg X) \equiv_T \neg(Fx(X)\vee Fx(\neg X))$$

3.3.4 Un seul des connecteurs Ind, Fx, Vr, $\rightarrow$, $\leftrightarrow$, $\leftarrow\rightarrow$ permet de retrouver les deux autres avec $\neg$, $\vee$ (resp. $\neg$, $\wedge$; resp. $\neg$, $\Rightarrow$).

a) En effet d'après 3.3.2, avec $\rightarrow$, $\neg$, on peut obtenir Ind, Fx, Vr, $\leftarrow\rightarrow$. Avec $\rightarrow$, $\neg$, $\wedge$ on obtient $\leftrightarrow$ car

$$A \leftrightarrow B \equiv_T (A \rightarrow B)\wedge(B \rightarrow A)\wedge(\neg A \rightarrow \neg B)\wedge(\neg B \rightarrow \neg A)$$

b) Avec $\leftrightarrow$ et $\neg$ on peut obtenir Ind car $Ind(X) \equiv_T X \leftrightarrow \neg X$. D'après 3.3.3 on peut donc obtenir Fx et Vr avec $\leftrightarrow$, $\neg$, et $\vee$ (puisque l'on obtient aussi $\wedge$).
Avec $\leftrightarrow$, $\neg$ et $\vee$ on peut obtenir $\rightarrow$. En effet:

$$A \rightarrow B \equiv_T Fx(A)\vee Ind(A)\vee(Vr(A)\wedge Vr(B))$$

et Vr, Fx, Ind s'expriment avec $\leftrightarrow$, $\neg$, et $\vee$ d'après 3.3.2.
On peut donc obtenir $\leftarrow\rightarrow$ avec $\leftrightarrow$, $\neg$ et $\vee$ puisque

$$A \leftarrow\rightarrow B \equiv_T (A \rightarrow B)\wedge(B \rightarrow A)$$
$$(\text{ou bien } A \leftarrow\rightarrow B \equiv_T \neg((A \rightarrow B) \rightarrow \neg(B \rightarrow A)))$$

c) Avec $\leftarrow\rightarrow$, $\neg$ et $\vee$ on peut obtenir $\leftrightarrow$. En effet:

$$A \leftrightarrow B \equiv_T (A \leftarrow\rightarrow B)\wedge(\neg A \leftarrow\rightarrow \neg B)$$

par conséquent, on peut obtenir Ind, Fx, Vr, $\rightarrow$ d'après ci-dessus.

d) Avec Ind, $\neg$ et $\vee$ on peut obtenir Vr et Fx (3.3.3), puis $\rightarrow$ car

$$A \rightarrow B \equiv_T Fx(A) \vee Ind(A) \vee (Vr(A) \wedge Vr(B))$$

puis $\leftrightarrow$ et $\leftarrow\rightarrow$ (a).

e) De même avec Fx, $\neg$ et $\vee$.

f) De même avec Vr, $\neg$ et $\vee$.

3.3.5 Avec $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$, $\rightarrow$, $\leftrightarrow$, $\leftarrow\rightarrow$, Vr, Fx, Ind, V, F on ne peut pas obtenir la constante I. La constante I fera donc partie du système minimal de connecteurs.

Si f est une fonction de $\{V, F, I\}^n$ dans $\{V, F, I\}$ utilisant tous ces connecteurs, on a:

f(V, ..., V) = V ou F (par récurrence sur le nombre de connecteurs de f); donc on ne peut obtenir la fonction constante I à partir de $\neg$, $\vee$, $\Rightarrow$, $\Leftrightarrow$, $\rightarrow$, $\leftrightarrow$, $\leftarrow\rightarrow$, Vr, Fx, Ind, V, F.

On obtient alors la proposition suivante:

Proposition 3.3.1 *Pour toute formule $f(A_1, ..., A_n)$ de $\mathcal{F}(\mathcal{P})$ ayant pour atomes $A_1, ..., A_n$, il existe une formule $g(A_1, ..., A_n)$ de $\mathcal{F}(\mathcal{P})$ ayant pour atomes $A_1, ..., A_n$, et ayant ses connecteurs dans l'ensemble $\{\rightarrow, \neg, \wedge, I\}$ logiquement équivalente à $f(A_1, ..., A_n)$.*

Démonstration Par induction sur la formule $f(A_1, ..., A_n)$.

Si f est un atome c'est terminé.

Si f est une constante:

Soit f = I et c'est terminé.
Soit f = V, alors $f \equiv_T I \rightarrow I$.
Soit f = F, alors $f \equiv_T (I \rightarrow I) \rightarrow I$.

Si f n'est ni une constante ni un atome, on procède par induction.

Si f = Vr(X), Fx(X), Ind(X), $A \leftarrow\rightarrow B$, on utilise:

$$Vr(X) \equiv_T \neg(X \rightarrow \neg X)$$
$$Fx(X) \equiv_T \neg(\neg X \rightarrow X)$$

$$\mathrm{Ind}(X) \equiv_T \neg((X \to \neg X) \to \neg(\neg X \to X))$$
$$A \leftarrow\rightarrow B \equiv_T \neg((A \to B) \to \neg(B \to A))$$

et on applique l'hypothèse d'induction à X, A, B.

Si f = $A \leftrightarrow B$, $A \vee B$, $A \Rightarrow B$, $A \Leftrightarrow B$, on utilise:

$$A \leftrightarrow B \equiv_T (A \to B) \wedge (B \to A) \wedge (\neg A \to \neg B) \wedge (\neg B \to \neg A)$$
$$A \vee B \equiv_T \neg(\neg A \wedge \neg B)$$
$$A \Rightarrow B \equiv_T \neg A \vee B$$
$$A \Leftrightarrow B \equiv_T (A \Rightarrow B) \wedge (B \Rightarrow A)$$

et on applique l'hypothèse d'induction à A et à B.

Si f = $A \to B$, $A \wedge B$, on applique l'hypothèse d'induction à A et B. ♦

Remarques: 1) On peut remplacer dans ce système ($\neg$, $\wedge$, $\to$, I):

- $\wedge$ par $\vee$ (3.3.1 b)
- $\wedge$ par $\Rightarrow$ (3.3.1 c)
- $\to$ successivement par Ind, Vr, Fx, $\leftrightarrow$, $\leftarrow\rightarrow$ (3.3.4).

2) La constante I fait forcément partie du système minimal de connecteurs.

3) On verra dans le paragraphe suivant que ($\neg$, $\wedge$, $\to$, I) est un système minimal de connecteurs.

3.4 UN SYSTEME GENERATEUR DE FONCTIONS DE $\{V, F, I\}^n$ DANS $\{V, F, I\}$

Définition 3.4.1 Soient (a_{ik}) les éléments de $\{V, F, I\}^n$, $1 \leq i \leq 3$, $1 \leq k \leq n$, et f_{ik} les fonctions de $\{V, F, I\}^n$ dans $\{V, F, I\}$ définies par:

$f_{ik}(a_j) = \mathrm{Vr}(a_j)$ si $a_{ik} = V$, $\mathrm{Fx}(a_j)$ si $a_{ik} = F$, $\mathrm{Ind}(a_j)$ si $a_{ik} = I$.

On a alors $f_{ik}(a_j) = V$ si $a_j = a_{ik}$, F sinon.

Proposition 3.4.2 *Soit f une fonction de $\{V, F, I\}^n$ dans $\{V, F, I\}$.*

a) Si $\forall(a_1, ..., a_n) \in \{V, F, I\}^n, f(a_1, ..., a_n) = F$, *alors:*

$f(a_1, ..., a_n) = g(a_1, ..., a_n)$ *avec*

$g(a_1, ..., a_n) = (((a_1\wedge\neg a_1)\wedge...\wedge(a_n\wedge\neg a_n)) \rightarrow a_n) \rightarrow (a_n\wedge\neg a_n)$.

b) Sinon $f(a_1, ..., a_n) = g(a_1, ..., a_n)$ *avec*

$$g(a_1, ..., a_n) =$$
$$(\vee_{\{(i_1,i_2,...,i_n)/\ f(a_{i_1},a_{i_2},...,a_{i_n}) = I\}} (\wedge_{k = 1,...,n} f_{i_k}(a_k)\wedge I))$$
$$\vee(\vee_{\{(i_1,i_2,...,i_n)/\ f(a_{i_1},a_{i_2},...,a_{i_n}) = V\}} (\wedge_{k = 1,...,n} f_{i_k}(a_k))).$$

Démonstration a) $\forall (a_1, ..., a_n) \in \{V, F, I\}^n$, $((a_1\wedge\neg a_1)\wedge...\wedge(a_n\wedge\neg a_n)) \rightarrow a_n$ prend la valeur V et $(a_n\wedge\neg a_n)$ prend la valeur F ou I, donc la fonction g du a) est constamment égale à F.

b) Soit $(b_1, ..., b_n)$ tel que $f(b_1, ..., b_n) = F$.

Alors $\forall(i_1, ..., i_n)$ tel que $f(a_{i_1}, ..., a_{i_n}) = I$, $\exists j/\ b_j \neq a_{i_j}$ et $f_{i_j}(b_j) = F$.

De même, $\forall(i_1, ..., i_n)$ tel que $f(a_{i_1}, ..., a_{i_n}) = V$, $\exists j/\ b_j \neq a_{i_j}$ et $f_{i_j}(b_j) = F$.
Donc $g(b_1, ..., b_n) = F$.

Soit $(b_1, ..., b_n)$ tel que $f(b_1, ..., b_n) = V$.

Alors $\exists(i_1, ..., i_n)$ tel que $f(a_{i_1}, ..., a_{i_n}) = V$, avec $(a_{i_1}, ..., a_{i_n}) = (b_1, ..., b_n)$. La deuxième partie de la disjonction définissant g prend la valeur V alors que la première prend la valeur F. Donc $g(b_1, ..., b_n) = V = f(b_1, ..., b_n)$.

Soit $(b_1, ..., b_n)$ tel que $f(b_1, ..., b_n) = I$.

Alors $\exists(i_1, ..., i_n)$ tel que $f(a_{i_1}, ..., a_{i_n}) = I$, avec $(a_{i_1}, ..., a_{i_n}) = (b_1, ..., b_n)$. La première partie de la disjonction définissant g prend la valeur I alors que la deuxième prend la valeur F.
Donc $g(b_1, ..., b_n) = I = f(b_1, ..., b_n)$. ♦

Corollaire 3.4.1 *$(\neg, \vee, \rightarrow, I)$ est un système générateur de l'ensemble des fonctions de $\{V, F, I\}^n$ dans $\{V, F, I\}$.*

Démonstration Les fonctions Vr, Fx, Ind servant à exprimer les f_{i_k} s'expriment avec $\neg$, $\rightarrow$, d'après 3.3.2 et le connecteur $\wedge$ s'exprime avec $\neg$ et $\vee$. ♦

Remarques: - La proposition 3.4.2 permet de retrouver la proposition 3.3.1. En effet, soit $f(A_1, ..., A_n)$ une formule de $\mathcal{F}(\mathcal{P})$ ayant pour atomes $A_1, ..., A_n$. A f est associée une fonction ϕ de $\{V, F, I\}^n$ dans $\{V, F, I\}$.

Si $a_1, ..., a_n$ sont les valeurs de vérité de $A_1, ..., A_n$, $\phi(a_1, ..., a_n) = g(a_1, ..., a_n)$, où g est une fonction ne comportant que les connecteurs $\neg$, $\vee$, $\rightarrow$, I. On a alors $f(A_1, ..., A_n) \equiv_T g(A_1, ..., A_n)$.

- De même que dans 3.3.1, on peut remplacer dans ce système ($\neg$, $\vee$, $\rightarrow$, I):

- $\vee$ par $\wedge$ (3.3.1 b)

- $\wedge$ par $\Rightarrow$ (3.3.1 c)

- $\rightarrow$ successivement par Ind, Vr, Fx, $\leftrightarrow$, $\leftarrow\rightarrow$ (3.3.4).

On obtient alors d'autres systèmes générateurs équivalents.

- La constante I fait forcément partie du système (3.3.5).

Proposition 3.4.3 *($\neg$, $\vee$, $\rightarrow$, I) est un système minimal pour exprimer les fonctions de $\{V, F, I\}^n$ dans $\{V, F, I\}$.*

Démonstration - On supprime le $\neg$.

Soit f une formule utilisant uniquement les connecteurs $\vee$, $\rightarrow$, I et exprimant le $\neg$.

La fonction f a au moins un connecteur car la fonction de $\{V, F, I\}$ dans $\{V, F, I\}$, qui à b, associe $\neg b$ n'est pas constante.

Elle possède forcément le connecteur $\vee$ comme connecteur de plus haut niveau car $g \rightarrow h$ ne peut prendre la valeur I alors que f peut.

On décompose f en une disjonction de formules g_i ayant la propriété suivante:

Soit $g_i = I$, soit g_i possède $\rightarrow$ comme connecteur de plus haut niveau.

On a: $f = g_1 \vee \ldots \vee g_n$ avec $n \geq 2$, d'après ci-dessus.

Si $b = I$, $f(b) = I$ et par conséquent l'un au moins des g_i vaut I, puisque les g_i possèdant $\rightarrow$ comme connecteur de plus haut niveau ne peuvent prendre la valeur I.
Si $b = V$, $f(b) = F$, or $g_1 \vee \ldots \vee g_n$ ne peut prendre la valeur F, puisque l'un des g_i vaut I.

- On supprime le $\vee$.

Soit f une formule utilisant uniquement les connecteurs $\neg$, $\rightarrow$, I et exprimant le $\vee$.

La fonction f a au moins un connecteur car la fonction de $\{V, F, I\}^2$ dans $\{V, F, I\}$, qui à (a,b) associe $a \vee b$ n'est pas constante.

Elle possède forcément le connecteur $\neg$ comme connecteur de plus haut niveau car $g \rightarrow h$ ne peut prendre la valeur I alors que f peut.

On décompose f de la manière suivante:

$f = \neg \ldots \neg(g \rightarrow h)$, le $\neg$ étant répété n fois avec $n \geq 1$ d'après ci-dessus. En effet la fonction f possède forcément le connecteur $\rightarrow$ car f ne prend pas constamment la valeur I.

Dans ce cas f ne prend jamais la valeur I.

- On supprime le I.

Si $I = f(A_1, \ldots, A_n)$, où f est une formule utilisant uniquement les connecteurs $\neg$, $\rightarrow$, $\vee$, alors en donnant à tous les A_i la valeur V, on n'obtiendra jamais la valeur I.

- On supprime le $\rightarrow$.

Supposons que $A \rightarrow B = f(A,B)$ où f est une formule ne comportant que les connecteurs $\neg$, $\vee$, I.

En donnant à A et B la valeur I, on aboutit à une contradiction. ♦

3.5. EXPRESSIVITE DES CONNECTEURS $\neg$ ET $\vee$

Dans ce paragraphe, nous allons caractériser les fonctions f ne s'exprimant qu'avec des $\neg$ et des $\vee$ par le théorème suivant:

3.5.1 Enoncé du théorème.

Théorème 3.5.1 *Soit f une fonction de $\{V, F, I\}^n$ dans $\{V, F, I\}$.*

f ne s'exprime qu'avec des $\neg$ et des $\vee$ si et seulement si:

f vérifie les trois propriétés suivantes:

1) f laisse stable $\{V, F\}$: $f(\{V, F\}^n) \subset \{V, F\}$.

2) $f(I, ..., I) = I$.

3) f est monotone de $\{V, F, I\}^n$ dans $\{V, F, I\}$, l'ordre sur $\{V, F, I\}^n$ étant l'ordre induit par l'ordre sur $\{V, F, I\}$: $I \leq V, I \leq F, I \leq I, F \leq F, V \leq V$.

3.5.2 Démonstration

a) Une fonction ne s'exprimant qu'avec des $\neg$ et des $\vee$ vérifie les trois propriétés ci-dessus: par récurrence sur le nombre k de connecteurs de la fonction.

- Si $k = 1$, $f(a_1, ..., a_n) = \neg a_i$, ou $f(a_1, ..., a_n) = a_i \vee a_j$, vérifient les propriétés 1, 2, 3.

- Si $f(a_1, ..., a_n) = \neg h(a_1, ..., a_n)$, où h ne s'exprime qu'avec des $\neg$ et des $\vee$, h vérifie les propriés 1, 2, 3 par induction et f les vérifie à cause du connecteur $\neg$.

- De même si $f(a_1, ..., a_n) = g(a_1, ..., a_n) \vee h(a_1, ..., a_n)$, par induction et à cause du connecteur $\vee$.

b) Nous allons montrer qu'une fonction vérifiant les propriétés 1, 2, 3 ne s'exprime qu'avec des $\neg$ et des $\vee$. Le résultat reste vrai si l'on fait intervenir le connecteur $\wedge$ puisqu'il ne s'exprime qu'avec des $\neg$ et des $\vee$.

Le cas $n = 1$ ne présente pas de difficultés:

Si f est une fonction de {V, F, I} dans {V, F, I}, vérifiant les propriétés 1, 2, 3, il n'existe que 4 possibilités pour f puisque f({V, F}) = {V, F} et f(I) = I.

1) A = V	A = F	A = I
f(A) = V	f(A) = F	f(A) = I

Dans ce cas f(A) = A

2) A = V	A = F	A = I
f(A) = V	f(A) = V	f(A) = I

Dans ce cas $f(A) = A \vee \neg A$

3) A = V	A = F	A = I
f(A) = F	f(A) = F	f(A) = I

Dans ce cas $f(A) = \neg(A \vee \neg A)$

4) A = V	A = F	A = I
f(A) = F	f(A) = V	f(A) = I

Dans ce cas $f(A) = \neg A$

Le cas n = 1 étant désormais éliminé, on suppose $n \geq 2$.

Première étape: On construit une fonction f sur $\{V, F, I\}^n$ associée à f vérifiant les propriétés suivantes:

1 f ne s'exprime qu'avec des $\wedge$, $\neg$, $\vee$.
(On a alors $f(I, ..., I) = I$).

2 La restriction $f/\{V, F\}^n$ de f à $\{V, F\}^n$ est égale à la restriction $f/\{V, F\}^n$ de f à $\{V, F\}^n$.

3 f est monotone sur $\{V, F, I\}^n$ (résulte du 1) et f est maximale sur $\{V, F, I\}^n$:

$$\forall (a_1, ..., (a_i), ..., a_n) \in \{V, F, I\}^{n-1}-(I)^{n-1},$$
$$f(a_1, ..., V, ..., a_n) = f(a_1, ..., F, ..., a_n) \Rightarrow$$
$$f(a_1, ..., I, ..., a_n) = f(a_1, ..., V, ..., a_n) = f(a_1, ..., F, ..., a_n).$$

Remarque: La maximalité de f permettra d'obtenir f à partir de f.

3.5.2.1 Construction de f.

Définition 3.5.2.1 Un monôme a_j est inclus dans f, (notation: $a_j \subset f$) si:

$$\forall (c_1, ..., (c_j), ..., c_n) \in \{V, F\}^{n-1}, f(c_1, ..., V, ..., c_n) = V.$$

Un monôme $\neg a_j$ est inclus dans f, (notation: $\neg a_j \subset f$) si:

$$\forall (c_1, ..., (c_j), ..., c_n) \in \{V, F\}^{n-1}, f(c_1, ..., F, ..., c_n) = V.$$

On définit ensuite par induction l'inclusion d'un produit de monômes dans f où le produit désigne la conjonction des monômes.

Un produit de monômes $a_{p_1}...a_{p_r}$ est inclus dans f (notation: $a_{p_1}...a_{p_r} \subset f$) si:
aucun des sous produits de monômes n'est inclus dans f et
$\forall (c_1, ..., (c_{p_1}), ..., (c_{p_r}), ..., c_n) \in \{V, F\}^{n-r}, f(c_1, ..., V, ..., V, ..., c_n) = V$.

On peut remplacer a_{p_i} par $\neg a_{p_i}$ à condition de remplacer V par F à l'endroit correspondant (la p_i-ème place) dans $f(c_1, ..., V, ..., V, ..., c_n)$.

La notation α_j désignera soit a_j soit $\neg a_j$.

Définition 3.5.2.2 On définit alors la fonction f par:

Si $\forall (a_1, ..., a_n) \in \{V, F\}^n, f(a_1, ..., a_n) = F$, alors:

$$f(a_1, ..., a_n) = (a_1 \wedge \neg a_1) \wedge ... \wedge (a_n \wedge \neg a_n).$$

Si f n'est pas constamment égale à F sur $\{V, F\}^n$,

$$\begin{aligned} f(a_1, ..., a_n) = & \bigvee_{\{j /\ \beta_j \subset f\}} (\alpha_j) \\ & \vee \bigvee_{\{(p,q)/\ \beta_p\beta_q \subset f\}} (\alpha_p\alpha_q) \\ & \vee ... \\ & ... \vee \bigvee_{\{\beta_1...\beta_n \subset f\}} (\alpha_1...\alpha_n). \end{aligned}$$

Le produit $\alpha_{p_1}...\alpha_{p_r}$ désigne la conjonction $\alpha_{p_1} \wedge ... \wedge \alpha_{p_r}$ et $\alpha_{p_i} = a_{p_i}$ si $\beta_{p_i} = b_{p_i}$ et $\neg a_{p_i}$ si $\beta_{p_i} = \neg b_{p_i}$.

Remarque: Sans faire appel à la définition 3.5.2.1, on peut définir f de la façon suivante:

$f(a_1, ..., a_n) =$

$\vee_{B_1 = \{b_j \in \{V,F\} / \forall (c_1,...,(c_j),...,c_n) \in \{V,F\}^{n-1}, f(c_1,...,b_j,...,c_n)=V\}} (\alpha_j)$

$\vee\vee_{B_2 = \{(b_p,b_q) \in \{V,F\}^2 / b_k \notin B_1 \text{ et } \forall (c_1,...,(c_p),...,(c_q),...,c_n) \in \{V,F\}^{n-2}, f(c_1,...,b_p,...,b_q,...,c_n)=V\}} (\alpha_p \alpha_q)$

$\vee ...$

$\vee\vee_{B_n = \{(b_1,...,b_n) \in \{V,F\}^n / \forall j \in \{1,...,n-1\} (b_{i_1},...,b_{i_j}) \notin B_j \text{ et } f(b_1,...,b_n)=V\}} (\alpha_1 ... \alpha_n).$

Le produit $\alpha_{p_1}...\alpha_{p_r}$ désigne la conjonction $\alpha_{p_1} \wedge ... \wedge \alpha_{p_r}$ et $\alpha_{p_i} = a_{p_i}$ si $b_{p_i} = V$ et $\neg a_{p_i}$ si $b_{p_i} = F$.

3.5.2.2 Exemple.

Soit f la fonction à trois variables définie sur par:

$$
\begin{aligned}
f(V,V,V) &= V \\
f(V,F,V) &= F \\
f(V,V,F) &= V \\
f(V,F,F) &= V \\
f(F,V,V) &= V \\
f(F,F,V) &= F \\
f(F,V,F) &= V \\
f(F,F,F) &= F
\end{aligned}
$$

On peut alors calculer la valeur de la fonction g maximale et monotone sur ayant même centre que f:

$$
\begin{aligned}
&f(V,V,V) \neq f(V,F,V) \Rightarrow g(V,I,V) = I. \\
&f(V,V,V) = f(V,V,F) = V \Rightarrow g(V,V,I) = V. \\
&f(V,V,V) = f(F,V,V) = V \Rightarrow g(I,V,V) = V. \\
&f(V,F,V) \neq f(V,F,F) \Rightarrow g(V,F,I) = I.
\end{aligned}
$$

$f(V,F,V) = f(F,F,V) = F \Rightarrow g(I,F,V) = F.$
$f(V,V,F) = f(V,F,F) = V \Rightarrow g(V,I,F) = V.$
$f(V,V,F) = f(F,V,F) = V \Rightarrow g(I,V,F) = V.$
$f(V,F,F) \neq f(F,F,F) \Rightarrow g(I,F,F) = I.$
$f(F,V,V) \neq f(F,F,V) \Rightarrow g(F,I,V) = I.$
$f(F,V,V) = f(F,V,F) = V \Rightarrow g(F,V,I) = V.$
$f(F,F,V) = f(F,F,F) = F \Rightarrow g(F,F,I) = F.$
$f(F,V,F) \neq f(F,F,F) \Rightarrow g(F,I,F) = I.$

$g(V,I,V) = I \Rightarrow g(I,I,V) = g(V,I,I) = g(I,I,I) = I.$(g est monotone).
$g(I,V,V) = g(I,V,F) = V \Rightarrow g(I,V,I) = V.$
$g(I,V,V) \neq g(I,F,V) \Rightarrow g(I,I,V) = I.$
$g(V,F,I) = I \Rightarrow g(I,F,I) = g(V,I,I) = g(I,I,I) = I.$
$g(I,F,F) = I \Rightarrow g(I,I,F) = g(I,F,I) = g(I,I,I) = I.$
$g(F,I,V) = I \Rightarrow g(I,I,V) = g(F,I,I) = g(I,I,I) = I.$
$g(F,V,I) \neq g(F,F,I) \Rightarrow g(F,I,I) = I.$

On a ainsi obtenu les 27 valeurs de vérité de g. On vérifiera par la suite que *f* et g coïncident.

Remarque: Il existe plusieurs fonctions monotones sur $\{V, F, I\}^n$ ayant même centre que f et vérifiant $f(I, ..., I) = I$.

On construit maintenant sur cet exemple *f* à partir de f.

$f(V,V,V) = f(V,V,F) = f(F,V,V) = f(F,V,F)$ donc $a_2 \subset f$.

D'autre part $f(V,V,F) = f(V,F,F) = V$ donc $a_1 \neg a_3 \subset f$.

Par conséquent $f(a_1, a_2, a_3) = a_2 \vee (a_1 \wedge \neg a_3)$.

On vérifie sur cet exemple que:

1) *f* a même centre que f.
2) $f(V,I,V) = I$ $f(V,V,I) = V$ $f(I,V,V) = V$ $f(V,F,I) = I$
$f(I,F,V) = F$ $f(V,I,F) = V$ $f(I,V,F) = V$ $f(I,F,F) = I$
$f(F,I,V) = I$ $f(F,V,I) = V$ $f(F,F,I) = F$ $f(F,I,F) = I$
$f(I,I,V) = I$ $f(V,I,I) = I$ $f(I,V,I) = V$ $f(I,F,I) = I$
$f(I,I,F) = I$ $f(F,I,I) = I$ $f(I,I,I) = I$

Donc f coïncide avec la fonction maximale g.

3.5.2.3 Montrons que la fonction f ainsi construite vérifie les propriétés voulues.

1) f vérifie la propriété 1 puisqu'elle ne s'exprime qu'avec des $\neg$, $\wedge$ et des $\vee$.
On a donc bien aussi $f(I, ..., I) = I$.

2) f a même centre que f.

- Soit $\forall(b_1, ..., b_n) \in \{V, F\}^n$, $f(b_1, ..., b_n) = F$, et par construction de f, $\forall(b_1, ..., b_n) \in \{V, F\}^n$, $f(b_1, ..., b_n) = F$.

- Si f n'est pas constamment égale à F, on vérifie que f a même centre que f.

- Soit $(a_1, ..., a_n) \in \{V, F\}^n$, $f(a_1, ..., a_n) = V$, deux cas se présentent alors:

a) Il existe $\{a_{i_1}, ..., a_{i_p}\} \subset \{a_1, ..., a_n\}$ tel que:
$\forall(c_1, ..., (a_{i_1}), ..., (a_{i_p}), ..., c_n) \in \{V, F\}^{n-p}$, $f(c_1, ..., a_{i_1}, ..., a_{i_p}, ..., c_n) = V$.
Soit r l'entier minimum vérifiant cette propriété.

Autrement dit:
$(\beta_{i_1}...\beta_{i_r}) \subset f(b_1, ..., b_n)$ avec $\beta_{i_j} = b_{i_j}$ si $a_{i_j} = V$, $= \neg b_{i_j}$ si $a_{i_j} = F$.

Donc $f(a_1, ..., a_n) = V$.

b) Si le cas a) ne se produit pas, dans $f(b_1, ..., b_n)$ intervient le produit $\beta_1...\beta_n$ avec $\beta_i = b_i$ si $a_i = V$, $= \neg b_i$ si $a_i = F$.

On a bien $f(a_1, ..., a_n) = V$.

- Soit $(a_1, ..., a_n) \in \{V, F\}^n$, $f(a_1, ..., a_n) = F$, dans ce cas il n'existe pas de $\{a_{i_1}, ..., a_{i_p}\} \subset \{a_1, ..., a_n\}$ tel que:
$\forall(c_1, ..., (a_{i_1}), ..., (a_{i_p}), ..., c_n) \in \{V, F\}^{n-p}$, $f(c_1, ..., a_{i_1}, ..., a_{i_p}, ..., c_n) = V$.
En effet cela entraînerait que $f(a_1, ..., a_n) = V$.

De plus, $\forall(b_1, ..., b_n) \in \{V, F\}^n$, $f(b_1, ..., b_n) = V$, on a:
$(b_1, ..., b_n) \neq (a_1, ..., a_n)$.

Donc, par construction de f, $f(a_1, ..., a_n) = F$.

3) f est maximale.

$\forall(a_1, ..., (a_i), ..., a_n) \in \{V, F, I\}^{n-1}-(I)^{n-1},$
$(f(a_1, ..., V, ..., a_n) = f(a_1, ..., F, ..., a_n) \Rightarrow$
$f(a_1, ..., I, ..., a_n) = f(a_1, ..., V, ..., a_n) = f(a_1, ..., F, ..., a_n)).$

Dans le cas où $f(a_1, ..., a_n) = (a_1 \wedge \neg a_1) \wedge ... \wedge (a_n \wedge \neg a_n)$,
$\forall(a_1, ..., (a_i), ..., a_n) \in \{V, F, I\}^{n-1}-(I)^{n-1},$
$f(a_1, ..., I, ..., a_n) = F$, donc on peut éliminer ce cas.

On démontre que:
$\forall((a_1), ..., a_n) \in \{V, F, I\}^{n-1}-(I)^{n-1},$
$f(V, ..., a_n) = f(F, ..., a_n) = \alpha \Rightarrow$
$f(I, ..., a_n) = \alpha$

(On suppose pour plus de simplicité que i = 1.)

On rappelle l'expression de f:

$$f(a_1, ..., a_n) = \vee_{\{j/\ \beta_j \subset f\}} (\alpha_j)$$
$$\vee \vee_{\{(p,q)/\ \beta_p\beta_q \subset f\}} (\alpha_p\alpha_q)$$
$$\vee ...$$
$$... \vee \vee_{\{\beta_1 ... \beta_n \subset f\}} (\alpha_1 ... \alpha_n).$$

<u>Premier cas:</u> <u>$\alpha = F$.</u>

A) Ni a_1 ni $\neg a_1$ n'apparaissent dans f.

Alors $f(a_1, ..., a_n) = \exp(a_2, ..., a_n)$ auquel cas on a $f(V, ..., a_n) = f(F, ..., a_n) = f(I, ..., a_n) = F$.

B) Le monôme a_1 apparaît dans la décomposition de f.

Vu l'expression de f on a:
$f(a_1, ..., a_n) \equiv_T a_1 \vee (\neg a_1 \exp_1(a_2, ..., a_n)) \vee (\exp_2(a_2, ..., a_n))$
(en effet le mônome a_1 n'apparaît plus dans un produit de mônomes)

Cette équivalence trivaluée est obtenue en factorisant $\neg a_1$ dans l'expression de f.

Alors $f(V, ..., a_n) = V$ donc c'est impossible.

C) De même si le monôme $\neg a_1$ apparaît dans la décomposition de f, $f(F, ..., a_n) = V$ donc c'est impossible.

D) Il reste donc le cas où:

$f(a_1, ..., a_n) \equiv_T (a_1 exp_1(a_2, ..., a_n)) \vee (\neg a_1 exp_2(a_2, ..., a_n)) \vee (exp_3(a_2, ..., a_n))$

Cette équivalence trivaluée est obtenue en factorisant dans l'expression de f a_1 et $\neg a_1$.

$f(V, ..., a_n) = F \Rightarrow exp_1(a_2, ..., a_n) = F$ et $exp_3(a_2, ..., a_n) = F$.

$f(F, ..., a_n) = F \Rightarrow exp_2(a_2, ..., a_n) = F$ et $exp_3(a_2, ..., a_n) = F$.

Donc $f(I, ..., a_n) = F$ et c'est terminé.

Deuxième cas: $\alpha = V$.

A) Ni a_1 ni $\neg a_1$ n'apparaissent dans f.

Alors $f(a_1, ..., a_n) = exp(a_2, ..., a_n)$ auquel cas on a $f(V, ..., a_n) = f(F, ..., a_n) = f(I, ..., a_n) = V$.

B) Le monôme a_1 apparaît dans la décomposition de f et le monôme $\neg a_1$ n'apparaît pas dans la décomposition de f.

Vu l'expression de f on a:

$$f(a_1, ..., a_n) \equiv_T a_1 \vee (exp_2(a_2, ..., a_n)).$$

Alors $f(F, ..., a_n) = V \Rightarrow exp_2(a_2, ..., a_n) = V$ et $f(I, ..., a_n) = V$.

C) De même si le monôme $\neg a_1$ apparaît dans la décomposition de f et le monôme $\neg a_1$ n'apparaît pas dans la décomposition de f.

Vu l'expression de f on a:

$$f(a_1, ..., a_n) \equiv_T \neg a_1 \vee (exp_2(a_2, ..., a_n)).$$

Alors $f(V, ..., a_n) = V \Rightarrow exp_2(a_2, ..., a_n) = V$ et $f(I, ..., a_n) = V$.

D) Le monôme a_1 ainsi que le monôme $\neg a_1$ apparaissent dans la décomposition de f.

$$f(a_1, ..., a_n) \equiv_T a_1 \vee \neg a_1 \vee (exp_2(a_2, ..., a_n))$$

On a alors: Pour $i \neq 1$,

$\forall (a_1, ..., (a_i), ..., a_n) \in \{V, F\}^{n-1}$,
$$f(a_1, ..., V, ..., a_n) = f(a_1, ..., F, ..., a_n) = V.$$

Comme f et f ont même centre,

$\forall (a_1, ..., (a_i), ..., a_n) \in \{V, F\}^{n-1}$,
$$\mathrm{f}(a_1, ..., V, ..., a_n) = \mathrm{f}(a_1, ..., F, ..., a_n) = V.$$

La construction de f implique que $a_i \vee \neg a_i$ apparaît dans l'expression de f, pour $i \neq 1$.

On a donc avec l'hypothèse initiale:

$$f(a_1, ..., a_n) \equiv_T \vee_{i \in \{1, ..., n\}} (a_i \vee \neg a_i).$$

On a bien $f(I, ..., a_n) = V$ si les a_i ne sont pas tous égaux à I.

E) Le monôme a_1 apparaît dans la décomposition de f et $\neg a_1$ figure dans un produit de monômes de la décomposition de f.

$$f(a_1, ..., a_n) \equiv_T a_1 \vee \alpha_{i_2} \vee ... \vee \alpha_{i_n} \vee (\vee \neg a_1 \alpha_q) \vee (\vee \alpha_p \alpha_q) \vee (\vee \neg a_1 \alpha_p \alpha_q) \vee ...$$

On a:

$$a_1 \equiv_T a_1 \vee a_1 \alpha_{i_1} ... \alpha_{i_r}$$

D'autre part:

$\neg a_1 \alpha_{i_1} ... \alpha_{i_r} \vee a_1 \alpha_{i_1} ... \alpha_{i_r}$ a la même valeur de vérité que $\alpha_{i_1} ... \alpha_{i_r}$ lorsque a_1 n'a pas la valeur Indéterminé.

A chaque terme de f contenant $\neg a_1$ c'est-à-dire $\neg a_1\alpha_{i_1}...\alpha_{i_r}$, on rajoute le terme $a_1\alpha_{i_1}...\alpha_{i_r}$, alors $f(a_1, ..., a_n)$ a la même valeur de vérité que:

$$g(a_1, ..., a_n) = a_1 \vee \alpha_{i_2} \vee ... \vee \alpha_{i_n} \vee (\vee(\alpha_{i_1}...\alpha_{i_r})) \vee (\vee \alpha_p \alpha_q) \vee ...$$

si a_1 ne prend pas la valeur de vérité Indéterminé.

On a donc:

$\forall (c_1, ..., (b_{i_1}), ..., (b_{i_r}), ..., c_n) \in \{V, F\}^{n-r}, f(c_1, ..., b_{i_1}, ..., b_{i_r}, ..., c_n) = V.$

avec $b_{i_j} = V$ si $\alpha_{i_j} = a_{i_j}$ dans l'expression de $g(a_1, ..., a_n)$ et $b_{i_j} = F$ si $\alpha_{i_j} = \neg a_{i_j}$ dans l'expression de $g(a_1, ..., a_n)$.

Comme f et f ont même centre,

$\forall (c_1, ..., (b_{i_1}), ..., (b_{i_r}), ..., c_n) \in \{V, F\}^{n-r}, \mathrm{f}(c_1, ..., b_{i_1}, ..., b_{i_r}, ..., c_n) = V.$

Ce qui signifie par construction de f que $\alpha_{i_1}...\alpha_{i_r}$ ou un sous produit de $\alpha_{i_1}...\alpha_{i_r}$ intervient dans l'expression de f, ce qui est impossible puisqu'au départ on a supposé que $\neg a_1\alpha_{i_1}...\alpha_{i_r}$ intervenait dans l'expression de f. Le cas E ne peut donc pas se produire.

F) Le monôme $\neg a_1$ apparaît dans la décomposition de f et a_1 figure dans un produit de monômes de la décomposition de f.

Ce cas se traite exactement comme le cas E.

G) Le dernier cas à traiter est celui où a_1 et $\neg a_1$ figurent dans un produit de monômes de la décomposition de f.

De la même façon que précédemment, on isole les expressions contenant a_1, celles contenant $\neg a_1$, et celles ne contenant ni a_1 ni $\neg a_1$.

On a $f(V, ..., a_n) = f(F, ..., a_n) = V$.

- Soit il existe dans la disjonction exprimant f un produit de monômes ne contenant ni a_1 ni $\neg a_1$ vrai en $(a_2, ..., a_n)$;
la valeur de vérité de ce produit ne dépendant pas de a_1, $f(I, ..., a_n) = V$.

- Soit toutes les expressions ne contenant ni ni a_1 ni $\neg a_1$ sont fausses ou indéterminées;

Dans ce cas, $f(V, ..., a_n) = V$ implique qu'il existe une expression contenant a_1 vraie et $f(F, ..., a_n) = V$ implique qu'il existe une expression contenant $\neg a_1$ vraie.

Soit $a_1\alpha_{p_1}...\alpha_{p_r}$ une expression contenant a_1 ayant la valeur Vrai et $\neg a_1\alpha_{q_1}...\alpha_{q_s}$ une expression contenant $\neg a_1$ ayant la valeur Vrai.

1) Si $\{\alpha_{p_1}, ..., \alpha_{p_r}\} \neq \{\alpha_{q_1}, ..., \alpha_{q_s}\}$, on multiplie $\alpha_{p_1}...\alpha_{p_r}$ par tous les α_{q_i} n'appartenant pas à $\{\alpha_{p_1}, ..., \alpha_{p_r}\}$ et $\alpha_{q_1}...\alpha_{q_s}$ par tous les α_{p_j} n'appartenant pas à $\{\alpha_{q_1}, ..., \alpha_{q_s}\}$.

Soit P le produit commun obtenu.

Comme $a_1\alpha_{p_1}...\alpha_{p_r} \equiv_T a_1\alpha_{p_1}...\alpha_{p_r} \vee a_1P$ et $\neg a_1\alpha_{q_1}...\alpha_{q_s} \equiv_T \neg a_1\alpha_{q_1}...\alpha_{q_s} \vee \neg a_1P$,

on a:

$$f(a_1, ..., a_n) \equiv_T f(a_1, ..., a_n) \vee a_1P \vee \neg a_1P.$$

D'autre part, si a_1 n'a pas la valeur Indéterminé,

$a_1P \vee \neg a_1P$ a la même valeur de vérité que P.

Donc si a_1 n'a pas la valeur Indéterminé, $f(a_1, ..., a_n)$ a la même valeur de vérité que $f(a_1, ..., a_n) \vee P$.

Si $P = \alpha_{i_1}...\alpha_{i_k}$,

$\forall(c_1, ..., (b_{i_1}), ..., (b_{i_k}), ..., c_n) \in \{V, F\}^{n-k}$, $f(c_1, ..., b_{i_1}, ..., b_{i_k}, ..., c_n) = V$.

avec $b_{i_j} = V$ si $\alpha_{i_j} = a_{i_j}$ dans l'expression de P et $b_{i_j} = F$ si $\alpha_{i_j} = \neg a_{i_j}$ dans l'expression de P.

Comme f et ont f même centre,

$\forall(c_1, ..., (b_{i_1}), ..., (b_{i_k}), ..., c_n) \in \{V, F\}^{n-k}$, $\mathrm{f}(c_1, ..., b_{i_1}, ..., b_{i_k}, ..., c_n) = V$.

Ce qui signifie par construction de f que $\alpha_{i_1}...\alpha_{i_k}$ ou un sous produit de $\alpha_{i_1}...\alpha_{i_k}$ intervient dans l'expression de f.

- Si ce sous produit est un sous produit de $\alpha_{p_1}...\alpha_{p_r}$ ou de $\alpha_{q_1}...\alpha_{q_s}$, c'est impossible par construction de $\mathcal{f}$.

- Sinon, ce sous produit ne contenant ni a_1 ni $\neg a_1$ conserve la valeur Vrai si a_1 prend la valeur Indéterminé.

2) Si $\{\alpha_{p_1}, ..., \alpha_{p_r}\} = \{\alpha_{q_1}, ..., \alpha_{q_s}\}$, si a_1 n'a pas la valeur Indéterminé, $\mathcal{f}(a_1, ..., a_n)$ a la même valeur de vérité que $\mathcal{f}(a_1, ..., a_n) \vee \alpha_{p_1}...\alpha_{p_r}$.

Par construction de $\mathcal{f}$, $\alpha_{p_1}...\alpha_{p_r}$ ou un sous produit de $\alpha_{p_1}...\alpha_{p_r}$ intervient dans l'expression de $\mathcal{f}$. Ceci est impossible puisqu'au départ $a_1\alpha_{p_1}...\alpha_{p_r}$ apparaît dans la décomposition de $\mathcal{f}$.

Deuxième étape: On construit f à partir de $\mathcal{f}$.

3.5.2.4 Construction de f à partir de $\mathcal{f}$.

4 cas se présentent:

- Cas 1: f = $\mathcal{f}$ et c'est terminé.

- Cas 2:

$\exists(a_1, ..., a_n) \in \{V, F, I\}^n$,
$f(a_1, ..., a_n) = I$ et $\mathcal{f}(a_1, ..., a_n) = V$.

$\forall(b_1, ..., b_n) \in \{V, F, I\}^n$,
$\mathcal{f}(b_1, ..., b_n) = F$ implique $f(b_1, ..., b_n) = F$.

- Cas 3:

$\exists(a_1, ..., a_n) \in \{V, F, I\}^n$,
$f(a_1, ..., a_n) = I$ et $\mathcal{f}(a_1, ..., a_n) = F$.

$\forall(b_1, ..., b_n) \in \{V, F, I\}^n$,
$\mathcal{f}(b_1, ..., b_n) = V$ implique $f(b_1, ..., b_n) = V$.

- Cas 4:

$\exists(a_1, ..., a_n) \in \{V, F, I\}^n$,
$f(a_1, ..., a_n) = I$ et $\mathcal{f}(a_1, ..., a_n) = V$.

$\exists(b_1, ..., b_n) \in \{V, F, I\}^n$,
$f(b_1, ..., b_n) = I$ et $f(b_1, ..., b_n) = F$.

- Ce sont les seuls cas à envisager car:

Si $\exists(c_1, ..., c_n) \in \{V, F, I\}^n$, $f(c_1, ..., c_n) \neq \mathcal{f}(c_1, ..., c_n)$, forcément:

1) $\exists i \in \{1, ..., n\}$, $c_i \neq I$ car $f(I, ..., I) = \mathcal{f}(I, ..., I) = I$.

2) $\exists i \in \{1, ..., n\}$, $c_i = I$ car f et $\mathcal{f}$ ont même centre (coïncident sur $\{V, F\}^n$).

3) Enfin, $f(c_1, ..., c_n) \notin \{V, F\}$:

En effet, soit $\{i_1, ..., i_r\}$ l'ensemble des indices i_j tels que $c_{i_j} = I$ (cet ensemble est non vide d'après 1)).

$f(c_1, ..., c_{i_1}, ..., c_{i_r}, ..., c_n) = \alpha \in \{V, F\}$ implique
$\forall(b_{i_1}, ..., b_{i_r}) \in \{V, F\}^r$, $f(c_1, ..., b_{i_1}, ..., b_{i_r}, ..., c_n) = \alpha$

par monotonie de f.

Comme f et $\mathcal{f}$ ont même centre, $\forall(b_{i_1}, ..., b_{i_r}) \in \{V, F\}^r$,
$\mathcal{f}(c_1, ..., b_{i_1}, ..., b_{i_r}, ..., c_n) = \alpha$

puisque $\forall(b_{i_1}, ..., b_{i_r}) \in \{V, F\}^r$, $(c_1, ..., b_{i_1}, ..., b_{i_r}, ..., c_n) \in \{V, F\}^n$.

Comme $\mathcal{f}$ est maximale, d'après 3.5.2.3, $\mathcal{f}(c_1, ..., c_{i_1}, ..., c_{i_r}, ..., c_n) = \alpha$.

Donc $f(c_1, ..., c_n) = \mathcal{f}(c_1, ..., c_n) = \alpha$.

Pour obtenir f à partir de $\mathcal{f}$ le cas 1) est trivial.

a) Pour les n-uplets $(a_1, ..., a_n) \in \{V, F, I\}^n$,
$f(a_1, ..., a_n) = I$ et $\mathcal{f}(a_1, ..., a_n) = V$, on construit des fonctions:

$$V(a_1, ..., a_n)$$

telles que:

$\forall(c_1, ..., c_n) \in \{V, F, I\}^n$,
$(c_1, ..., c_n) \leq (a_1, ..., a_n)$ implique $V(a_1, ..., a_n)(c_1, ..., c_n) = I$.

non $((c_1, ..., c_n) \leq (a_1, ..., a_n))$ implique $V(a_1, ..., a_n)(c_1, ..., c_n) = V$.

b) Pour les n-uplets $(b_1, ..., b_n) \in \{V, F, I\}^n$,

$$f(b_1, ..., b_n) = I \text{ et } f(b_1, ..., b_n) = F,$$

on construit des fonctions:

$$F(b_1, ..., b_n)$$

telles que:

$\forall (c_1, ..., c_n) \in \{V, F, I\}^n$,
$(c_1, ..., c_n) \leq (b_1, ..., b_n)$ implique $F(b_1, ..., b_n)(c_1, ..., c_n) = I$.
non $((c_1, ..., c_n) \leq (b_1, ..., b_n))$ implique $F(b_1, ..., b_n)(c_1, ..., c_n) = F$.

a) $V(a_1, ..., a_n)(c_1, ..., c_n) =$

$$\bigvee(i/a_i = I)(c_i \vee \neg c_i) \vee \bigvee(i/a_i = F)(c_i) \vee \bigvee(i/a_i = V)(\neg c_i).$$

L'ensemble des $i \in \{1, ..., n\}$, $a_i = I$ est non vide d'après 1).

Montrons que:
$\forall (c_1, ..., c_n) \in \{V, F, I\}^n$,
$(c_1, ..., c_n) \leq (a_1, ..., a_n)$ implique $V(a_1, ..., a_n)(c_1, ..., c_n) = I$.
non $((c_1, ..., c_n) \leq (a_1, ..., a_n))$ implique $V(a_1, ..., a_n)(c_1, ..., c_n) = V$.

- $(c_1, ..., c_n) \leq (a_1, ..., a_n)$ implique:

$$a_j = I \Rightarrow c_j = I$$
$$a_j = F \Rightarrow c_j = I \text{ ou } F$$
$$a_j = V \Rightarrow c_j = I \text{ ou } V$$

Donc $V(a_1, ..., a_n)(c_1, ..., c_n) = I \vee (F \text{ ou } I) \vee (F \text{ ou } I) = I$.

- non $((c_1, ..., c_n) \leq (a_1, ..., a_n))$ implique: $\exists j \in \{1, ..., n\}$, non $(c_j \leq a_j)$

$$\exists j \in \{1, ..., n\}, c_j = V \text{ et } a_j = F \text{ ou}$$
$$\exists j \in \{1, ..., n\}, c_j = V \text{ et } a_j = I \text{ ou}$$
$$\exists j \in \{1, ..., n\}, c_j = F \text{ et } a_j = V \text{ ou}$$
$$\exists j \in \{1, ..., n\}, c_j = F \text{ et } a_j = I.$$

Dans ce cas, $V(a_1, ..., a_n)(c_1, ..., c_n) = V$.

On a alors si $(c_1, ..., c_n) \leq (a_1, ..., a_n)$,

$$(V(a_1, ..., a_n) \wedge f)(c_1, ..., c_n) = I.$$

En effet, par monotonie de f, $f(c_1, ..., c_n) \leq f(a_1, ..., a_n)$; donc comme $f(a_1, ..., a_n) = V$, $f(c_1, ..., c_n) = I$ ou V et $V(a_1, ..., a_n)(c_1, ..., c_n) = I$.

Si non$((c_1, ..., c_n) \leq (a_1, ..., a_n))$,

$$(V(a_1, ..., a_n) \wedge f)(c_1, ..., c_n) = f(c_1, ..., c_n).$$

Car $V(a_1, ..., a_n)(c_1, ..., c_n) = V$.

b) $F(b_1, ..., b_n)(c_1, ..., c_n) =$

$$\bigwedge(i/b_i = I)(c_i \wedge \neg c_i) \wedge \bigwedge(i/b_i = F)(\neg c_i) \wedge \bigwedge(i/b_i = V)(c_i).$$

L'ensemble des $i \in \{1, ..., n\}$, $b_i = I$ est non vide d'après 1).

Montrons que:
$\forall (c_1, ..., c_n) \in \{V, F, I\}^n$,
$(c_1, ..., c_n) \leq (b_1, ..., b_n)$ implique $F(b_1, ..., b_n)(c_1, ..., c_n) = I$.
non $((c_1, ..., c_n) \leq (b_1, ..., b_n))$ implique $F(b_1, ..., b_n)(c_1, ..., c_n) = F$.

- $(c_1, ..., c_n) \leq (b_1, ..., b_n)$ implique:

$$b_j = I \Rightarrow c_j = I$$
$$b_j = F \Rightarrow c_j = I \text{ ou } F$$
$$b_j = V \Rightarrow c_j = I \text{ ou } V$$

Donc $F(b_1, ..., b_n)(c_1, ..., c_n) = I \wedge (V \text{ ou } I) \wedge (V \text{ ou } I) = I$.

- non $((c_1, ..., c_n) \leq (b_1, ..., b_n))$ implique: $\exists j \in \{1, ..., n\}$, non $(c_j \leq b_j)$

$$\exists j \in \{1, ..., n\}, c_j = V \text{ et } b_j = F \text{ ou}$$
$$\exists j \in \{1, ..., n\}, c_j = V \text{ et } b_j = I \text{ ou}$$
$$\exists j \in \{1, ..., n\}, c_j = F \text{ et } b_j = V \text{ ou}$$
$$\exists j \in \{1, ..., n\}, c_j = F \text{ et } b_j = I.$$

Dans ce cas, $F(b_1, ..., b_n)(c_1, ..., c_n) = F$.

On a alors si $(c_1, ..., c_n) \leq (b_1, ..., b_n)$,

$$(F(b_1, ..., b_n) \vee f)(c_1, ..., c_n) = I.$$

En effet, par monotonie de f, $f(c_1, ..., c_n) \leq f(b_1, ..., b_n)$; donc comme $f(b_1, ..., b_n) = F$, $f(c_1, ..., c_n) = I$ ou F et $F(b_1, ..., b_n)(c_1, ..., c_n) = I$.

Si non$((c_1, ..., c_n) \leq (a_1, ..., a_n))$,

$$(F(b_1, ..., b_n) \vee f)(c_1, ..., c_n) = f(c_1, ..., c_n).$$

Car $F(b_1, ..., b_n)(c_1, ..., c_n) = F$.

<u>On peut maintenant effectuer la construction de f à partir de *f*.</u>

4 cas se présentent:

- <u>Cas 1</u>: f = *f* et c'est terminé.

- <u>Cas 2</u>:

$\exists (a_1, ..., a_n) \in \{V, F, I\}^n$,
$$f(a_1, ..., a_n) = I \text{ et } f(a_1, ..., a_n) = V.$$

$\forall (b_1, ..., b_n) \in \{V, F, I\}^n$,
$$f(b_1, ..., b_n) = F \text{ implique } f(b_1, ..., b_n) = F.$$

Soit $g(c_1, ..., c_n) =$

$$\wedge_{\{(a_1, ..., a_n)/\ f(a_1, ..., a_n) = I \text{ et } f(a_1, ..., a_n) = V\}} (V(a_1, ..., a_n) \wedge f)(c_1, ..., c_n)$$

Soit $G = \{(a_1, ..., a_n)/\ f(a_1, ..., a_n) = I \text{ et } f(a_1, ..., a_n) = V\}$.

Montrons que:

$\forall (c_1, ..., c_n) \in \{V, F, I\}^n$,

$$f(c_1, ..., c_n) = g(c_1, ..., c_n).$$

Soit $(c_1, ..., c_n) \in \{V, F, I\}^n$.

<u>1) Il existe $(a_1^0, ..., a_n^0) \in G/\ (c_1, ..., c_n) \leq (a_1^0, ..., a_n^0)$.</u>

$(V(a_1^0, ..., a_n^0) \wedge f)(c_1, ..., c_n) = I$, d'après ci-dessus.

Pour les autres $(a_1, ..., a_n) \in G$,

$(V(a_1, ..., a_n) \wedge f)(c_1, ..., c_n) = f(c_1, ..., c_n)$ ou I, d'après ci-dessus.

D'autre part,
$(c_1, ..., c_n) \leq (a_1^0, ..., a_n^0)$ et $f(a_1^0, ..., a_n^0) = V$.
Puisque f est monotone, $f(c_1, ..., c_n) = V$ ou I, donc $g(c_1, ..., c_n) = I$.

D'autre part, $\mathrm{f}(a_1^0, ..., a_n^0).= I$ et f étant monotone,
$(c_1, ..., c_n) \leq (a_1^0, ..., a_n^0)$ implique $\mathrm{f}(c_1, ..., c_n) = I = g(c_1, ..., c_n)$.

2) $\forall(a_1, ..., a_n) \in G$, $\mathrm{non}((c_1, ..., c_n) \leq (a_1, ..., a_n))$, dans ce cas:

$(V(a_1, ..., a_n) \wedge f)(c_1, ..., c_n) = f(c_1, ..., c_n)$ d'après ci-dessus, et donc:

$$g(c_1, ..., c_n) = f(c_1, ..., c_n).$$

Mais $f(c_1, ..., c_n) = \mathrm{f}(c_1, ..., c_n)$ puisque les seuls n-uplets où f et f diffèrent sont les éléments de G.

Donc $\mathrm{f}(c_1, ..., c_n) = g(c_1, ..., c_n)$.

- Cas 3:

$\exists(a_1, ..., a_n) \in \{V, F, I\}^n$,

$$\mathrm{f}(a_1, ..., a_n) = I \text{ et } f(a_1, ..., a_n) = F.$$

$\forall(b_1, ..., b_n) \in \{V, F, I\}^n$,

$$f(b_1, ..., b_n) = V \text{ implique } \mathrm{f}(b_1, ..., b_n) = V.$$

Soit $h(c_1, ..., c_n) =$

$$\bigvee_{\{(b_1, ..., b_n)/\mathrm{f}(b_1, ..., b_n) = I \text{ et } f(b_1, ..., b_n) = F\}} (F(b_1, ..., b_n) \vee f)(c_1, ..., c_n)$$

Soit $H = \{(b_1, ..., b_n)/\mathrm{f}(b_1, ..., b_n) = I \text{ et } f(b_1, ..., b_n) = F\}$.

Montrons que:

$\forall(c_1, ..., c_n) \in \{V, F, I\}^n$,

$$\mathrm{f}(c_1, ..., c_n) = h(c_1, ..., c_n).$$

Soit $(c_1, ..., c_n) \in \{V, F, I\}^n$.

1) Il existe $(b_1^0, ..., b_n^0) \in H/ (c_1, ..., c_n) \leq (b_1^0, ..., b_n^0)$.

$(F(b_1^0, ..., b_n^0) \vee f)(c_1, ..., c_n) = I$, d'après ci-dessus.

Pour les autres $(a_1, ..., a_n) \in G$,

$(F(a_1, ..., a_n)\vee f)(c_1, ..., c_n) = f(c_1, ..., c_n)$ ou I, d'après ci-dessus.

D'autre part,
$(c_1, ..., c_n) \leq (b_1^0, ..., b_n^0)$ et $f(b_1^0, ..., b_n^0) = F$.
Puisque f est monotone, $f(c_1, ..., c_n) = F$ ou I, donc $h(c_1, ..., c_n) = I$.

D'autre part, $f(b_1^0, ..., b_n^0) .= I$ et f étant monotone, $(c_1, ..., c_n) \leq (b_1^0, ..., b_n^0)$ implique $f(c_1, ..., c_n) = I = h(c_1, ..., c_n)$.

<u>2) $\forall(b_1, ..., b_n) \in H$, non$((c_1, ..., c_n) \leq (b_1, ..., b_n))$</u>, dans ce cas:

$(F(b_1, ..., b_n)\vee f)(c_1, ..., c_n) = f(c_1, ..., c_n)$ d'après ci-dessus, et donc:
$$h(c_1, ..., c_n) = f(c_1, ..., c_n).$$

Mais $f(c_1, ..., c_n) = f(c_1, ..., c_n)$ puisque les seuls n-uplets où f et f diffèrent sont les éléments de H.

Donc $f(c_1, ..., c_n) = h(c_1, ..., c_n)$.

- <u>Cas 4</u>:

$\exists(a_1, ..., a_n) \in \{V, F, I\}^n$,
$$f(a_1, ..., a_n) = I \text{ et } f(a_1, ..., a_n) = V.$$

$\exists(b_1, ..., b_n) \in \{V, F, I\}^n$,
$$f(b_1, ..., b_n) = I \text{ et } f(b_1, ..., b_n) = F.$$

Soit $k(c_1, ..., c_n) =$
$$\bigvee_{\{(b_1, ..., b_n)/\ f(b_1, ..., b_n) = I \text{ et } f(b_1, ..., b_n) = F\}} (F(b_1, ..., b_n)\vee g)(c_1, ..., c_n)$$

avec:

$g(c_1, ..., c_n) =$
$$\bigwedge_{\{(a_1, ..., a_n)/f(a_1, ..., a_n) = I \text{ et } f(a_1, ..., a_n) = V\}} (V(a_1, ..., a_n)\wedge f)(c_1, ..., c_n)$$

Soit $G = \{(a_1, ..., a_n)/\ f(a_1, ..., a_n) = I \text{ et } f(a_1, ..., a_n) = V\}$.

Soit $H = \{(b_1, ..., b_n)/\ f(b_1, ..., b_n) = I \text{ et } f(b_1, ..., b_n) = F\}$.

Montrons que:

$\forall(c_1, ..., c_n) \in \{V, F, I\}^n$,

$$f(c_1, ..., c_n) = k(c_1, ..., c_n).$$

Soit $(c_1, ..., c_n) \in \{V, F, I\}^n$.

<u>1) Il existe $(b_1^0, ..., b_n^0) \in H/ (c_1, ..., c_n) \leq (b_1^0, ..., b_n^0)$.</u>
Alors:

a) $f(c_1, ..., c_n) = I$ par monotonie de f, puisque $f(b_1^0, ..., b_n^0) = I$ et $(c_1, ..., c_n) \leq (b_1^0, ..., b_n^0)$.
b) $f(c_1, ..., c_n) = I$ ou F par monotonie de f, puisque $f(b_1^0, ..., b_n^0) = F$ et $(c_1, ..., c_n) \leq (b_1^0, ..., b_n^0)$.

- <u>$f(c_1, ..., c_n) = F$</u>

$\forall(a_1, ..., a_n) \in G$, on a non$((c_1, ..., c_n) \leq (a_1, ..., a_n))$, car sinon on aurait: $f(c_1, ..., c_n) = I$ ou V par monotonie de f. On a donc:
$g(c_1, ..., c_n) = f(c_1, ..., c_n) = F$.

Comme $F(b_1^0, ..., b_n^0)(c_1, ..., c_n) = I$ car $(c_1, ..., c_n) \leq (b_1^0, ..., b_n^0)$,
$k(c_1, ..., c_n) = I = f(c_1, ..., c_n)$.

<u>- $f(c_1, ..., c_n) = I$</u>

$g(c_1, ..., c_n) = (V(a_1, ..., a_n) \wedge f)(c_1, ..., c_n) = I$ ou $f(c_1, ..., c_n)$;
comme $f(c_1, ..., c_n)= I$, $g(c_1, ..., c_n) = I$.
Comme $F(b_1^0, ..., b_n^0)(c_1, ..., c_n) = I$ et pour les autres $(b_1, ..., b_n) \in H$, $F(b_1, ..., b_n)(c_1, ..., c_n) = I$ ou F, on a donc: $k(c_1, ..., c_n) = I = f(c_1, ..., c_n)$.

<u>2) $\forall(b_1, ..., b_n) \in H$, non$((c_1, ..., c_n) \leq (b_1, ..., b_n))$</u>, dans ce cas:

$\forall(b_1, ..., b_n) \in H$, $F(b_1, ..., b_n)(c_1, ..., c_n) = F$, et donc:
$$(F(b_1, ..., b_n) \vee g)(c_1, ..., c_n) = g(c_1, ..., c_n) = k(c_1, ..., c_n).$$

Calculons $g(c_1, ..., c_n)$.

<u>- Soit il existe $(a_1^0, ..., a_n^0) \in G/ (c_1, ..., c_n) \leq (a_1^0, ..., a_n^0)$.</u>

On a alors: a) $f(c_1, ..., c_n) = I$ par monotonie de f, puisque $f(a_1^0, ..., a_n^0) = I$ et $(c_1, ..., c_n) \leq (b_1^0, ..., b_n^0)$.

b) $f(c_1, ..., c_n)$ = I ou V par monotonie de f, puisque $f(a_1^0, ..., a_n^0)$ = V et $(c_1, ..., c_n) \leq (a_1^0, ..., a_n^0)$.

Dans ce cas:

$(V(a_1^0, ..., a_n^0) \wedge f)(c_1, ..., c_n)$ = I, et $g(c_1, ..., c_n)$ = I, puisque pour les autres $(a_1, ..., a_n) \in G$, $V(a_1, ..., a_n)(c_1, ..., c_n)$ = I ou V et donc $(V(a_1, ..., a_n) \wedge f)(c_1, ..., c_n)$ = V ou I

Donc $k(c_1, ..., c_n) = \mathrm{f}(c_1, ..., c_n)$ = I.

<u>- Soit $\forall(a_1, ..., a_n) \in G$, on a: non($(c_1, ..., c_n) \leq (a_1, ..., a_n)$)</u>, dans ce cas:

$g(c_1, ..., c_n) = f(c_1, ..., c_n) = \mathrm{f}(c_1, ..., c_n)$ puisque les seuls n-uplets où f et f diffèrent sont les éléments de H et les éléments de G. (on a aussi $\forall(b_1, ..., b_n) \in H$,: non($(c_1, ..., c_n) \leq (b_1, ..., b_n)$))

Donc $\mathrm{f}(c_1, ..., c_n) = k(c_1, ..., c_n)$.

Conclusion: Comme les fonctions $F(a_1, ..., a_n)$, $V(a_1, ..., a_n)$, f ne s'expriment qu'avec des $\neg$ et des $\vee$, une fonction f vérifiant les propriétés 1), 2), 3) du théorème ne s'exprime elle aussi qu'avec des $\neg$ et des $\vee$. ♦

3.5.2.5 Exemple final.

On reprend la construction de f à partir de f dans l'exemple 3.5.2.2.

La fonction f était la fonction à trois variables définie par:

f(V,V,V) = V
f(V,F,V) = F
f(V,V,F) = V
f(V,F,F) = V
f(F,V,V) = V
f(F,F,V) = F
f(F,V,F) = V
f(F,F,F) = F

On avait: $f(a_1, a_2, a_3) = a_2 \vee (a_1 \wedge \neg a_3)$.

$f(V,I,V) = I$	$\mathrm{f}(V,I,V) = I$
$f(V,V,I) = V$	$\mathrm{f}(V,V,I) = V$
$\underline{f(I,V,V) = V}$	$\underline{\mathrm{f}(I,V,V) = I}$
$\underline{f(I,V,F) = V}$	$\underline{\mathrm{f}(I,V,F) = I}$
$\underline{f(I,F,V) = F}$	$\underline{\mathrm{f}(I,F,V) = I}$
$f(V,I,F) = V$	$\mathrm{f}(V,I,F) = V$
$f(V,F,I) = I$	$\mathrm{f}(V,F,I) = I$
$f(I,F,F) = I$	$\mathrm{f}(I,F,F) = I$
$f(F,I,V) = I$	$\mathrm{f}(F,I,V) = I$
$f(F,V,I) = V$	$\mathrm{f}(F,V,I) = V$
$\underline{f(F,F,I) = F}$	$\underline{\mathrm{f}(F,F,V) = I}$
$f(F,I,F) = I$	$\mathrm{f}(F,I,F) = I$
$f(I,I,V) = I$	$\mathrm{f}(I,I,V) = I$
$f(V,I,I) = I$	$\mathrm{f}(V,I,I) = I$
$\underline{f(I,V,I) = V}$	$\underline{\mathrm{f}(I,V,I) = I}$
$f(I,F,I) = I$	$\mathrm{f}(I,F,I) = I$
$f(I,I,F) = I$	$\mathrm{f}(I,I,F) = I$
$f(F,F,I) = I$	$\mathrm{f}(F,F,I) = I$
$f(I,I,I) = I$	$\mathrm{f}(I,I,I) = I$

On choisit pour f les mêmes valeurs que pour f sauf que l'on a:

$\underline{f(I,V,V) = V}$	$\underline{\mathrm{f}(I,V,V) = I}$
$\underline{f(I,V,F) = V}$	$\underline{\mathrm{f}(I,V,F) = I}$
$\underline{f(I,V,I) = V}$	$\underline{\mathrm{f}(I,V,I) = I}$
$\underline{f(I,F,V) = F}$	$\underline{\mathrm{f}(I,F,V) = I}$
$\underline{f(F,F,I) = F}$	$\underline{\mathrm{f}(F,F,V) = I}$

D'où la construction de f à partir de f:

$$\begin{aligned}\mathrm{f}(a_1, a_2, a_3) = (&F_{(I,F,V)}(a_1, a_2, a_3) \vee F_{(F,F,I)}(a_1, a_2, a_3)) \vee \\ &(V_{(I,V,V)}(a_1, a_2, a_3) \wedge V_{(I,V,I)}(a_1, a_2, a_3) \wedge \\ &V_{(I,V,F)}(a_1, a_2, a_3) \wedge f(a_1, a_2, a_3))\end{aligned}$$

Ce qui donne sachant que:

$$f(a_1, a_2, a_3) = a_2 \vee (a_1 \wedge \neg a_3).$$

$$\begin{aligned}\mathrm{f}(a_1, a_2, a_3) = (&(a_1 \wedge \neg a_1 \wedge \neg a_2 \wedge a_3) \vee (a_3 \wedge \neg a_3 \wedge \neg a_1 \wedge \neg a_2)) \vee \\ &((a_1 \vee \neg a_1 \vee \neg a_2 \vee \neg a_3) \wedge (a_1 \vee \neg a_1 \vee \neg a_2 \vee a_3 \vee \neg a_3) \\ &\quad (a_1 \vee \neg a_1 \vee \neg a_2 \vee a_3) \wedge (a_2 \vee (a_1 \wedge \neg a_3))).\end{aligned}$$

On peut aisément vérifier que l'expression ainsi trouvée redonne les valeurs de vérité de f.

CHAPITRE 4: TROIS THEOREMES DE COMPLETUDE EN LOGIQUE TRIVALUEE

4.1 INTRODUCTION

Comme nous l'avons dit dans l'introduction, par un souci à la fois de rigueur et de généralisation, nous avons voulu étudier les propriétés de cette logique trivaluée et en particulier, sa complétude.
Cette logique se révèle posséder de bonnes propriétés théoriques et les trois théorèmes de complétude obtenus nous confortent dans le choix du connecteur d'implication.
Nous étudions ici ces trois théorèmes de complétude dont deux sont obtenus en calcul propositionnel et l'un en calcul des prédicats.

En calcul propositionnel, si A est une formule comportant uniquement les connecteurs $\neg$ et $\rightarrow$, nous obtenons l'équivalence entre une propriété syntaxique et une propriété sémantique suivante:

$$\vdash A \Leftrightarrow \models_T A$$

Le cas où A ne comporte que les connecteurs $\neg$, $\rightarrow$, et $\wedge$ est analogue.

En calcul des prédicats, si A est une formule ne comportant que les connecteurs $\neg$ et $\rightarrow$ et le quantificateur $\exists$, nous obtenons aussi l'équivalence:

$$\vdash A \Leftrightarrow \models_T A.$$

L'expression $\models_T$ A signifie que A est vraie dans toute interprétation trivaluée. L'expression $\vdash$ A signifie que A est obtenu par déduction à partir d'un système formel défini dans chacun des cas.

Ou encore qu'il existe une suite $A_1, A_2, ..., A_n$ telle que $A_n = A$ et A_i soit ou un axiome du système ou obtenu à partir de A_j et A_k, avec $i, j < k$ par une règle d'inférence du même système.

L'ensemble des axiomes du système est défini dans chacun des cas.

Dans les deux premiers cas la règle d'inférence utilisée est le modus ponens (m.p.) alors que dans le dernier cas on peut aussi utiliser la généralisation universelle (u.g.).

La démonstration du dernier théorème de complétude fournit des résultats intermédiaires intéressants pour cette logique trivaluée.

Ces théorèmes de complétude permettent aussi de montrer que l'ensemble des tautologies obtenues dans chaque cas est récursivement énumérable puisque l'ensemble des théorèmes obtenus à partir d'un système formel est récursivement énumérable.

Le cadre de travail est celui défini au chapitre 2, en 2.1 pour le calcul propositionnel et en 2.2 pour le calcul des prédicats.

4.2 NOTATIONS EMPLOYEES

On définit les notations employées dans les différents théorèmes:

Définitions 4.2.1 En calcul propositionnel (resp. en calcul des prédicats), $\vdash A$ signifie qu'il existe une suite finie $A_1, A_2, ..., A_n$, telle que $A_n = A$ et A_i soit ou bien un axiome ou obtenu à partir de A_j et de A_k avec $j, k < i$ par le modus ponens (m.p.:$B, B \rightarrow C \vdash C$) (resp. par le m.p. et la généralisation universelle (u.g.: $A \vdash \forall x_i A$)).

Si Δ est un ensemble de formules du langage, $\Delta \vdash A$ signifie qu'il existe une suite finie $A_1, A_2, ..., A_n$, telle que $A_n = A$ et A_i soit ou bien un axiome, ou un élément de Δ ou obtenu à partir de A_j et de A_k avec $j, k < i$ par le modus ponens et éventuellement la généralisation universelle en calcul des prédicats.
On le note aussi $\vdash_\Delta A$.

Définition 4.2.2 On note $Ind_1(B)$ la formule:

$$\neg((B \to \neg B) \to \neg(\neg B \to B)).$$

La formule $\mathrm{Ind}_1(B)$ est logiquement équivalente à:

$$(B \leftrightarrow \neg B)$$

et à:

$$(B \leftarrow \rightarrow \neg B)$$

et à:

$$(\neg B \to B) \wedge (B \to \neg B).$$

$\mathrm{Ind}_2(B)$ denote la formule:

$$(\neg B \to B) \wedge (B \to \neg B).$$

(utilisée dans le théorème 4.4.1).

On a $\mathrm{Ind}_1(B) \equiv_T \mathrm{Ind}_2(B) \equiv_T \mathrm{Ind}(B)$ où $\mathrm{Ind}(B)$ est la formule introduite dans le chapitre 3 (3.2).

Remarque: Ces formules ont la valeur Vrai si B a la valeur Indeterminé, la valeur Faux sinon.

4.3 PREMIER THEOREME DE COMPLETUDE

Théorème 4.3.1 *On considère le système formel dont les axiomes sont les axiomes $A_1, A_2, ..., A_9$ suivants*:

A_1 : $A \to (B \to A)$;
A_2 : $(A \to (B \to C)) \to ((A \to B) \to (A \to C))$;
A_3 : $A \to (\neg A \to B)$;
A_4 : $B \to (\neg C \to \neg(B \to C))$;
A_5 : $A \to \neg\neg A$;
A_6 : $Ind_1(B) \to Ind_1(\neg B)$;
A_7 : $Ind_1(C) \to (B \to \neg(B \to C))$;
A_8 : $\neg(A \to B) \to A$;
A_9 : $(p \to A) \to ((\neg p \to A) \to ((Ind_1(p) \to A) \to A))$;

et dont la règle d'inférence est le modus ponens: $A \to B, A \vdash B$;

Si A est une formule du langage L ayant pour seuls connecteurs $\neg$, $\to$, *on a:*

$$\vdash A \Leftrightarrow \models_T A.$$

Démonstration $\vdash A$ implique $\models_T A$ se démontre par induction sur la longueur de la preuve dans $\vdash A$.

On vérifie que les axiomes sont des tautologies; c'est-à-dire que pour toute interprétation i, $vv_i(A_j) = \text{Vrai}$, pour tout $j = 1, ..., 9$.

A_1: Si $vv_i(A) = V$ alors $vv_i(B \to A) = V$ d'après la table de vérité de $\to$.

A_2: Si $vv_i(A) = I$ ou F alors $vv_i(A \to B) = V$ et $vv_i(A \to C) = V$.

Si $vv_i(A) = V$ et $vv_i(B \to C) = V$ alors soit $vv_i(B) = I$ ou F et $vv_i(A \to B) = F$, soit $vv_i(B) = V$ et $vv_i(C) = V$ auquel cas $vv_i(A \to B) = V$ et $vv_i(A \to C) = V$.

A_3: Si $vv_i(A) = V$ alors $vv_i(\neg A) = F$ et $vv_i(\neg A \to B) = V$.

A_4: Si $vv_i(B) = V$ et $vv_i(\neg C) = V$ alors $vv_i(B \to C) = F$ et $vv_i(\neg(B \to C)) = V$.

A_5: Si $vv_i(A) = V$ alors $vv_i(\neg\neg A) = V$.

A_6: Si $vv_i(Ind_1(B)) = V$, alors $vv_i(B) = I$ et $vv_i(\neg B) = I$ donc $vv_i(Ind_1(\neg B)) = V$.

A_7: Si $vv_i(Ind_1(C)) = V$, alors $vv_i(C) = I$ et si $vv_i(B) = V$ alors $vv_i(B \to C) = F$ et $vv_i(\neg(B \to C)) = V$.

A_8: Si $vv_i(\neg(A \to B)) = V$ alors $vv_i(A) = V$ et $vv_i(B) = F$ ou I. Dans ce cas $vv_i(\neg(A \to B) \to A) = V$.

A_9: Si $vv_i(p \to A) = V$, $vv_i(\neg p \to A) = V$ et $vv_i(Ind_1(p) \to A) = V$, alors comme p est soit V, soit F, soit I, on a $vv_i(p) = V$, ou $vv_i(\neg p) = V$, ou $vv_i(Ind_1(p)) = V$ et donc $vv_i(A) = V$.

Si $n > 1$ et s'il existe une suite finie $A_1, A_2, ..., A_n$, telle que $A_n = A$ et A_i soit ou bien un axiome, ou obtenu à partir de A_j et de A_k avec $j, k < i$ par le modus ponens, le seul cas à envisager est celui où A a été obtenu par modus ponens à partir de A_i, A_j, on a $vv_i(A_i) = V$ et $vv_i(A_i \rightarrow A) = V$ par induction, donc $vv_i(A) = V$ d'après la table de vérité de $\rightarrow$.

Pour montrer la réciproque, $\models_T A$ implique $\vdash A$:
1) On démontre d'abord le théorème de déduction:

Théorème 4.3.1.1 *Si $\Delta \cup \{p, q\}$ est un ensemble de formules du langage L,*

$$\Delta \cup \{p\} \vdash q$$

est equivalent à:

$$\Delta \vdash (p \rightarrow q).$$

Démonstration $\Delta \vdash (p \rightarrow q)$ implique $\Delta \cup \{p\} \vdash q$ par application du modus ponens.

On démontre que: $\Delta \cup \{p\} \vdash q$ implique $\Delta \vdash (p \rightarrow q)$ par induction sur la longueur n de la preuve dans $\Delta \cup \{p\} \vdash q$.

Si $n = 1$, soit $q = p$, soit q est un axiome, ou enfin $q \in \Delta$.

- Démontrons que l'on a $\vdash (p \rightarrow p)$:

A_1: $p \rightarrow (q \rightarrow p)$;

A_2: $(p \rightarrow (q \rightarrow p)) \rightarrow ((p \rightarrow q) \rightarrow (p \rightarrow p))$;

Par le modus ponens, on en déduit $(p \rightarrow q) \rightarrow (p \rightarrow p)$.
En remplaçant q par $(p \rightarrow p)$, on obtient $(p \rightarrow (p \rightarrow p)) \rightarrow (p \rightarrow p)$.
En utilisant A_1: $p \rightarrow (p \rightarrow p)$ et le modus ponens, on a le résultat.

- Si q est un axiome A_i, démontrons que l'on a $\vdash (p \rightarrow A_i)$:

A_1: $A_i \rightarrow (p \rightarrow A_i)$; avec A_i et le modus ponens on en déduit $\vdash (p \rightarrow A_i)$;

- Si $q \in \Delta$, démontrons que l'on a $\Delta \vdash (p \rightarrow q)$:

A_1: $q \rightarrow (p \rightarrow q)$; avec q et le modus ponens on en déduit $\Delta \vdash (p \rightarrow q)$;

Supposons maintenant $n > 1$. Il y a une suite $q_1, q_2, ..., q_n$ telle que $q_n = q$ et q_i soit ou un axiome ou un élément de Δ ou p ou obtenu par m.p. à partir de q_j et de q_k, avec $j,k < i$.

Le seul cas à envisager est le cas où q_n est obtenu par m.p. à partir de q_j et de q_k, avec $j,k < n$, les autres cas ayant été traités pour $n = 1$.

On a: $\Delta \vdash (p \to q_j)$ et $\Delta \vdash (p \to q_k)$ par induction; de plus $q_k = q_j \to q_n$; par A_2 et le m.p., on en déduit $\Delta \vdash (p \to q_j) \to (p \to q_n)$; en appliquant une nouvelle fois le m.p. on obtient $\Delta \vdash (p \to q_n)$. ♦

2) Démontrons maintenant que: $\models_T A$ implique $\vdash A$.

On considère pour toute interprétation v, la formule A^v définie par:

$A^v = A$ si $vv_v(A) =$ Vrai; $A^v = \neg A$ si $vv_v(A) =$ Faux;

$A^v = Ind_1(A) = \neg((A \to \neg A) \to \neg(\neg A \to A))$ si $vv_v(A) =$ Indéterminé;

a) On démontre d'abord que $p_1^v, p_2^v, ..., p_n^v \vdash A^v$ par induction sur le nombre k de connecteurs de la formule A, si $p_1, p_2, ..., p_n$ sont les variables propositionnelles de A (les atomes de A).

- Si $k = 0$, $A = p$ (A est un atome);

On a bien $p^v \vdash p^v$.

- Si $k > 0$, $A = B \to C$ ou $A = \neg B$;

- <u>$A = \neg B$</u>.

Soit $vv_v(A) =$ Vrai et alors $vv_v(B) =$ Faux et $B^v = \neg B = A^v$ et $p_1^v, p_2^v, ..., p_n^v \vdash B^v$ par induction, donc $p_1^v, p_2^v, ..., p_n^v \vdash A^v$.

Soit $vv_v(A) =$ Faux et alors $vv_v(B) =$ Vrai, $B^v = B$ et $A^v = \neg\neg B$. D'après A_5, $B \to \neg\neg B$. Par induction, $p_1^v, p_2^v, ..., p_n^v \vdash B^v$, donc $p_1^v, p_2^v, ..., p_n^v \vdash A^v$ car $B^v \vdash A^v$ (par A_5 et le m.p.).

Soit $vv_v(A) =$ Indéterminé, et alors $vv_v(B) = I$, $B^v = Ind_1(B)$ et $A^v = Ind_1(\neg B)$. D'après A_6, $Ind_1(B) \to Ind_1(\neg B)$. Par induction, $p_1^v, p_2^v, ..., p_n^v \vdash B^v$, donc $p_1^v, p_2^v, ..., p_n^v \vdash A^v$ car $B^v \vdash A^v$ (par A_6 et le m.p.).

- <u>$A = B \to C$</u>.

On a toujours $p_1^V, p_2^V, ..., p_n^V \vdash B^V$ et $p_1^V, p_2^V, ..., p_n^V \vdash C^V$ par récurrence.

Nous allons montrer que: $B^V, C^V \vdash (B \rightarrow C)^V$.

9 cas sont à envisager.

1) $vv_V(B)$ = Vrai et $vv_V(C)$ = Vrai, dans ce cas , $B^V = B$, $C^V = C$ et $vv_V(B \rightarrow C)$ = Vrai donc $(B \rightarrow C)^V = B \rightarrow C$.

On a: $B, C \vdash (B \rightarrow C)$; en effet d'après A_7: $C \rightarrow (B \rightarrow C)$ et le m.p., on a: $C \vdash (B \rightarrow C)$.

2) $vv_V(B)$ = Vrai et $vv_V(C)$ = Faux, dans ce cas , $B^V = B$, $C^V = \neg C$ et $vv_V(B \rightarrow C)$ = Faux donc $(B \rightarrow C)^V = \neg(B \rightarrow C)$.

On a: $B, \neg C \vdash \neg(B \rightarrow C)$ d'après A_4: $B \rightarrow (\neg C \rightarrow \neg(B \rightarrow C))$, et d'après le m.p.

3) $vv_V(B)$ = Vrai et $vv_V(C)$ = I, dans ce cas , $B^V = B$, $C^V = Ind_1(C)$, et $vv_V(B \rightarrow C)$ = Faux donc $(B \rightarrow C)^V = \neg(B \rightarrow C)$.

On a: $B, Ind_1(C) \vdash \neg(B \rightarrow C)$ d'après A_7:
$Ind_1(C) \rightarrow (B \rightarrow \neg(B \rightarrow C))$, et d'après le m.p.

4) $vv_V(B)$ = Faux et $vv_V(C)$ = Vrai, dans ce cas , $B^V = \neg B$, $C^V = C$, et $vv_V(B \rightarrow C)$ = Vrai donc $(B \rightarrow C)^V = (B \rightarrow C)$.

On a: $\neg B, C \vdash (B \rightarrow C)$ puisque $C \vdash (B \rightarrow C)$ d'après A_1: $C \rightarrow (B \rightarrow C)$, et d'après le m.p.

5) $vv_V(B)$ = Faux et $vv_V(C)$ = Faux, dans ce cas , $B^V = \neg B$, $C^V = \neg C$, et $vv_V(B \rightarrow C)$ = Vrai donc $(B \rightarrow C)^V = (B \rightarrow C)$.

On a: $\neg B, \neg C \vdash (B \rightarrow C)$; en effet d'après A_3: $B \rightarrow (\neg B \rightarrow C)$, et d'après le m.p, $\neg B, B \vdash C$, donc par le théorème de déduction, $\neg B \vdash (B \rightarrow C)$.

6) $vv_V(B)$ = Faux et $vv_V(C)$ = I, dans ce cas , $B^V = \neg B$, $C^V = Ind_1(C)$, et $vv_V(B \rightarrow C)$ = Vrai donc $(B \rightarrow C)^V = (B \rightarrow C)$.

On a: $\neg B$, $Ind_1(C) \vdash (B \to C)$; en effet d'après A_3: $B \to (\neg B \to C)$, et d'après le m.p, $\neg B, B \vdash C$, donc par le théorème de déduction, $\neg B \vdash (B \to C)$.

7) $vv_v(B) = I$ et $vv_v(C) = Vrai$, dans ce cas , $B^v = Ind_1(B)$, $C^v = C$, et $vv_v(B \to C) = Vrai$ donc $(B \to C)^v = (B \to C)$.

On a: $Ind_1(B), C \vdash (B \to C)$; en effet d'après A_1: $C \to (B \to C)$, et d'après le m.p, $C \vdash B \to C$.

8) $vv_v(B) = I$ et $vv_v(C) = Faux$, dans ce cas , $B^v = Ind_1(B)$, $C^v = \neg C$, et $vv_v(B \to C) = Vrai$ donc $(B \to C)^v = (B \to C)$.

On a: $Ind_1(B), \neg C \vdash (B \to C)$; en effet d'après A_8: $\neg(A \to B) \to A$, et d'après le m.p, $\neg(A \to B) \vdash A$; par conséquent, $\neg((B \to \neg B) \to \neg(\neg B \to B)) \vdash (B \to \neg B)$ ou encore $Ind_1(B) \vdash (B \to \neg B)$;

or d'après A_2: $B \to (\neg B \to C)$,

d'après A_3: $(A \to (B \to C)) \to ((A \to B) \to (A \to C))$,

et d'après le m.p., on a $\vdash (B \to \neg B) \to (B \to C)$ donc par le m.p., on a: $Ind_1(B) \vdash (B \to C)$.

9) $vv_v(B) = I$ et $vv_v(C) = I$, dans ce cas , $B^v = Ind_1(B)$, $C^v = Ind_1(C)$, et $vv_v(B \to C) = Vrai$ donc $(B \to C)^v = (B \to C)$.

On a: $Ind_1(B), Ind_1(C) \vdash (B \to C)$; en effet:

$Ind_1(B) \vdash (B \to C)$ de même que précédemment.

On a donc montré que: $p_1^v, p_2^v, ..., p_n^v \vdash A^v$, et par conséquent que $p_1^v, p_2^v, ..., p_n^v \vdash A$ puisque A est une tautologie.

b) On montre ensuite par une induction décroissante que: $p_1^v, p_2^v, ..., p_k^v \vdash A$, pour tout $1 \leq k \leq n$.

Pour $k = n$, le résultat est démontré ci-dessus.

Démontrons que $p_1^v, p_2^v, ..., p_{k+1}^v \vdash A$ implique que $p_1^v, p_2^v, ..., p_k^v \vdash A$.

On a $p_1^v, p_2^v, ..., p_k^v, p_{k+1} \vdash A$, $p_1^v, p_2^v, ..., p_k^v, \neg p_{k+1} \vdash A$, et $p_1^v, p_2^v, ..., p_k^v, Ind_1(p_{k+1}) \vdash A$ puisque $p_1^v, p_2^v, ..., p_{k+1}^v \vdash A$ est vrai pour n'importe quelle interprétation v.

Par le théorème de déduction, on en déduit:

$p_1^v, p_2^v, ..., p_k^v \vdash (p_{k+1} \rightarrow A)$,
$p_1^v, p_2^v, ..., p_k^v \vdash (\neg p_{k+1} \rightarrow A)$,
$p_1^v, p_2^v, ..., p_k^v \vdash (Ind_1(p_{k+1}) \rightarrow A)$.

Par A_9: $(p \rightarrow A) \rightarrow ((\neg p \rightarrow A) \rightarrow ((Ind_1(p) \rightarrow A) \rightarrow A))$,

et par le m.p., on a: $p_1^v, p_2^v, ..., p_k^v \vdash A$.

c) On a donc $p^v \vdash A$, pour toute interprétation v.

Donc $p \vdash A$, $\neg p \vdash A$, $Ind_1(p) \vdash A$, en choisissant différentes interprétations.

Avec le théorème de déduction, on en déduit:

$\vdash (p \rightarrow A)$, $\vdash (\neg p \rightarrow A)$ et $\vdash (Ind_1(p) \rightarrow A)$;

avec A_9 et le m.p.,on obtient $\vdash A$ ♦

Corollaire 4.3.1 *L'ensemble des tautologies du langage L, en calcul propositionnel, ayant pour seuls connecteurs $\neg$, $\rightarrow$, est récursivement énumérable.*

4.4 DEUXIEME THEOREME DE COMPLETUDE

Théorème 4.4.1 *On considère le système formel constitué des axiomes $A_1, A_2, ..., A_{17}$ suivants:*

A_1 : $A \rightarrow (B \rightarrow A)$;
A_2 : $(A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))$;
A_3 : $A \rightarrow (\neg A \rightarrow B)$;
A_4 : $B \rightarrow (\neg C \rightarrow \neg(B \rightarrow C))$;
A_5 : $A \rightarrow \neg\neg A$;

A_6 : $\neg\neg A \rightarrow A$;
A_7 : $A \rightarrow ((\neg B \rightarrow B) \rightarrow ((B \rightarrow \neg B) \rightarrow \neg(A \rightarrow B)))$;
A_8 : $\neg A \rightarrow \neg(A \wedge B)$;
A_9 : $\neg B \rightarrow \neg(A \wedge B)$;
A_{10} : $A \rightarrow ((\neg B \rightarrow B) \rightarrow ((B \rightarrow \neg B) \rightarrow Ind_2(A \wedge B)))$;
A_{11} : $(\neg A \rightarrow A) \rightarrow ((A \rightarrow \neg A) \rightarrow (B \rightarrow (Ind_2(A \wedge B)))$;
A_{12} : $(\neg A \rightarrow A) \rightarrow ((A \rightarrow \neg A) \rightarrow$
$$((\neg B \rightarrow B) \rightarrow ((B \rightarrow \neg B) \rightarrow Ind_2(A \wedge B)));$$
A_{13} : $(A \rightarrow B) \rightarrow ((B \rightarrow C) \rightarrow (A \rightarrow C))$;
A_{14} : $(A \rightarrow B) \rightarrow ((\neg A \rightarrow B) \rightarrow ((Ind_2(A) \rightarrow B) \rightarrow B))$;
A_{15} : $A \rightarrow (B \rightarrow (A \wedge B))$;
A_{16} : $(A \wedge B) \rightarrow A$;
A_{17} : $(A \wedge B) \rightarrow B$
et de la règle d'inférence: modus ponens.

Si A est une formule ayant pour seuls connecteurs $\neg$, $\wedge$ et $\rightarrow$, on a l'équivalence suivante:

$$\vdash A \Leftrightarrow \models_T A.$$

Démonstration $\vdash$ A implique $\models_T$ A se démontre par induction sur la longueur de la preuve dans $\vdash$ A.

On vérifie que les axiomes sont des tautologies. Pour toute interprétation i, $vv_i(A_j)$ = Vrai, pour tout j = 1, ..., 17.

Les 5 premiers axiomes sont ceux du théorème 4.3.1.

A_6: Si $vv_i(\neg\neg A) = V$ alors $vv_i(A) = V$.

A_7: Si $vv_i(A) = V$, $vv_i(\neg B \rightarrow B)) = V$ et $vv_i(B \rightarrow \neg B) = V$ alors $vv_i(B) = I$ et $vv_i(\neg(A \rightarrow B)) = V$.

A_8: Si $vv_i(\neg A) = V$ alors $vv_i(A \wedge B) = F$ et $vv_i(\neg A \rightarrow \neg(A \wedge B)) = V$.

A_9: De même que A_8.

A_{10}: Si $vv_i(A) = V$, $vv_i(\neg B \rightarrow B)) = V$ et $vv_i(B \rightarrow \neg B) = V$ alors $vv_i(B) = I$ et $vv_i(A \wedge B) = I$; par conséquent $vv_i((Ind_2(A \wedge B)) = V$.

A_{11}: Si $vv_i(B) = V$, $vv_i(\neg A \rightarrow A)) = V$ et $vv_i(A \rightarrow \neg A) = V$ alors $vv_i(A) = I$ et $vv_i(A \wedge B) = I$; par conséquent $vv_i((Ind_2(A \wedge B)) = V$.

A_{12}: Si $vv_i(\neg A \to A)) = V$ et $vv_i(A \to \neg A) = V$, $vv_i(\neg B \to B)) = V$ et $vv_i(B \to \neg B) = V$ alors $vv_i(A) = I$, $vv_i(B) = I$ et $vv_i(A \wedge B) = I$; par conséquent $vv_i((Ind_2(A \wedge B)) = V$.

A_{13}: Si $vv_i(A \to B) = V$ et $vv_i(B \to C) = V$ alors soit $vv_i(A) = I$ ou F et $vv_i(A \to C) = V$, soit $vv_i(A) = V$ et $vv_i(B) = V$ auquel cas et $vv_i(C) = V$ et $vv_i(A \to C) = V$.

A_{14}: Si $vv_i(A \to B) = V$, $vv_i(\neg A \to B) = V$ et $vv_i(Ind_2(A) \to B) = V$, alors comme $vv_i(A) = V$, F ou I, $vv_i(A) = V$ ou $vv_i(\neg A) = V$ ou $vv_i(Ind_2(A)) = V$ et $vv_i(B) = V$. (Cet axiome est l'analogue de l'axiome A_9 du théorème 4 3.1 avec Ind_2 au lieu de Ind_1).

A_{15}: Si $vv_i(A) = V$ et $vv_i(B) = V$ alors $vv_i(A \wedge B) = V$

A_{16}: Si $vv_i(A \wedge B) = V$ alors $vv_i(A) = V$.

A_{17}: Si $vv_i(A \wedge B) = V$ alors $vv_i(B) = V$.

Si A a été obtenu par modus ponens à partir de A_i et A_j on a $vv_i(A) = V$ de même que dans le théorème 4 3.1.

Le théorème 4.3.1.1 reste valable puisque la règle d'inférence est toujours la même. On a donc: Si $\Delta \cup \{p, q\}$ est un ensemble de formules du langage L,

$$\Delta \cup \{p\} \vdash q \text{ est equivalent à } \Delta \vdash (p \to q).$$

On démontre que: $\models_T A$ implique $\vdash A$ comme dans le théorème 4.3.1.

On considère pour toute interprétation v, la formule A^v définie par:

$A^v = A$ si $vv_v(A) =$ Vrai; $A^v = \neg A$ si $vv_v(A) =$ Faux;
$A^v = Ind_2(A) = (\neg A \to A) \wedge (A \to \neg A)$ si $vv_v(A) =$ Indéterminé;

a) On démontre d'abord que $p_1{}^v, p_2{}^v, ..., p_n{}^v \vdash A^v$ par induction sur le nombre k de connecteurs de la formule A, si $p_1, p_2, ..., p_n$ sont les variables propositionnelles de A (les atomes de A).

-Si $k = 0$, $A = p$ (A est un atome);

On a bien $p^v \vdash p^v$.

-Si $k > 0$, $A = B \rightarrow C$ ou $A = \neg B$ ou $A = B \wedge C$;

- <u>$A = \neg B$</u>.

Soit $vv_v(A)$ = Vrai et alors $vv_v(B)$ = Faux et $B^v = \neg B = A^v$ et par induction $p_1^v, p_2^v, ..., p_n^v \vdash B^v$.

Soit $vv_v(A)$ = Faux et alors $vv_v(B)$ = Vrai, $B^v = B$ et $A^v = \neg\neg B$. D'après A_5, $B \rightarrow \neg\neg B$. Par induction, $p_1^v, p_2^v, ..., p_n^v \vdash B^v$, donc $p_1^v, p_2^v, ..., p_n^v \vdash A^v$ car $B^v \vdash A^v$.

Soit $vv_v(A)$ = Indéterminé, et alors $vv_v(B)$ = I, $B^v = Ind_2(B)$ et $A^v = Ind_2(\neg B)$. Par induction, $p_1^v, p_2^v, ..., p_n^v \vdash B^v = (\neg B \rightarrow B) \wedge (B \rightarrow \neg B)$, avec A_{16} et le m.p., on en déduit $p_1^v, p_2^v, ..., p_n^v \vdash (\neg B \rightarrow B)$, avec A_{17} et le m.p., on en déduit $p_1^v, p_2^v, ..., p_n^v \vdash (B \rightarrow \neg B)$. Avec A_{13}:

$$(\neg\neg B \rightarrow B) \rightarrow ((B \rightarrow \neg B) \rightarrow (\neg\neg B \rightarrow \neg B)),$$

et le m.p. on en déduit:

$\neg\neg B \rightarrow B, B \rightarrow \neg B \vdash \neg\neg B \rightarrow \neg B$. Donc,

avec A_6: $\neg\neg B \rightarrow B$, on en déduit $p_1^v, p_2^v, ..., p_n^v \vdash \neg\neg B \rightarrow \neg B$. (1)

Avec A_{13}: $(\neg B \rightarrow B) \rightarrow ((B \rightarrow \neg\neg B) \rightarrow (\neg B \rightarrow \neg\neg B))$,

et le m.p. on en déduit:

$B \rightarrow \neg\neg B, \neg B \rightarrow B \vdash \neg B \rightarrow \neg\neg B$.

Avec A_5: $B \rightarrow \neg\neg B$, on en déduit $p_1^v, p_2^v, ..., p_n^v \vdash \neg B \rightarrow \neg\neg B$. (2)

Avec A_{15}: $A \rightarrow (B \rightarrow (A \wedge B))$ et le m.p on déduit de (1) et (2):

$p_1^v, p_2^v, ..., p_n^v \vdash (\neg\neg B \rightarrow \neg B) \wedge (\neg B \rightarrow \neg\neg B) = A^v = Ind_2(\neg B)$.

Donc $p_1^v, p_2^v, ..., p_n^v \vdash A^v$ car $B^v \vdash A^v$.

- $\underline{A = B \rightarrow C}$.

On a toujours $p_1^V, p_2^V, ..., p_n^V \vdash B^V$ et $p_1^V, p_2^V, ..., p_n^V \vdash C^V$ par induction.

Nous allons montrer que: $B^V, C^V \vdash (B \rightarrow C)^V$.

9 cas sont à envisager.

1) $vv_V(B)$ = Vrai et $vv_V(C)$ = Vrai, dans ce cas , $B^V = B$, $C^V = C$, et $vv_V(B \rightarrow C)$ = Vrai donc $(B \rightarrow C)^V = B \rightarrow C$.

On a: $B, C \vdash (B \rightarrow C)$ car d'après A_1: $C \rightarrow (B \rightarrow C)$ et le m.p.,$C \vdash (B \rightarrow C)$.

2) $vv_V(B)$ = Vrai et $vv_V(C)$ = Faux, dans ce cas $B^V = B$, $C^V = \neg C$, et $vv_V(B \rightarrow C)$ = Faux donc $(B \rightarrow C)^V = \neg(B \rightarrow C)$.

On a: $B, \neg C \vdash \neg(B \rightarrow C)$ d'après A_4: $B \rightarrow (\neg C \rightarrow \neg(B \rightarrow C))$, et d'après le m.p.

3) $vv_V(B)$ = Vrai et $vv_V(C)$ = I, dans ce cas, $B^V = B$, $C^V = Ind_2(C)$, et $vv_V(B \rightarrow C)$ = Faux donc $(B \rightarrow C)^V = \neg(B \rightarrow C)$.

On a: $B, Ind_2(C) \vdash \neg(B \rightarrow C)$.

En effet $Ind_2(C) \vdash \neg C \rightarrow C$, et $Ind_2(C) \vdash C \rightarrow \neg C$,
d'après A_{16}, A_{17} et le m.p.

D'après A_7: $B \rightarrow ((\neg C \rightarrow C) \rightarrow ((C \rightarrow \neg C) \rightarrow \neg(B \rightarrow C)))$, et le m.p. on a le résultat

4) $vv_V(B)$ = Faux et $vv_V(C)$ = Vrai, dans ce cas , $B^V = \neg B$, $C^V = C$, et $vv_V(B \rightarrow C)$ = Vrai donc $(B \rightarrow C)^V = (B \rightarrow C)$.

On a: $\neg B, C \vdash (B \rightarrow C)$ puisque $C \vdash (B \rightarrow C)$ d'après A_1: $C \rightarrow (B \rightarrow C)$, et le m.p.

5) $vv_V(B)$ = Faux et $vv_V(C)$ = Faux, dans ce cas , $B^V = \neg B$, $C^V = \neg C$, donc $vv_V(B \rightarrow C)$ = Vrai et $(B \rightarrow C)^V = (B \rightarrow C)$.

On a: $\neg B, \neg C \vdash (B \to C)$; en effet d'après A_3: $B \to (\neg B \to C)$, et d'après le m.p, $\neg B, B \vdash C$, donc par le théorème de déduction, $\neg B \vdash (B \to C)$.

6) $vv_v(B)$ = Faux et $vv_v(C) = I$, dans ce cas, $B^v = \neg B$, $C^v = Ind_2(C)$, et $vv_v(B \to C)$ = Vrai donc $(B \to C)^v = (B \to C)$.

On a: $\neg B, Ind_2(C) \vdash (B \to C)$; puisque $\neg B \vdash (B \to C)$, d'après 5).

7) $vv_v(B) = I$ et $vv_v(C)$ = Vrai, dans ce cas , $B^v = Ind_2(B)$, $C^v = C$, et $vv_v(B \to C)$ = Vrai donc $(B \to C)^v = (B \to C)$.

On a: $Ind_2(B), C \vdash (B \to C)$; en effet d'après A_1: $C \to (B \to C)$, et d'après le m.p, $C \vdash B \to C$.

8) $vv_v(B) = I$ et $vv_v(C)$ = Faux, dans ce cas, $B^v = Ind_2(B)$, $C^v = \neg C$, et $vv_v(B \to C)$ = Vrai donc $(B \to C)^v = (B \to C)$.

On a: $Ind_2(B), \neg C \vdash (B \to C)$; en effet $Ind_2(B) \vdash (B \to C)$;

D'après A_{17} et le m.p., $Ind_2(B) \vdash B \to \neg B$.

D'après A_2: $B \to (\neg B \to C)$,

A_3: $(A \to (B \to C)) \to ((A \to B) \to (A \to C))$ et le m.p.,

$(B \to \neg B) \to (B \to C)$ donc $Ind_2(B) \vdash (B \to C)$.

9) $vv_v(B) = I$ et $vv_v(C) = I$, dans ce cas , $B^v = Ind_2(B)$, $C^v = Ind_2(C)$, et $vv_v(B \to C)$ = Vrai donc $(B \to C)^v = (B \to C)$.

On a: $Ind_2(B), Ind_2(C) \vdash (B \to C)$; en effet $Ind_2(B) \vdash (B \to C)$ de même que précédemment.

- $\underline{A = B \wedge C}$.

On a toujours $p_1^v, p_2^v, \dots, p_n^v \vdash B^v$ et $p_1^v, p_2^v, \dots, p_n^v \vdash C^v$ par induction.

Nous allons montrer que: $B^v, C^v \vdash (B \wedge C)^v$.

9 cas sont à envisager.

1) $vv_V(B)$ = Vrai et $vv_V(C)$ = Vrai, dans ce cas , $B^V = B$, $C^V = C$, et $vv_V(B \wedge C)$ = Vrai donc $(B \wedge C)^V = B \wedge C$.

On a: $B, C \vdash (B \wedge C)$ d'après A_{15}: $B \rightarrow (C \rightarrow (B \wedge C)$ et le m.p.

2) $vv_V(B)$ = Vrai et $vv_V(C)$ = Faux, dans ce cas $C^V = \neg C$, et $vv_V(B \wedge C)$ = Faux donc $(B \wedge C)^V = \neg(B \wedge C)$.

On a: $\neg C \vdash \neg(B \wedge C)$ d'après A_9: $\neg C \rightarrow \neg(B \wedge C)$, et d'après le m.p.

3) $vv_V(B)$ = Vrai et $vv_V(C)$ = I, dans ce cas, $B^V = B$, $C^V = Ind_2(C)$, et $vv_V(B \wedge C)$ = I donc $(B \wedge C)^V = Ind_2(B \wedge C)$.

On a: $B, Ind_2(C) \vdash Ind_2(B \wedge C)$.

En effet $Ind_2(C) \vdash \neg C \rightarrow C, C \rightarrow \neg C$, d'après A_{16}, A_{17} et le m.p.
Or $B, \neg C \rightarrow C, C \rightarrow \neg C \vdash Ind_2(B \wedge C)$ par A_{10} et le m.p.

4) $vv_V(B)$ = Faux et $vv_V(C)$ = Vrai, dans ce cas , $B^V = \neg B$, $C^V = C$, et $vv_V(B \wedge C)$ = Faux donc $(B \wedge C)^V = \neg(B \wedge C)$.

On a: $\neg B, C \vdash \neg(B \wedge C)$ puisque $\neg B \vdash \neg(B \wedge C)$ d'après A_8 et le m.p.

5) $vv_V(B)$ = Faux et $vv_V(C)$ = Faux, dans ce cas , $B^V = \neg B$, $C^V = \neg C$, donc $vv_V(B \wedge C)$ = Faux et $(B \wedge C)^V = \neg(B \wedge C)$.

On a: $\neg B, \neg C \vdash (B \wedge C)$; en effet d'après A_8: $\neg B \rightarrow \neg(B \wedge C)$, et d'après le m.p., on a: $\neg B \vdash \neg(B \wedge C)$ le résultat

6) $vv_V(B)$ = Faux et $vv_V(C)$ = I, dans ce cas, $B^V = \neg B$, $C^V = Ind_2(C)$, et $vv_V(B \wedge C)$ = Faux donc $(B \wedge C)^V = \neg(B \wedge C)$.

On a: $\neg B, Ind_2(C) \vdash \neg(B \wedge C)$; puisque $\neg B \vdash \neg(B \wedge C)$, d'après 5).

7) $vv_V(B)$ = I et $vv_V(C)$ = Vrai, dans ce cas , $B^V = Ind_2(B)$, $C^V = C$, et $vv_V(B \wedge C)$ = I donc $(B \wedge C)^V = Ind_2(B \wedge C)$.

Or $Ind_2(B) \vdash (\neg B \to B)$, par A_{16} et le m.p, $Ind_2(B) \vdash (B \to \neg B)$, par A_{17} et le m.p.

$(\neg B \to B), (B \to \neg B), C \vdash Ind_2(B \wedge C)$ par par A_{11} et le m.p.

8) $vv_v(B) = I$ et $vv_v(C) = Faux$, dans ce cas, $B^v = Ind_2(B)$, $C^v = \neg C$, et $vv_v(B \wedge C) = Faux$ donc $(B \wedge C)^v = \neg(B \wedge C)$.

On a: $\neg C \vdash \neg(B \wedge C)$ par A_9 et le m.p.

9) $vv_v(B) = I$ et $vv_v(C) = I$, dans ce cas , $B^v = Ind_2(B)$, $C^v = Ind_2(C)$, et $vv_v(B \wedge C) = I$ donc $(B \wedge C)^v = Ind_2(B \wedge C)$.

On a: $Ind_2(B), Ind_2(C) \vdash Ind_2(B \wedge C)$; en effet $Ind_2(B) \vdash (\neg B \to B)$, par A_{16} et le m.p, $Ind_2(B) \vdash (B \to \neg B)$, par A_{17} et le m.p, de même en remplaçant B par C, donc on a le résultat avec A_{12} et le m.p.

On a donc montré que: $p_1^v, p_2^v, ..., p_n^v \vdash A^v$, et par conséquent que $p_1^v, p_2^v, ..., p_n^v \vdash A$ puisque A est une tautologie.

b) On montre ensuite par une induction décroissante que: $p_1^v, p_2^v, ..., p_k^v \vdash A$, pour tout $1 \leq k \leq n$.

Pour $k = n$, le résultat est démontré ci-dessus.

Démontrons que $p_1^v, p_2^v, ..., p_{k+1}^v \vdash A$ implique que $p_1^v, p_2^v, ..., p_k^v \vdash A$.

On a $p_1^v, p_2^v, ..., p_k^v, p_{k+1} \vdash A$, $p_1^v, p_2^v, ..., p_k^v, \neg p_{k+1} \vdash A$, et $p_1^v, p_2^v, ..., p_k^v, Ind_2(p_{k+1}) \vdash A$ puisque $p_1^v, p_2^v, ..., p_{k+1}^v \vdash A$ est vrai pour n'importe quelle interprétation v. Par le théorème de déduction, on en déduit:
$p_1^v, p_2^v, ..., p_k^v \vdash (p_{k+1} \to A)$, $p_1^v, p_2^v, ..., p_k^v \vdash (\neg p_{k+1} \to A)$, $p_1^v, p_2^v, ..., p_k^v \vdash (Ind_2(p_{k+1}) \to A)$.
Par A_{14}:$(p \to A) \to ((\neg p \to A) \to ((Ind_2(p) \to A) \to A))$; donc par le m.p., $p_1^v, p_2^v, ..., p_k^v \vdash A$.

c) On a donc $p^v \vdash A$, pour toute interprétation v.

Donc p $\vdash$ A, $\neg$p $\vdash$ A, $Ind_2(p)$ $\vdash$ A, en choisissant différentes interprétations. Avec le théorème de déduction, on en déduit:
$\vdash (p \to A)$, $\vdash (\neg p \to A)$ et $\vdash (Ind_2(p) \to A)$;
avec A_{14} et le m.p.,on obtient $\vdash$ A ♦

Corollaire 4.4.1 *L'ensemble des tautologies du langage L, en calcul propositionnel, ayant pour seuls connecteurs $\neg$, $\to$, $\wedge$ est récursivement énumérable.*

4.5 TROISIEME THEOREME DE COMPLETUDE

Le théorème de complétude en calcul des prédicats est plus difficile à obtenir. La démonstration qui suit adapte la démonstration du théorème de complétude de Gödel à la logique trivaluée.

Une version de ce théorème (théorème de complétude en logique bivaluée en calcul des prédicats) se trouve dans [45] et le formalisme utilisé est le même.

Nous utiliserons le formalisme de 2.2 et la notion d'interprétation.

Théorème 4.5.1 *On considère le système formel constitué des axiomes $B_1, B_2, B_3, B_4, A_1, A_2, ..., A_{22}$ suivants:*

$B_1 : A \to (B \to A)$;

$B_2 : (A \to (B \to C)) \to ((A \to B) \to (A \to C))$;

$B_3 : \forall x\, (A \to B) \to (A \to \forall x B)$ *si A ne contient pas d'occurrence libre de x;*

$B_4 : (\forall x_i\, A(x_i)) \to A(t)$ *(si t est libre pour x_i dans A, i.e il n'y a pas de x_i dans la portée d'un quantificateur de x_j dans A, x_j variable libre de t).*

A_1: $(\neg A \to C) \to ((\neg A \to \neg C) \to ((Ind_1(A) \to D) \to$
$((Ind_1(A) \to \neg D) \to A)))$;

A_2: $(\neg A \to C) \to ((\neg A \to \neg C) \to ((Ind_1(A) \to Ind_1(D)) \to$
$((Ind_1(A) \to D) \to A)))$;

A_3: $(\neg A \to C) \to ((\neg A \to \neg C) \to ((Ind_1(A) \to Ind_1(D)) \to$
$((Ind_1(A) \to \neg D) \to A)))$;

A_4: $(\neg A \to C) \to ((\neg A \to Ind_1(C)) \to ((Ind_1(A) \to D) \to$
$((Ind_1(A) \to \neg D) \to A)))$;

A_5: $(\neg A \to C) \to ((\neg A \to Ind_1(C)) \to ((Ind_1(A) \to \neg D) \to$
$((Ind_1(A) \to Ind_1(D)) \to A)))$;

A_6: $(\neg A \rightarrow C) \rightarrow ((\neg A \rightarrow Ind_1(C)) \rightarrow ((Ind_1(A) \rightarrow D) \rightarrow$
$((Ind_1(A) \rightarrow Ind_1(D)) \rightarrow A)))$;

A_7: $(\neg A \rightarrow \neg C) \rightarrow ((\neg A \rightarrow Ind_1(C)) \rightarrow ((Ind_1(A) \rightarrow D) \rightarrow$
$((Ind_1(A) \rightarrow \neg D) \rightarrow A)))$;

A_8: $(\neg A \rightarrow \neg C) \rightarrow ((\neg A \rightarrow Ind_1(C)) \rightarrow ((Ind_1(A) \rightarrow D) \rightarrow$
$((Ind_1(A) \rightarrow Ind_1(D)) \rightarrow A)))$;

A_9: $(\neg A \rightarrow \neg C) \rightarrow ((\neg A \rightarrow Ind_1(C)) \rightarrow ((Ind_1(A) \rightarrow \neg D) \rightarrow$
$((Ind_1(A) \rightarrow Ind_1(D)) \rightarrow A)))$;

A_{10}: $\forall y\, (A(y) \rightarrow C) \rightarrow (\exists y\, A(y) \rightarrow C)$; *y non variable libre de C;*

A_{11}: $(C \rightarrow \exists y\, A(y)) \rightarrow \exists y\, (C \rightarrow A(y))$; *y non variable libre de C;*

A_{12}: $\exists y\, (C \rightarrow A(y)) \rightarrow (C \rightarrow \exists y\, A(y))$; *y non variable libre de C;*

A_{13}: $\exists x\, A(x) \rightarrow \exists y\, A(y)$; *y n'apparaît pas dans A(x);*

A_{14}: $Ind_1(\exists x\, A(x)) \rightarrow \exists y\, Ind_1(A(y))$; *y n'apparaît pas dans A(x);*

A_{15}: $A \rightarrow \neg\neg A$;

A'_{15}: $\neg\neg A \rightarrow A$;

A_{16}: $Ind_1(A) \rightarrow Ind_1(\neg A)$;

A'_{16}: $Ind_1(\neg A) \rightarrow Ind_1(A)$;

A_{17}: $\neg(A \rightarrow C) \rightarrow A$;

A_{18}: $A \rightarrow (\neg C \rightarrow \neg(A \rightarrow C))$;

A_{19}: $Ind_1(C) \rightarrow (A \rightarrow \neg(A \rightarrow C))$;

A_{20}: $B(t) \rightarrow \exists x_i\, B(x_i)$ *si t est libre pour x_i dans B;*

A_{21}: $Ind_1(B \rightarrow C) \rightarrow A$;

A_{22}: $\neg\exists x\, A(x) \rightarrow \forall x\, \neg A(x)$;

et des deux règles d'inférence suivantes:

A $\vdash \forall x_i\, A$ *(généralisation universelle: u.g.);*
Modus ponens (m.p.);

On a alors, si A est une formule ayant pour seuls connecteurs $\neg$, $\rightarrow$ et le quantificateur existentiel $\exists$, l'équivalence suivante:

$$\vdash A \Leftrightarrow \models_T A$$

A n'est pas nécessairement une formule close.

N.B: $\forall$x A est la formule: $\neg(\exists x\, \neg A)$ et $Ind_1(A)$ est définie en 4.2.2.

Démonstration On définit quelques notions et on démontre des résultats intermédiaires.

Définition 4.5.1 Une théorie logique $\mathcal{T}$ est un ensembles de formules du language L(*) clos par déduction:

$$A \in \mathcal{T} \text{ implique } \forall x\, A \in \mathcal{T};$$
$$A,\ A \to B \in \mathcal{T} \text{ implique } B \in \mathcal{T};$$

Si Γ est un ensemble de formules de L(*), on note $D(\Gamma)$ le plus petit ensemble clos par déduction contenant Γ.

Définition 4.5.2 Une théorie **du premier ordre** est une théorie logique qui contient les formules universellement valides du langage(les tautologies).

Définition 4.5.3 Une théorie logique $\mathcal{T}$ est **consistante** s'il n'existe pas de formule A telle que:

$A \in \mathcal{T}$ et $Ind_1(A) \in \mathcal{T}$, ou $A \in \mathcal{T}$ et $\neg A \in \mathcal{T}$, ou $\neg A \in \mathcal{T}$ et $Ind_1(A) \in \mathcal{T}$.

Définition 4.5.4 Une théorie logique $\mathcal{T}$ est **complète** si, pour toute formule <u>close</u> A du langage L(*), on a:

soit $A \in \mathcal{T}$ ou $\neg A \in \mathcal{T}$ ou $Ind_1(A) \in \mathcal{T}$.

Définition 4.5.5 On note V(*) la théorie logique $D(\mathcal{A})$ où $\mathcal{A}$ est l'ensembles des axiomes du théorème 4.5.1. On a $V(*) = \{A \in For(L(*)) / \vdash A\}$.

Une théorie V(*)-basée $\mathcal{T}$ est une théorie $D(\mathcal{A} \cup \Gamma) = D(V(*) \cup \Gamma)$, dont l'ensemble des axiomes contiens $\mathcal{A}$.

Si $\{A_1, A_2, ..., A_n\}$ est un ensemble de formules de L(*), on note aussi:
$\{A_1, A_2, ..., A_n\} \cup \Gamma \vdash B$ par $A_1, A_2, ..., A_n \vdash_{\mathcal{T}} B$.
On a $\mathcal{T} = \{A \in For(L(*)) / \vdash_{\mathcal{T}} A\}$.

Théorème 4.5.2 *Soit $\mathcal{U}$ une interprétation sur L(*). Soit Γ l'ensemble de toutes les formules vraies dans $\mathcal{U}$ (pas nécessairement closes). Alors Γ est une théorie du premier ordre complète et consistante.*

Démonstration Γ contient les tautologies vraies à fortiori dans $\mathfrak{U}$.

D'autre part Γ est une théorie. Si A et $A \rightarrow B \in \Gamma$, $\forall\phi$, $vv_\phi(A) = V$ et $\forall\phi$, $vv_\phi(A \rightarrow B) = V$ donc $\forall\phi$, $vv_\phi(B) = V$ d'après la table de vérité du connecteur d'implication, et $B \in \Gamma$. Si $A \in \Gamma$, $\forall\phi$, $vv_\phi(A) = V$ donc $\forall\phi$, $vv_\phi(\forall x\, A) = V$ et $\forall x\, A \in \Gamma$.

Γ est complète: la valeur de vérité d'une formule close ne dépend pas de l'assignation; si $vv_{\mathfrak{U}}(A) = V$, $A \in \Gamma$, si $vv_{\mathfrak{U}}(A) = I$, $Ind_1(A) \in \Gamma$ et si $vv_{\mathfrak{U}}(A) = F$, $\neg A \in \Gamma$.

Γ est consistante: Si $Ind_1(A) \in \Gamma$, alors $\forall\phi$, $vv_\phi(Ind_1(A)) = V$.et donc $\forall\phi$, $vv_\phi(A) = I$, par conséquent ni A, ni $\neg A$ n'appartiennent à Γ.
De même si $A \in \Gamma$, $\forall\phi$, $vv_\phi(A) = V$, par conséquent ni $\neg A$, ni $Ind_1(A)$ n'appartiennent à Γ.
Enfin si $\neg A \in \Gamma$, $\forall\phi$, $vv_\phi(A) = F$ et par conséquent ni A, ni $Ind_1(A)$ n'appartiennent à Γ. ♦

Théorème 4.5.3 *Si un ensemble $\mathfrak{F}$ de formules clos par déduction a un modèle $\mathfrak{U}$, alors $\mathfrak{F}$ est une théorie consistante.*

Démonstration S'il existe $\mathfrak{U}$ tel que $\forall A \in \mathfrak{F}$, $\forall\phi$, $vv_\phi(A) = V$, alors $\neg A \notin \mathfrak{F}$, car sinon on aurait $\forall\phi$, $vv_\phi(A) = F$, et $Ind_1(A) \notin \mathfrak{F}$, car sinon on aurait $\forall\phi$, $vv_\phi(A) = I$.
De même pour les autres cas. ♦

Corollaire 4.5.1 *Si une théorie du premier ordre a un modèle, elle est consistante.*

Théorème 4.5.4 *V(*) est consistante.*

Démonstration Soit θ: $F(L(*)) \rightarrow F(P_0)$.

F(L(*)) désigne les formules de For(L(*)) ayant pour seuls connecteurs $\neg$, $\rightarrow$ et le quantificateur existentiel $\exists$.

$F(P_0)$ désigne les formules du calcul des prédicats sans quantificateurs, ayant pour seuls connecteurs $\neg$, $\rightarrow$.

On définit θ inductivement de la manière suivante:

- Si A est atomique, $A = p(t_1, t_2, ..., t_n)$, $\theta(A) = p(t_1, t_2, ..., t_n)$.

- Si $A = \neg B$, $\theta(\neg B) = \neg\theta(B)$;

- Si $A = A \rightarrow C$, $\theta(B \rightarrow C) = \theta(B) \rightarrow \theta(C)$;

-Si $A = \exists x\, F$, $\theta(\exists x\, F) = \theta(F)$;

On montre par induction sur la longueur de la preuve dans $\vdash A$ que:

$\vdash A$ implique $\theta(A)$ est une tautologie.

On vérifie pour chaque axiome Ax du système formel défini dans le théorème 4.5.1 que $\theta(Ax)$ est une tautologie.

Si A a été obtenue par m.p. à partir de C et $C \rightarrow A$, $\theta(C)$ et $\theta(C \rightarrow A)$ sont des tautologies par induction; or $\theta(C \rightarrow A) = \theta(C) \rightarrow \theta(A)$ et par conséquent $\theta(A)$ est une tautologie d'après la table du connecteur $\rightarrow$.

Si A a été obtenue par u.g. à partir de B, $A = \forall x\, B$, $\theta(\neg\exists x\, \neg B) = \neg\neg\theta(B)$ et comme $\theta(B)$ est une tautologie par induction, $\theta(A)$ est une tautologie.

On a d'autre part $\theta(\neg A) = \neg\theta(A)$ et $\theta(Ind_1(A)) = Ind_1(\theta(A))$; on ne peut donc avoir:

$\vdash A$ et $\vdash Ind_1(A)$ ce qui impliquerait $vv(\theta(A)) = V$ et $vv(\theta(A)) = I$, ni
$\vdash A$ et $\vdash \neg A$ ce qui impliquerait $vv(\theta(A)) = V$ et $vv(\theta(A)) = F$, ni
$\vdash \neg A$ et $\vdash Ind_1(A)$ ce qui impliquerait $vv(\theta(A)) = F$ et $vv(\theta(A)) = I$.

Donc V(*) est consistante. ♦

Théorème 4.5.5 *Si $\vdash A$ alors $A \in \mathcal{V}(*)$.*
(A est une tautologie: cf. définition 2.2.6)

Démonstration Par induction sur la longueur de la preuve dans $\vdash A$, en montrant d'abord que tous les axiomes sont des tautologies.

Les deux premiers axiomes ont été vus dans les théorèmes de complétude précédents.

D'après le lemme 2.2.5, $vv(\forall A) = V \Leftrightarrow \forall\phi,\ vv_\phi(A) = V$.

$\forall\phi,\ vv_\phi(B_3) = V \Leftrightarrow$

$\forall\phi,\ vv_\phi(\forall x\, (A \rightarrow B) \rightarrow (A \rightarrow \forall x\, B)\ ) = V \Leftrightarrow$

$\forall\phi,\ (\forall a,\ vv_{\phi(x \leftarrow a)}(A \rightarrow B) = V$ implique $vv_\phi(A \rightarrow \forall x\, B) = V)$;

ce qui est vrai puisque $vv_{\phi}(A) = vv_{\phi(x\leftarrow a)}(A)$ parce que x n'est pas variable libre de A.

$B_4 : (\forall x_i\, A(x_i)) \rightarrow A(t)$ (si t est libre pour x_i dans A, i.e il n'y a pas de x_i dans la portée d'un quantificateur de x_j dans A , x_j variable libre de t). On vérifie que:

$vv_{\phi}(\forall x_i\, A(x_i)) = V$ implique que $vv_{\phi}(A(t)) = V$;

Supposons que $\forall a$, $vv_{\phi(x_i\leftarrow a)}(A(x_i)) = V$; soit $a = \phi(t)$, alors par le lemme 2.2.6 on a: $vv_{\phi}(A(t)) = vv_{\phi(x_i\leftarrow a)}(A(x_i)) = V$ car $\phi_{(x_i\leftarrow a)}(x_i) = \phi(t)$ et $\phi_{(x_i\leftarrow a)}$ et ϕ coïncident sur toutes les variables autres que x_i.

Donc $vv_{\phi}(\forall x_i\, A(x_i)) = V$ implique $\forall a$, $vv_{\phi(x_i\leftarrow a)}(A(x_i)) = V$ et $vv_{\phi}(A(t)) = V$ en prenant $a = \phi(t)$.

$$A_1: (\neg A \rightarrow C) \rightarrow ((\neg A \rightarrow \neg C) \rightarrow ((Ind_1(A) \rightarrow D) \rightarrow ((Ind_1(A) \rightarrow \neg D) \rightarrow A)));$$

On a soit:
- $vv_{\phi}(A) = V$ auquel cas $vv_{\phi}(A_1) = V$;
- $vv_{\phi}(A) = I$ et soit:
  - $vv_{\phi}(D) = F$ ou I auquel cas $vv_{\phi}(Ind_1(A) \rightarrow D) = F$ et $vv_{\phi}(A_1) = V$;
  - $vv_{\phi}(D) = V$ et $vv_{\phi}(Ind_1(A) \rightarrow \neg D) = F$ et $vv_{\phi}(A_1) = V$;
- $vv_{\phi}(A) = F$ et soit:
  - $vv_{\phi}(C) = F$ ou I auquel cas $vv_{\phi}(\neg A \rightarrow C) = F$ et $vv_{\phi}(A_1) = V$;
  - $vv_{\phi}(C) = V$ et $vv_{\phi}(\neg A \rightarrow \neg C) = F$ et $vv_{\phi}(A_1) = V$;

On vérifie de la même façon que tous les autres A_i, pour $i = 1, ..., 9$, sont des tautologies.

A_{10}: $\forall y\, (A(y) \rightarrow C) \rightarrow (\exists y\, A(y) \rightarrow C)$; y non variable libre de C;

Supposons que $vv_{\phi}(\forall y\, (A(y) \rightarrow C)) = V$ et montrons que:
$vv_{\phi}(\exists y\, A(y) \rightarrow C) = V$.

$\forall a$, $vv_{\phi(y\leftarrow a)}(A(y) \rightarrow C) = V$. Si $vv_{\phi}(\exists y\, A(y)) = V$, alors $\exists a$, $vv_{\phi(y\leftarrow a)}(A(y)) = V$ et $vv_{\phi(y\leftarrow a)}(C) = V = vv_{\phi}(C)$ car y n'est pas variable libre de C (lemme 2.2.3 car $\phi_{(y\leftarrow a)}$ et ϕ coïncident sur les variables libres deC). Donc $vv_{\phi}(\exists y\, A(y) \rightarrow C) = V$.

A_{11}: $(C \rightarrow \exists y\, A(y)) \rightarrow \exists y\, (C \rightarrow A(y))$; y non variable libre de C;

Supposons que $vv_\phi(C \rightarrow \exists y\, A(y)) = V$ et montrons que $vv_\phi(\exists y\, (C \rightarrow A(y))) = V$.

Si $vv_\phi(C) = F$ ou I, alors $\forall a$, $vv_{\phi(y\leftarrow a)}(C \rightarrow A(y)) = V$ car $vv_{\phi(y\leftarrow a)}(C) = vv_\phi(C)$ (puisque y n'est pas variable libre de C).

Si $vv_\phi(C) = V$, alors $vv_\phi(\exists y\, A(y)) = V$ et $\exists a$, $vv_{\phi(y\leftarrow a)}(A(y)) = V$ et comme $vv_{\phi(y\leftarrow a)}(C) = V = vv_\phi(C)$ car y n'est pas variable libre de C, $\exists a$, $vv_{\phi(y\leftarrow a)}(C \rightarrow A(y)) = V$ donc $vv_\phi(\exists y\, (C \rightarrow A(y))) = V$.

De même pour A_{12}.

A_{13}: $\exists x\, A(x) \rightarrow \exists y\, A(y)$;

Supposons que $vv_\phi(\exists x\, A(x)) = V$ et montrons que $vv_\phi(\exists y\, A(y)) = V$. C'est-à-dire que si $\exists a$, $vv_{\phi(x\leftarrow a)}(A(x)) = V$ alors $\exists a'$, $vv_{\phi(y\leftarrow a')}(A(y)) = V$.

On a $vv_{\phi(x\leftarrow a)}(A(x)) = vv_{\phi(x\leftarrow a)(y\leftarrow a)}(A(x))$ d'après le lemme 2.2.3 car y n'apparaît pas dans A(x).

Or $\phi(x\leftarrow a)(y\leftarrow a)(x) = \phi(y\leftarrow a)(y)$ et $\phi(x\leftarrow a)(y\leftarrow a)$ et $\phi(y\leftarrow a)$ coïncident sur toutes les variables autres que x et par le lemme 2.2.6, on a:

$$vv_{\phi(x\leftarrow a)(y\leftarrow a)}(A(x)) = vv_{\phi(y\leftarrow a)}(A(y)) = V.$$

A_{14}: $Ind_1(\exists x\, A(x)) \rightarrow \exists y\, Ind_1(A(y))$;

Supposons que $vv_\phi(Ind_1(\exists x\, A(x))) = V$; alors $vv_\phi(\exists x\, A(x)) = I$ et donc $\forall a$, $vv_{\phi(x\leftarrow a)}(A(x)) = F$ ou I et $\exists d$, $vv_{\phi(x\leftarrow d)}(A(x)) = I$.
Montrons qu'alors $\forall a$, $vv_{\phi(y\leftarrow a)}(A(y)) = F$ ou I et $\exists d'$, $vv_{\phi(y\leftarrow d')}(A(y)) = I$.

On a $vv_{\phi(x\leftarrow a)}(A(x)) = vv_{\phi(x\leftarrow a)(y\leftarrow a)}(A(x))$ d'après le lemme 2.2.3 car y n'apparaît pas dans A(x).
Or $\forall a$, $\phi(x\leftarrow a)(y\leftarrow a)(x) = \phi(y\leftarrow a)(y)$ et $\phi(x\leftarrow a)(y\leftarrow a)$ et $\phi(y\leftarrow a)$ coïncident sur toutes les variables autres que x donc par le lemme 2.2.6, on a:

$vv_{\phi(x\leftarrow a)(y\leftarrow a)}(A(x)) = vv_{\phi(y\leftarrow a)}(A(y))$.

Donc $\forall a$, $vv_{\phi(y\leftarrow a)}(A(y)) = F$ ou I et $vv_{\phi(y\leftarrow d)}(A(y)) = I$.

Le fait que:

A_{15}: $A \to \neg\neg A$;

A'_{15}: $\neg\neg A \to A$;

A_{16}: $Ind_1(A) \to Ind_1(\neg A)$;

A'_{16}: $Ind_1(\neg A) \to Ind_1(A)$;

A_{17}: $\neg(A \to C) \to A$;

A_{18}: $A \to (\neg C \to \neg(A \to C))$;

A_{19}: $Ind_1(C) \to (A \to \neg(A \to C))$;

A_{21}: $Ind_1(B \to C) \to A$;

soient des tautologies est facile à vérifier.

A_{20}: $A(t) \to \exists x\, A(x)$ si t est libre pour x dans A;

si $vv_\phi(A(t)) = V$, soit $a = \phi(t)$; $\phi(x\leftarrow\phi(t))(x) = \phi(t)$ et $\phi(x\leftarrow\phi(t))$ et ϕ coïncident sur toutes les variables autres que x et par le lemme 2.2.6, on a:

$vv_{\phi(x\leftarrow\phi(t))}(A(x)) = vv_\phi(A(t))$.

Donc $vv_\phi(A(t)) = V$ implique $\exists a\; vv_{\phi(x\leftarrow a)}(A(x)) = V$, en prenant $a = \phi(t)$.

A_{22}: $\neg\exists x\, A(x) \to \forall x\, \neg A(x)$;

si ϕ est une valuation telle que $vv_\phi(\neg\exists x\, A(x)) = V$, alors $vv_\phi(\exists x\, A(x)) = F$ et $\forall d$, $vv_{(x\leftarrow d)}(A(x)) = F$ et donc $\forall d$, $vv_{(x\leftarrow d)}(\neg A(x)) = V$ et donc $vv_\phi(\forall x\, \neg A(x)) = V$

On suppose maintenant que A ait été obtenu par m.p. à partir de $C \to A$ et C; par induction $vv_\phi(C \to A) = V$ et $vv_\phi(C) = V$, donc $vv_\phi(A) = V$.

Si A a été obtenu par u.g., $A = \forall x\, B$, alors par induction B est une tautologie et $\forall\phi$, $vv_\phi(B) = V$ donc $vv_\phi(A) = V$. ♦

On a alors le théorème de déduction:

Théorème 4.5.6 *Si $\mathcal{T}$ est une théorie V(*)-basée, $A_1, A_2, \ldots, A_n$ sont des formules* <u>*closes*</u>*, et B une formule quelconque du langage L(*), alors:*

$$A_1, A_2, \ldots, A_n \vdash_{\mathcal{T}} B,$$

est équivalent à:

$$A_1, A_2, \ldots, A_{n-1} \vdash_{\mathcal{T}} A_n \to B.$$

Démonstration Le seul sens non trivial est $A_1, A_2, \ldots, A_n \vdash_{\mathcal{T}} B$ implique $A_1, A_2, \ldots, A_{n-1} \vdash_{\mathcal{T}} A_n \to B$, l'autre sens étant une application du modus ponens.

On le démontre par induction sur la longueur de la preuve dans $A_1, A_2, \ldots, A_n \vdash_{\mathcal{T}} B$.

Si B est obtenu en une étape à partir de $A_1, A_2, \ldots, A_n$, soit B est un axiome,
soit $B = A_i$, $i \neq n$, soit $B = A_n$.

Si B est un axiome, par B_1, on a: $B \to (A_n \to B)$ et avec B (axiome) et le m.p., on en déduit: $A_n \to B$.

Si $B = A_i$, $i \neq n$, on a $A_1, A_2, \ldots, A_{n-1} \vdash_{\mathcal{T}} B$, par B_1, on a: $B \to (A_n \to B)$ et avec le m.p., comme $B = A_i$, $i \neq n$, on en déduit: $A_1, A_2, \ldots, A_{n-1} \vdash_{\mathcal{T}} A_n \to B$.

Si $B = A_n$, on a: $\vdash A_n \to A_n$. En effet, par B_2, on a:

$$\vdash (A \to (B \to A)) \to ((A \to B) \to (A \to A)).$$

Par B_1 et le m.p.,on en déduit $\vdash (A \to B) \to (A \to A)$. En prenant $B = A \to A$, et réappliquant B_1 et le m.p., on en déduit $\vdash (A \to A)$.

Supposons que B ait été obtenu en k étapes à partir de $A_1, A_2, \ldots, A_n$.

Soit par le m.p., $A_1, A_2, \ldots, A_n \vdash_{\mathcal{T}} C \to B$ et $A_1, A_2, \ldots, A_n \vdash_{\mathcal{T}} C$. Alors par induction, on a: $A_1, A_2, \ldots, A_{n-1} \vdash_{\mathcal{T}} A_n \to C$ et

$A_1, A_2, ..., A_{n-1} \vdash_{\mathcal{T}} A_n \rightarrow (C \rightarrow B)$.
En appliquant B_2, on a:

$$\vdash (A_n \rightarrow (C \rightarrow B)) \rightarrow ((A_n \rightarrow C) \rightarrow (A_n \rightarrow B)).$$

Par le m.p., on a $A_1, A_2, ..., A_{n-1} \vdash_{\mathcal{T}} A_n \rightarrow B$.

Soit par u.g., $A_1, A_2, ..., A_n \vdash_{\mathcal{T}} C$ et $A_1, A_2, ..., A_n \vdash_{\mathcal{T}} \forall x\, C$ avec $B = \forall x\, C$.

On a $A_1, A_2, ..., A_{n-1} \vdash_{\mathcal{T}} A_n \rightarrow C$ par induction. Donc par u.g., $A_1, A_2, ..., A_{n-1} \vdash_{\mathcal{T}} \forall x\, (A_n \rightarrow C)$.

Comme A_n est close, elle ne contient pas d'occurrence de la variable x et on peut appliquer B_3, $\forall x\, (A_n \rightarrow C) \rightarrow (A_n \rightarrow \forall x\, C)$ et on en déduit le résultat:

$A_1, A_2, ..., A_{n-1} \vdash_{\mathcal{T}} A_n \rightarrow \forall x\, C$ par m.p. ♦

Théorème 4.5.7 *Si $\mathcal{T}$ est une théorie V(*)-basée consistante (ce qui assure l'existence d'une formule close telle que non $\vdash_{\mathcal{T}} A$), alors:*

si A est une formule close telle que non $\vdash_{\mathcal{T}} A$,

$\mathcal{T} \cup \{\neg A\}$ ou $\mathcal{T} \cup \{Ind_1(A)\}$ est une théorie V(*)-basée consistante.

Démonstration On suppose que $\mathcal{T} \cup \{\neg A\}$ et $\mathcal{T} \cup \{Ind_1(A)\}$ sont inconsistantes. Il y a 9 cas à vérifier qui sont éliminés par les axiomes A_1, A_2, ..., A_9.

Si $\mathcal{T} \cup \{\neg A\}$ et $\mathcal{T} \cup \{Ind_1(A)\}$ sont inconsistantes, il existe alors C telle que:

$\vdash_{\mathcal{T} \cup \{\neg A\}} C$ et $\vdash_{\mathcal{T} \cup \{\neg A\}} \neg C$ (1) ou:

$\vdash_{\mathcal{T} \cup \{\neg A\}} C$ et $\vdash_{\mathcal{T} \cup \{\neg A\}} Ind_1(C)$ (2) ou:

$\vdash_{\mathcal{T} \cup \{\neg A\}} \neg C$ et $\vdash_{\mathcal{T} \cup \{\neg A\}} Ind_1(C)$ (3).

De plus, il existe D telle que:

$\vdash_{\mathcal{T} \cup \{Ind_1(A)\}} D$ et $\vdash_{\mathcal{T} \cup \{\neg A\}} \neg D$ (1') ou:

$\vdash_{\mathcal{T} \cup \{Ind_1(A)\}} D$ et $\vdash_{\mathcal{T} \cup \{\neg A\}} Ind_1(D)$ (2') ou:

$\vdash_{\mathcal{T} \cup \{Ind_1(A)\}} \neg D$ et $\vdash_{\mathcal{T} \cup \{\neg A\}} Ind_1(D)$ (3').

-Le cas (1), (1') est éliminé par:
A_1: $(\neg A \to C) \to ((\neg A \to \neg C) \to ((Ind_1(A) \to D) \to ((Ind_1(A) \to \neg D) \to A)));$

On aurait alors avec le théorème de déduction (qui s'applique car A est close):

$\vdash_{\mathcal{T}} \neg A \to C$ et $\vdash_{\mathcal{T}} \neg A \to \neg C$ et $\vdash_{\mathcal{T}} Ind_1(A) \to D$ et $\vdash_{\mathcal{T}} Ind_1(A) \to \neg D$; on en déduirait donc: $\vdash_{\mathcal{T}} A$ par le m.p. et par A_1.

Le cas (1), (2') est éliminé par:

A_2: $(\neg A \to C) \to ((\neg A \to \neg C) \to ((Ind_1(A) \to Ind_1(D)) \to ((Ind_1(A) \to D) \to A)));$

On aurait alors avec le théorème de déduction: $\vdash_{\mathcal{T}} A$.

Le cas (1), (3') est éliminé par:

A_3: $(\neg A \to C) \to ((\neg A \to \neg C) \to ((Ind_1(A) \to Ind_1(D)) \to ((Ind_1(A) \to \neg D) \to A)));$

On aurait alors avec le théorème de déduction: $\vdash_{\mathcal{T}} A$.

Le cas (2), (1') est éliminé par:

A_4: $(\neg A \to C) \to ((\neg A \to Ind_1(C)) \to ((Ind_1(A) \to D) \to ((Ind_1(A) \to \neg D) \to A)));$

et le théorème de déduction.

Le cas (2), (3') est éliminé par:
A_5: $(\neg A \to C) \to ((\neg A \to Ind_1(C)) \to ((Ind_1(A) \to \neg D) \to ((Ind_1(A) \to Ind_1(D)) \to A)));$

et le théorème de déduction.

Le cas (2), (2') est éliminé par:

$$A_6: (\neg A \to C) \to ((\neg A \to \mathrm{Ind}_1(C)) \to ((\mathrm{Ind}_1(A) \to D) \\ \to ((\mathrm{Ind}_1(A) \to \mathrm{Ind}_1(D)) \to A)));$$

et le théorème de déduction.

Le cas (3), (1') est éliminé par:

$$A_7: (\neg A \to \neg C) \to ((\neg A \to \mathrm{Ind}_1(C)) \to \\ ((\mathrm{Ind}_1(A) \to D) \to ((\mathrm{Ind}_1(A) \to \neg D) \to A)));$$

et le théorème de déduction.

Le cas (3), (2') est éliminé par:

$$A_8: (\neg A \to \neg C) \to ((\neg A \to \mathrm{Ind}_1(C)) \to ((\mathrm{Ind}_1(A) \to D) \\ \to ((\mathrm{Ind}_1(A) \to \mathrm{Ind}_1(D)) \to A)));$$

et le théorème de déduction.

Le cas (3), (3') est éliminé par:

$$A_9: (\neg A \to \neg C) \to ((\neg A \to \mathrm{Ind}_1(C)) \to ((\mathrm{Ind}_1(A) \to \neg D) \\ \to ((\mathrm{Ind}_1(A) \to \mathrm{Ind}_1(D)) \to A)));$$

et le théorème de déduction. ♦

Nous avons montré que: $V(*) \subseteq \mathcal{V}(*)$. Nous allons maintenant prouver l'inclusion inverse:

$$\underline{\mathcal{V}(*) \subseteq V(*).}$$

Cela entraînera que toute théorie V(*)-basée est une théorie logique du premier ordre au sens où c'est un ensemble clos par déduction qui contient les tautologies.

On aura alors $V(*) = \mathcal{V}(*)$ et donc $\vdash A \Leftrightarrow\ \models_T A$.

On commence par démontrer que si A est une formule close, alors $\models_T A$ implique $\vdash A$, en montrant la contraposée.

C'est-à-dire que si A est une formule <u>close</u> non prouvable, A n'est pas une tautologie.

Le théorème 4.5.4 assure que V(*) est consistante et le théorème 4.5.7 s'applique:

Si A est une formule close non prouvable:

$V(*) \cup \{\neg A\}$ ou $V(*) \cup \{Ind_1(A)\}$ est une théorie V(*)-basée consistante.

Si on montre que $V(*) \cup \{\neg A\}$ ou $V(*) \cup \{Ind_1(A)\}$ a un modèle, A ne sera pas une tautologie.

En effet cela signifie que $\exists \mathcal{U}$ tel que $\forall \phi$, $vv_\phi(\neg A) = V$ ou $\exists \mathcal{U}$ tel que $\forall \phi$, $vv_\phi(Ind_1(A)) = V$. Par conséquent, $\exists \mathcal{U}$ tel que $\forall \phi$, $vv_\phi(A) = F$ ou $\exists \mathcal{U}$ tel que $\forall \phi$, $vv_\phi(A) = I$.

Nous allons donc montrer:

<u>$V(*) \cup \{B\}$ est consistante implique $V(*) \cup \{B\}$ a un modèle, où B est une formule de L(*).</u>

Le théorème principal est le suivant:

Théorème 4.5.8 *Toute théorie V(*)-basée consistante a un modèle.*

Dans toute la suite $\mathcal{T}$ désigne une théorie V(*)-basée consistante, $\mathcal{T} = D(V(*) \cup \Gamma)$ où Γ est un ensemble de formules de L(*).

On démontre deux résultats intermédiaires: les théorèmes 4.5.9 et 4.5.10.

Théorème 4.5.9 *Si $\mathcal{T}$ est une théorie V(*)-basée consistante ($\mathcal{T} = D(V(*) \cup \Gamma)$ où Γ est un ensemble de formules de L(*)) alors $\mathcal{T}$ a une extension complète et consistante $\mathcal{T}_E$.*

Démonstration On énumère les formules closes du langage L(*) par C_1, C_2, ..., C_n, ... et on définit inductivement T_i par:

$T_0 = \mathcal{T}$; si non $\vdash_{T_{i-1}} C_i$, alors $T_i = T_{i-1} \cup \{\neg C_i\}$ ou $T_i = T_{i-1} \cup \{Ind_1(C_i)\}$ en en choisissant une consistante par le théorème 4.5.7; autrement $T_i = T_{i-1}$.

Soit $\mathcal{T}_E = \cup_i T_i$.

Montrons que chaque T_i est consistante.

$T_0 = \mathcal{T}$ l'est par hypothèse. Si T_{i-1} est consistante, soit $T_i = T_{i-1}$ l'est, soit T_i l'est par construction.

$-\mathcal{T}_E$ est consistante d'après le lemme suivant car les $(T_i)_{i\in\mathcal{N}}$ forment une suite croissante d'extensions consistantes.

Lemme 4.5.1 *Si $\mathcal{T}$ est une théorie V(*)-basée consistante, $(T_i)_{i\in\mathcal{N}}$ une suite croissante d'extensions consistantes, $\mathcal{T}^* = \cup_i T_i$ est une extension consistante de $\mathcal{T}$.*

Démonstration Si $\vdash_{\mathcal{T}^*} A, \neg A, Ind_1(A)$ alors il existe n tel que $\vdash_{T_n} A, \neg A, Ind_1(A)$.

$-\mathcal{T}_E$ est complète car pour chaque formule close, on a soit $\vdash_{\mathcal{T}_E} C_i$, soit $\vdash_{\mathcal{T}_E} \neg C_i$, $\vdash_{\mathcal{T}_E} Ind_1(C_i)$ par construction. ♦

On ajoute maintenant des constantes $s_1, s_2, ..., s_n, ...$ au langage L(*), qui n'apparaissent pas dans L(*).

Soit L(*') le langage $L(*) \cup \{s_1, s_2, ..., s_n, ...\}$.

Soit V(*') la théorie dont les axiomes sont ceux de V(*) avec plus d'instanciations.

Soit $\mathcal{T}'$ la théorie $D(V(*') \cup \Gamma)$ déduite de $V(*') \cup \Gamma$ et $\mathcal{T}$ la théorie $D(V(*) \cup \Gamma)$ déduite de $V(*) \cup \Gamma$ où Γ est un ensemble de formules de L(*).

$\mathcal{T}'$ est consistante, puisque $\mathcal{T}$ l'est, selon le théorème suivant:

Théorème 4.5.10 *Si A est une formule ayant $x_{i_1}, ..., x_{i_n}$ pour variables libres, on a:*

$$\vdash_{D(V(*) \cup \Gamma)} A(x_{i_1}, ..., x_{i_n}) \Leftrightarrow \vdash_{D(V(*') \cup \Gamma)} A(s_{j_1}, ..., s_{j_n}),$$
$$\text{pour tout } \{s_{j_1}, s_{j_2}, ..., s_{j_n}\} \subseteq \{s_1, s_2, ..., s_n, ...\}.$$

Démonstration Soit $\mathcal{T}$ la théorie $D(V(*) \cup \Gamma)$ et $\mathcal{T}'$ la théorie $D(V(*') \cup \Gamma)$.

-Comme $\mathcal{T} \subseteq \mathcal{T}'$, $\vdash_{\mathcal{T}} A(x_{i_1}, ..., x_{i_n})$ implique $\vdash_{\mathcal{T}'} A(x_{i_1}, ..., x_{i_n})$.

Avec la règle u.g., $\vdash_{\mathcal{T}'} \forall x_{i_1} \forall x_{i_2} \ldots \forall x_{i_n} A(x_{i_1}, \ldots, x_{i_n})$;

par l'axiome B_4: $\forall x_i A(x_i) \rightarrow A(t)$ si t est libre pour x_i dans A, et le théorème de déduction, on a:

$\vdash \forall x_1 \forall x_2 \ldots \forall x_n A(x_1, \ldots, x_n) \rightarrow A(t_1, \ldots, t_n)$ si t_j est libre pour x_i dans A.

Ceci se démontre par récurrence sur n.
Si n = 1, c'est l'axiome B_4.
Sinon, on a en considérant la formule à 1 variable libre x_1,
$\forall x_2 \ldots \forall x_n A(x_1, \ldots, x_n)$,
$\vdash \forall x_1 \forall x_2 \ldots \forall x_n A(x_1, \ldots, x_n) \rightarrow \forall x_2 \ldots \forall x_n A(t_1, \ldots, x_n)$.
Donc, par le m.p., $\forall x_1 \forall x_2 \ldots \forall x_n A(x_1, \ldots, x_n) \vdash \forall x_2 \ldots \forall x_n A(t_1, \ldots, x_n)$.
Par induction, $\vdash \forall x_2 \ldots \forall x_n A(t_1, \ldots, x_n) \rightarrow A(t_1, \ldots, t_n)$ si t_j est libre pour x_i dans A ($i \geq 2$). Par le m.p., $\forall x_2 \ldots \forall x_n A(t_1, \ldots, x_n) \vdash A(t_1, \ldots, t_n)$.
Donc, $\forall x_1 \forall x_2 \ldots \forall x_n A(x_1, \ldots, x_n) \vdash A(t_1, \ldots, t_n)$ et comme $\forall x_1 \forall x_2 \ldots \forall x_n A(x_1, \ldots, x_n)$ est une formule close, le théorème de déduction s'applique et on a:

$\vdash \forall x_1 \forall x_2 \ldots \forall x_n A(x_1, \ldots, x_n) \rightarrow A(t_1, \ldots, t_n)$.

Donc:
$\vdash_{\mathcal{T}'} \forall x_{i_1} \forall x_{i_2} \ldots \forall x_{i_n} A(x_{i_1}, \ldots, x_{i_n}) \rightarrow A(s_{j_1}, \ldots, s_{j_n})$ car s_{j_p} est libre pour x_{i_p} dans A puisque c'est une constante.

D'où:

$\vdash_{\mathcal{T}'} A(s_{j_1}, \ldots, s_{j_n})$ en utilisant le modus ponens.

- Montrons que $\vdash_{\mathcal{T}'} A(s_{j_1}, \ldots, s_{j_n})$ implique $\vdash_{\mathcal{T}} A(x_{i_1}, \ldots, x_{i_n})$.

$\vdash_{\mathcal{T}'} A(s_{j_1}, \ldots, s_{j_n})$ signifie qu'il existe une suite de formules constituant une preuve de $A(s_{j_1}, \ldots, s_{j_n})$ dans $\mathcal{T}'$.

Soit $s_{j_1}, \ldots, s_{j_n}, \ldots, s_{j_t}$ la liste des constantes de $\{s_1, s_2, \ldots, s_n, \ldots\}$ qui apparaissent dans la preuve.

Soit $x_{m_1}, \ldots, x_{m_n}, \ldots, x_{m_t}, \ldots, x_{m_{t+n}}$ un ensemble de variables n'apparaissant pas dans la preuve et distinctes de $x_{i_1}, \ldots, x_{i_n}$.

Dans chaque formule de la preuve on remplace s_{j_i} par x_{m_i} pour $i = 1, ..., t$. Au même moment si $x_{i_j} \in \{x_{i_1}, x_{i_2}, ..., x_{i_n}\}$, on le remplace par $x_{m_{t+j}}$. Comme les axiomes de $\mathcal{T}'$ sont ceux de $\mathcal{T}$ avec plus d'instanciations, la suite de formules obtenue est une preuve dans $\mathcal{T}$.

On a donc $\vdash_{\mathcal{T}'} A(s_{j_1}, ..., s_{j_n})$ implique:

$\vdash_{\mathcal{T}} A(x_{m_1}, ..., x_{m_n})$.

Avec l'u.g., on en déduit $\vdash_{\mathcal{T}} \forall x_{m_1} \forall x_{m_2} ... \forall x_{m_n} A(x_{m_1}, ..., x_{m_n})$;

d'autre part on a:
$\vdash_{\mathcal{T}} \forall x_{m_1} \forall x_{m_2} ... \forall x_{m_n} A(x_{m_1}, ..., x_{m_n}) \to A(x_{i_1}, ..., x_{i_n})$
puisque x_{i_j} est libre pour x_{m_j} dans A car x_{m_j} n'apparaît pas dans la portée d'un quantificateur de x_{i_j} dans A puisque l'on a supprimé les x_{i_j} de la preuve.

Par m.p., on en déduit $\vdash_{\mathcal{T}} A(x_{i_1}, ..., x_{i_n})$. ♦

Corollaire 4.5.1 *$D(V(*) \cup \Gamma)$ est consistante si et ssi $D(V(*') \cup \Gamma)$ est consistante.*

On est maitenant en mesure de prouver le théorème 4.5.8.

On utilise les notations ci-dessus et on énumère les formules closes de F(L(*')) qui commencent par un quantificateur existentiel:

$$\exists x_{i_1} B_1(x_{i_1}), ..., \exists x_{i_j} B_j(x_{i_j}), ...$$

Il n'y a qu'un nombre fini de s_i qui apparaissent dans une formule B_j

Soit:

$$-H_j = \exists x_{i_j} B_j(x_{i_j}) \to B_j(s_{i_j})$$

où s_{i_j} est le premier s_k n'apparaissant ni dans B_j ni dans $\{H_1, ..., H_{j-1}\}$.

$$-G_j = Ind_1(\exists x_{i_j} B_j(x_{i_j})) \to Ind_1(B_j(s_{i_j})).$$

(s_{i_j} n'apparaît ni dans B_j ni dans $\{G_1, ..., G_{j-1}\}$ car les constantes de G_j sont les mêmes que celles de H_j).

On va utiliser ces H_j et G_j pour construire une suite infinie d'extensions de $\mathcal{T}'$ dont chaque élément est consistante.

On construira une extension complète et consistante $\mathcal{T}^{**}$ de $\mathcal{T}'$ qui contient H_j et G_j.

Pour cela on construit d'abord $\mathcal{T}'^{*}$ telle que:

$T'_1 = \mathcal{T}'$; $T'_n = T'_{n-1} \cup \{H_n\}$;
$\mathcal{T}'^{*} = \cup_n T'_n$.

Puis $\mathcal{T}^{*}$ telle que $T_1 = \mathcal{T}'^{*}$; $T_n = T_{n-1} \cup \{G_n\}$; $\mathcal{T}^{*} = \cup_n T_n$.

Si $\mathcal{T}^{*}$ est une théorie V(*)-basée consistante, elle admet une extension complète et consistante $\mathcal{T}^{**}$ par le théorème 4.5.9.

Montrons que $\mathcal{T}^{*}$ est une théorie V(*)-basée consistante.

Pour cela on commence par montrer que $\mathcal{T}'^{*}$ est une théorieV(*)-basée consistante.

Soit $T'_1 = \mathcal{T}'$; $T'_n = T'_{n-1} \cup \{H_n\}$; $\mathcal{T}'^{*} = \cup_n T'_n$.

Montrons que chaque T'_n est consistante.

T'_1 l'est par hypothèse.

Supposons que T'_i est consistante pour $i < k$ et montrons que T'_k l'est.

Si T'_k n'est pas consistante, il existe une formule A telle que:

Soit $\vdash_{T'_k} A$ et $\vdash_{T'_k} \neg A$ (1) ou:

$\vdash_{T'_k} A$ et $\vdash_{T'_k} \mathrm{Ind}_1(A)$ (2) ou:

$\vdash_{T'_k} \neg A$ et $\vdash_{T'_k} \mathrm{Ind}_1(A)$ (3).

Comme H_k est une formule close, on peut appliquer le théorème de déduction et on a:

Soit $\vdash_{T'_{k-1}} H_k \to A$ et $\vdash_{T'_{k-1}} H_k \to \neg A$ (1) ou:

$\vdash_{T'_{k-1}} H_k \rightarrow A$ et $\vdash_{T'_{k-1}} H_k \rightarrow \mathrm{Ind}_1(A)$ (2) ou:

$\vdash_{T'_{k-1}} H_k \rightarrow \neg A$ et $\vdash_{T'_{k-1}} H_k \rightarrow \mathrm{Ind}_1(A)$ (3).

On va montrer que si B est une formule close telle que:

$$\vdash_{T'_{k-1}} H_k \rightarrow B$$

alors on a:

$$\vdash_{T'_{k-1}} B$$

ce qui contredira le fait que T'_{k-1} est consistante.

Comme s_{i_k} est le premier s_j n'apparaissant ni dans B_k ni dans $\{H_1, ..., H_{k-1}\}$, on peut dans $\vdash_{T'_{k-1}} H_k \rightarrow B$ remplacer s_{i_k} par une variable y n'apparaissant pas dans la preuve.

On a alors: $\vdash_{T'_{k-1}} (\exists x_{i_k} B_k(x_{i_k}) \rightarrow B_k(y)) \rightarrow B$.

Par u.g., on a:

$\vdash_{T'_{k-1}} \forall y\ ((\exists x_{i_k} B_k(x_{i_k}) \rightarrow B_k(y)) \rightarrow B)$.

Comme y n'est pas variable libre de B qui est une formule close,

A_{10}: $\forall y\ (A(y) \rightarrow C) \rightarrow ((\exists y\ A(y)) \rightarrow C)$ s'applique et l'on en déduit:

$$\vdash \forall y\ ((\exists x_{i_k} B_k(x_{i_k}) \rightarrow B_k(y)) \rightarrow B) \rightarrow (\exists y\ (\exists x_{i_k} B_k(x_{i_k}) \rightarrow B_k(y)) \rightarrow B).$$

Donc par le m.p., $\vdash_{T'_{k-1}} \exists y\ (\exists x_{i_k} B_k(x_{i_k}) \rightarrow B_k(y)) \rightarrow B$.

D'après A_{11}: $(C \rightarrow (\exists y\ A(y))) \rightarrow (\exists y\ (C \rightarrow A(y)))$ si y non variable libre de C, comme y n'est pas variable libre de $\exists x_{i_k} B_k(x_{i_k})$, on a:

$\vdash (\exists x_{i_k} B_k(x_{i_k}) \rightarrow \exists y\ B_k(y)) \rightarrow (\exists y\ (\exists x_{i_k} B_k(x_{i_k}) \rightarrow B_k(y)))$.

Comme d'après A_{13}: $\exists x\ A(x) \rightarrow \exists y\ A(y)$ si y n'apparaît pas dans A(x), on a:

$\vdash \exists x_{i_k} B_k(x_{i_k}) \rightarrow \exists y\ B_k(y)$ car y n'apparaît pas dans $\exists x_{i_k} B_k(x_{i_k})$, par deux applications de m.p., on en déduit:

$\vdash_{T'_{k-1}} B$.

Chaque T'_k est donc consistante et $\mathcal{T}'^*$ l'est par le lemme 4.5.1.

Montrons maintenant que $\mathcal{T}^*$ est consistante.

Soit $T_1 = \mathcal{T}'^*$; $T_n = T_{n-1} \cup \{G_n\}$; $\mathcal{T}^* = \cup_n T_n$.

Montrons que chaque T_n est consistante.

T_1 l'est d'après ci-dessus.

Supposons que T_i est consistante pour $i < k$ et montrons que T_k l'est.
Si T_k n'est pas consistante, il existe une formule A telle que:

soit $\vdash_{T_k} A$ et $\vdash_{T_k} \neg A$ (1) ou:

$\vdash_{T_k} A$ et $\vdash_{T_k} \mathrm{Ind}_1(A)$ (2) ou:

$\vdash_{T_k} \neg A$ et $\vdash_{T_k} \mathrm{Ind}_1(A)$ (3).

Comme G_k est une formule close, on peut appliquer le théorème de déduction et on a:

Soit $\vdash_{T_{k-1}} G_k \rightarrow A$ et $\vdash_{T_{k-1}} G_k \rightarrow \neg A$ (1) ou:

$\vdash_{T_{k-1}} G_k \rightarrow A$ et $\vdash_{T_{k-1}} G_k \rightarrow \mathrm{Ind}_1(A)$ (2) ou:

$\vdash_{T_{k-1}} G_k \rightarrow \neg A$ et $\vdash_{T_{k-1}} G_k \rightarrow \mathrm{Ind}_1(A)$ (3).

On va montrer que si B est une formule close telle que:

$$\vdash_{T_{k-1}} G_k \rightarrow B$$

alors on a:

$$\vdash_{T_{k-1}} B$$

ce qui contredira le fait que T_{k-1} est consistante.

Comme s_{i_k} est le premier s_j n'apparaissant ni dans B_k ni dans $\{G_1, ..., G_{k-1}\}$, on peut dans $\vdash_{T_{k-1}} G_k \rightarrow B$ remplacer s_{i_k} par une variable y n'apparaissant pas dans la preuve.

On a alors: $\vdash_{T_{k-1}} (\mathrm{Ind}_1(\exists x_{i_k} B_k(x_{i_k})) \to \mathrm{Ind}_1(B_k(y))) \to B$.

Par u.g., on a:

$\vdash_{T_{k-1}} \forall y\, ((\mathrm{Ind}_1(\exists x_{i_k} B_k(x_{i_k})) \to \mathrm{Ind}_1(B_k(y))) \to B)$.

Comme y n'est pas variable libre de B qui est une formule close, A_{10}:

$\forall y\, (A(y) \to C) \to ((\exists y\, A(y)) \to C)$ s'applique et l'on en déduit:

$$\vdash \forall y\, ((\mathrm{Ind}_1(\exists x_{i_k} B_k(x_{i_k})) \to \mathrm{Ind}_1(B_k(y))) \to B) \to$$
$$(\exists y\, (\mathrm{Ind}_1(\exists x_{i_k} B_k(x_{i_k})) \to \mathrm{Ind}_1(B_k(y))) \to B).$$

Donc par le m.p., $\vdash_{T_{k-1}} \exists y\, (\mathrm{Ind}_1(\exists x_{i_k} B_k(x_{i_k})) \to \mathrm{Ind}_1(B_k(y)) \to B$.

D'après A_{11}: $(C \to (\exists y\, A(y))) \to (\exists y\, (C \to A(y)))$ si y non variable libre de C, comme y n'est pas variable libre de $\mathrm{Ind}_1(\exists x_{i_k} B_k(x_{i_k}))$, on a:

$$\vdash \mathrm{Ind}_1(\exists x_{i_k} B_k(x_{i_k})) \to \exists y\, \mathrm{Ind}_1(B_k(y)) \to$$
$$(\exists y\, (\mathrm{Ind}_1(\exists x_{i_k} B_k(x_{i_k})) \to \mathrm{Ind}_1(B_k(y)))).$$

Comme d'après A_{14}: $\mathrm{Ind}_1(\exists x\, A(x)) \to \exists y\, \mathrm{Ind}_1(A(y))$ si y n'apparaît pas dans A(x), on a: $\vdash \mathrm{Ind}_1(\exists x_{i_k} B_k(x_{i_k})) \to \exists y\, \mathrm{Ind}_1(B_k(y))$ car y n'apparaît pas dans $B_k(x_{i_k})$, par deux applications de m.p., on en déduit:

$\vdash_{T_{k-1}} B$.

Chaque T_k est donc consistante et $\mathcal{T}^*$ l'est par le lemme 4.5.1.

Si $\mathcal{T}^*$ est une théorie V(*)-basée consistante, elle admet une extension complète et consistante $\mathcal{T}^{**}$ par le théorème 4.5.9.

$\mathcal{T}^{**}$ contient les formules -$H_j = \exists x_{i_j} B_j(x_{i_j}) \to B_j(s_{i_j})$
-$G_j = \mathrm{Ind}_1(\exists x_{i_j} B_j(x_{i_j})) \to \mathrm{Ind}_1(B_j(s_{i_j}))$.

On construit un modèle de $\mathcal{T}^{**}$ qui sera un modèle de $\mathcal{T}$ puisque $\mathcal{T} \subseteq \mathcal{T}^{**}$.

On prend une interprétation de Herbrand $\mathcal{U}$ dans laquelle le domaine D est l'univers de tous les termes clos de L(*').

Si t est un terme clos, on note [t] son interprétation. Chaque constante est interprétée par elle-même; chaque symbole de fonction f d'arité n est interprété de manière canonique par:
f: $([h_1], [h_2], ..., [h_n]) \rightarrow [f(h_1, h_2, ..., h_n)]$.

Il reste à donner une valeur de vérité aux symboles de prédicats de L(*'); on associe à chaque symbole de prédicat p une fonction de D^n dans {V, F, I} de la manière suivante:

$vv_{\mathcal{U}}(p(h_1, h_2, ..., h_n))$ = Vrai ssi $\vdash_{\mathcal{T}^{**}} p(h_1, h_2, ..., h_n)$
$vv_{\mathcal{U}}(p(h_1, h_2, ..., h_n))$ = Faux ssi $\vdash_{\mathcal{T}^{**}} \neg p(h_1, h_2, ..., h_n)$
$vv_{\mathcal{U}}(p(h_1, h_2, ..., h_n))$ = Indéterminé sinon;

Ceci est parfaitement défini car $\mathcal{T}^{**}$ est consistante et définit bien une interprétation trivaluée.

Nous allons montrer par induction pour toute formule close A de F(*') que:

$vv_{\mathcal{U}}(A)$ = Vrai ssi $\vdash_{\mathcal{T}^{**}} A$
$vv_{\mathcal{U}}(A)$ = Faux ssi $\vdash_{\mathcal{T}^{**}} \neg A$
$vv_{\mathcal{U}}(A)$ = Indéterminé ssi $\vdash_{\mathcal{T}^{**}} Ind_1(A)$.

-<u>Si A est atomique</u>;

$A = p(h_1, h_2, ..., h_n)$; par définition:

$vv_{\mathcal{U}}(p(h_1, h_2, ..., h_n))$ = Vrai ssi $\vdash_{\mathcal{T}^{**}} p(h_1, h_2, ..., h_n)$ (1)
$vv_{\mathcal{U}}(p(h_1, h_2, ..., h_n))$ = Faux ssi $\vdash_{\mathcal{T}^{**}} \neg p(h_1, h_2, ..., h_n)$ (2)
$vv_{\mathcal{U}}(p(h_1, h_2, ..., h_n))$ = Indéterminé sinon; (3)

- Les cas $vv_{\mathcal{U}}(A) = V$ et $vv_{\mathcal{U}}(A) = F$ sont traités par (1) et (2).

- Si $vv_{\mathcal{U}}(p(h_1, h_2, ..., h_n)) = I$ alors on a:
non $\vdash_{\mathcal{T}^{**}} p(h_1, h_2, ..., h_n)$ et non $\vdash_{\mathcal{T}^{**}} \neg p(h_1, h_2, ..., h_n)$ d'après (1) et (2).
Comme $\mathcal{T}^{**}$ est complète, on a: $\vdash_{\mathcal{T}^{**}} Ind_1(A)$ puisque A est close.

- Réciproquement, si $\vdash_{\mathcal{T}^{**}} Ind_1(A)$ on a non $\vdash_{\mathcal{T}^{**}} A$ et non $\vdash_{\mathcal{T}^{**}} \neg A$ car $\mathcal{T}^{**}$ est consistante; donc $vv_{\mathcal{U}}(A) = I$ car la valeur de vérité d'une formule close est parfaitement déterminée.

- Soit $A = \neg B$;

- $vv_{\mathcal{U}}(A) = V$ ssi $vv_{\mathcal{U}}(B) = F$ ssi $\vdash_{\mathcal{T}^{**}} \neg B$ par induction ssi $\vdash_{\mathcal{T}^{**}} A$.

- $vv_{\mathcal{U}}(A) = F$ ssi $vv(B) = V$ ssi $\vdash_{\mathcal{T}^{**}} B$ par induction.
Avec les axiomes $B \rightarrow \neg\neg B$ et $\neg\neg B \rightarrow B$ (A_{15} et A'_{15}), on a:

$$\vdash_{\mathcal{T}^{**}} B \text{ ssi } \vdash_{\mathcal{T}^{**}} \neg\neg B \text{ ssi } \vdash_{\mathcal{T}^{**}} \neg A.$$

- $vv_{\mathcal{U}}(A) = I$ ssi $vv_{\mathcal{U}}(B) = I$ ssi $\vdash_{\mathcal{T}^{**}} Ind_1(B)$ par induction. Avec les axiomes $Ind_1(B) \rightarrow Ind_1(\neg B)$ et $Ind_1(\neg B) \rightarrow B$ (A_{16} et A'_{16}), on a:

$$\vdash_{\mathcal{T}^{**}} Ind_1(B) \text{ ssi } \vdash_{\mathcal{T}^{**}} Ind_1(\neg B) \text{ ssi } \vdash_{\mathcal{T}^{**}} Ind_1(A).$$

- Soit $A = B \rightarrow C$;

Montrons que $\vdash_{\mathcal{T}^{**}} B \rightarrow C$ ssi $vv_{\mathcal{U}}(B \rightarrow C) = V$;

comme $B \rightarrow C$ ne prend pas la valeur I, cela revient à montrer que:

non $\vdash_{\mathcal{T}^{**}} B \rightarrow C$ ssi $vv_{\mathcal{U}}(B \rightarrow C) = F$;

Or non $\vdash_{\mathcal{T}^{**}} B \rightarrow C$ équivaut à $\vdash_{\mathcal{T}^{**}} \neg(B \rightarrow C)$ ou $\vdash_{\mathcal{T}^{**}} Ind_1(B \rightarrow C)$ car la théorie est complète et A est close.

D'autre part la théorie est consistante et A_{21}: $Ind_1(B \rightarrow C) \rightarrow A$ prouve que l'on n'a jamais:

$\vdash_{\mathcal{T}^{**}} Ind_1(B \rightarrow C)$ (sinon on aurait $\vdash_{\mathcal{T}^{**}} A$ et $\vdash_{\mathcal{T}^{**}} \neg A$ par le m.p.).

On montre donc que $\vdash_{\mathcal{T}^{**}} \neg(B \rightarrow C)$ équivaut à $vv_{\mathcal{U}}(B \rightarrow C) = F$;

– $\vdash_{\mathcal{T}^{**}} \neg(B \rightarrow C)$ implique $\vdash_{\mathcal{T}^{**}} B$ par A_{17}: $\neg(B \rightarrow C) \rightarrow B$ et le m.p.. Donc $vv_{\mathcal{U}}(B) = V$ par induction.

Montrons que $vv_{\mathfrak{u}}(C) = F$ ou I et l'on aura: $vv_{\mathfrak{u}}(B \to C) = F$;

$vv_{\mathfrak{u}}(C) = F$ équivaut à $\vdash_{\mathcal{T}^{**}} \neg C$ et $vv_{\mathfrak{u}}(C) = I$ équivaut à $\vdash_{\mathcal{T}^{**}} Ind_1(C)$ par induction. Si non $\vdash_{\mathcal{T}^{**}} \neg C$, alors $\vdash_{\mathcal{T}^{**}} C$ ou $\vdash_{\mathcal{T}^{**}} Ind_1(C)$ car la théorie est complète et C est close.

Mais d'après B_1: $C \to (B \to C)$ et le m.p.,

$\vdash_{\mathcal{T}^{**}} C$ implique $\vdash_{\mathcal{T}^{**}} B \to C$ ce qui est impossible car la théorie est consistante.

Donc soit $vv_{\mathfrak{u}}(C) = F$ ou $vv_{\mathfrak{u}}(C) = I$.

- Réciproquement, si $vv_{\mathfrak{u}}(B \to C) = F$, alors $vv_{\mathfrak{u}}(B) = V$ et $vv_{\mathfrak{u}}(C) = F$ ou I; on a donc:

$\vdash_{\mathcal{T}^{**}} B$ et $\vdash_{\mathcal{T}^{**}} \neg C$ ou $\vdash_{\mathcal{T}^{**}} Ind_1(C)$ par induction.

Si $\vdash_{\mathcal{T}^{**}} B$ et $\vdash_{\mathcal{T}^{**}} \neg C$ alors $\vdash_{\mathcal{T}^{**}} \neg(B \to C)$ par A_{18}: $B \to (\neg C \to \neg(B \to C))$ et le m.p.;

Si $\vdash_{\mathcal{T}^{**}} B$ et $\vdash_{\mathcal{T}^{**}} Ind_1(C)$ alors $\vdash_{\mathcal{T}^{**}} \neg(B \to C)$ par A_{19}: $Ind_1(C) \to (B \to \neg(B \to C))$ et le m.p.;

Donc si $vv_{\mathfrak{u}}(B \to C) = F$, alors $\vdash_{\mathcal{T}^{**}} \neg(B \to C)$.

- Soit $A = \exists x_{i_k} B_k(x_{i_k})$ (c'est ainsi qu'on a énuméré les formules closes commençant par un quantificateur existenciel).

Montrons que: $vv_{\mathfrak{u}}(A) = \text{Vrai}$ ssi $\vdash_{\mathcal{T}^{**}} A$
$vv_{\mathfrak{u}}(A) = \text{Faux}$ ssi $\vdash_{\mathcal{T}^{**}} \neg A$
$vv_{\mathfrak{u}}(A) = \text{Indéterminé}$ ssi $\vdash_{\mathcal{T}^{**}} Ind_1(A)$.

- Si $\vdash_{\mathcal{T}^{**}} \exists x_{i_k} B_k(x_{i_k})$, alors $\vdash_{\mathcal{T}^{**}} B_k(s_{i_k})$ par le m.p. et parce que $\mathcal{T}^{**}$contient toutes les formules H_k. Donc $vv_{\mathfrak{u}}(B_k(s_{i_k})) = V$ par induction ($B_k(s_{i_k})$ est une formule close) et $vv_{\mathfrak{u}}(\exists x_{i_k} B_k(x_{i_k})) = V$.

En effet $\forall \phi\ \exists d_\phi\ vv_{\phi(x_{i_k} \leftarrow d_\phi)}(B_k(x_{i_k})) = V$. Il suffit de prendre $d_\phi = s_{i_k}$.

On a bien $vv_{\phi(x_{i_k} \leftarrow s_{i_k})}(B_k(x_{i_k})) = vv_\phi(B_k(s_{i_k})) = V$ car:

d'une part $\phi_{(x_{i_k} \leftarrow s_{i_k})}(x_{i_k}) = s_{i_k} = \phi(s_{i_k})$ car les termes clos de L(*') sont interprétés par eux-mêmes dans $\mathfrak{U}$. D'autre part $\phi_{(x_{i_k} \leftarrow s_{i_k})}$ et ϕ coïncident sur toutes les variables autres que x_{i_k} donc par le lemme 2.2.6, on a: $vv_{\phi(x_{i_k} \leftarrow s_{i_k})}(B_k(x_{i_k})) = vv_\phi(B_k(s_{i_k}))$.

Réciproquement, si $vv_{\mathfrak{U}}(\exists x_{i_k} B_k(x_{i_k})) = V$ alors $\forall \phi\ \exists d_\phi\ vv_{\phi(x_{i_k} \leftarrow d_\phi)}(B_k(x_{i_k})) = V$. Donc, $\forall \phi\ \exists d_\phi\ vv_\phi(B_k(d_\phi)) = V$ (lemme 2.2.6). En effet $\phi_{(x_{i_k} \leftarrow d_\phi)}(x_{i_k}) = d_\phi = \phi(d_\phi)$ et $\phi_{(x_{i_k} \leftarrow d_\phi)}$ et ϕ coïncident sur les autres variables que x_{i_k} donc $vv_{\phi(x_{i_k} \leftarrow d_\phi)}(B_k(x_{i_k})) = vv_\phi(B_k(d_\phi))$.
Mais $B_k(d_\phi)$ est une formule close et l'on a $vv_\phi(B_k(d_\phi)) = vv_{\phi'}(B_k(d_\phi))$ (lemme 2.2.2).
Donc, $\exists d, \forall \phi\ vv_\phi(B_k(d)) = V$ et $\exists d, vv_{\mathfrak{U}}(B_k(d)) = V$.

Donc $\vdash_{\mathbf{T}^{**}} B_k(d)$ par induction et comme par A_{20} on a:

$\vdash_{\mathbf{T}^{**}} B_k(d) \rightarrow \exists x_{i_k} B_k(x_{i_k})$, on en déduit $\vdash_{\mathbf{T}^{**}} \exists x_{i_k} B_k(x_{i_k})$ par le m.p..

- Si $\vdash_{\mathbf{T}^{**}} \neg(\exists x_{i_k} B_k(x_{i_k}))$, montrons que $vv_{\mathfrak{U}}(\exists x_{i_k} B_k(x_{i_k})) = F$ par l'absurde.

-Si $vv_{\mathfrak{U}}(\exists x_{i_k} B_k(x_{i_k})) = V$ alors $\exists d, \forall \phi\ vv_\phi(B_k(d)) = V$ (cf. ci-dessus).
Donc $\vdash_{\mathbf{T}^{**}} B_k(d)$ par induction et comme par A_{20} on a:

$\vdash_{\mathbf{T}^{**}} B_k(d) \rightarrow \exists x_{i_k} B_k(x_{i_k})$, car d est libre pour dans puisque c'est une constante.
On en déduit alors: $\vdash_{\mathbf{T}^{**}} \exists x_{i_k} B_k(x_{i_k})$.
Le cas $vv_{\mathfrak{U}}(\exists x_{i_k} B_k(x_{i_k})) = V$ est donc à éliminer.

par le m.p. ce qui est impossible car $\mathbf{T}^{**}$ est consistante.

-Si $vv_{\mathfrak{U}}(\exists x_{i_k} B_k(x_{i_k})) = I$ alors $\forall d\ vv(B_k(d)) \neq V$ et $\exists d$, $vv(B_k(d)) = I$.

Comme:

$\vdash_{\mathcal{T}^{**}} \neg(\exists x_{i_k} B_k(x_{i_k}))$, on a: $\vdash_{\mathcal{T}^{**}} \forall x_{i_k} \neg B_k(x_{i_k})$ par A_{22} et par B_4 on a pour tout terme clos d (alors libre pour x_{i_k} dans B_k), $\vdash_{\mathcal{T}^{**}}$ $\neg B_k(d)$.

Par induction, on a $vv_{\mathcal{U}}(B_k(d)) = F$, pour tout terme clos d et on ne peut donc avoir: $vv(\exists x_{i_k} B_k(x_{i_k})) = I$.

- Réciproquement, supposons que $vv_{\mathcal{U}}(\exists x_{i_k} B_k(x_{i_k})) = F$ et montrons que $\vdash_{\mathcal{T}^{**}} \neg(\exists x_{i_k} B_k(x_{i_k}))$.

Si non $\vdash_{\mathcal{T}^{**}} \neg(\exists x_{i_k} B_k(x_{i_k}))$ alors soit $\vdash_{\mathcal{T}^{**}} (\exists x_{i_k} B_k(x_{i_k}))$, soit $\vdash_{\mathcal{T}^{**}} Ind_1(\exists x_{i_k} B_k(x_{i_k}))$, car la théorie est complète et $\exists x_{i_k} B_k(x_{i_k})$ est close.

-Si $\vdash_{\mathcal{T}^{**}} (\exists x_{i_k} B_k(x_{i_k}))$, alors $\vdash_{\mathcal{T}^{**}} B_k(s_{i_k})$ car $\mathcal{T}^{**}$contient toutes les formules H_k. Donc $vv_{\mathcal{U}}(B_k(s_{i_k})) = V$ par induction ($B_k(s_{i_k})$ est une formule close) et $vv_{\mathcal{U}}(\exists x_{i_k} B_k(x_{i_k})) = V$ car $\exists d = s_{i_k}$ tel que:.$vv_{\phi(x_{i_k} \leftarrow d)}(B_k(x_{i_k})) = vv_{\phi}(B_k(s_{i_k})) = V$.

- Si $\vdash_{\mathcal{T}^{**}} Ind_1(\exists x_{i_k} B_k(x_{i_k}))$, alors $\vdash_{\mathcal{T}^{**}} Ind_1(B_k(s_{i_k}))$ car $\mathcal{T}^{**}$contient toutes les formules G_k. Donc $vv_{\mathcal{U}}(B_k(s_{i_k})) = I$ par induction ($B_k(s_{i_k})$ est une formule close) et $vv_{\mathcal{U}}(\exists x_{i_k} B_k(x_{i_k})) \neq F$.

- Le troisième cas: $\vdash_{\mathcal{T}^{**}} Ind_1(\exists x_{i_k} B_k(x_{i_k})) \Leftrightarrow vv_{\mathcal{U}}(\exists x_{i_k} B_k(x_{i_k})) = I$ est résolu par les deux autres, la théorie étant complète, consistante et la valeur de vérité d'une formule parfaitement déterminée.

L'interprétation $\mathcal{U}$ est alors un modèle de $\mathcal{T}^{**}$:

En effet si $A(x_1, x_2, ..., x_n)$ est une formule telle que $\vdash_{\mathcal{T}^{**}} A(x_1, x_2, ..., x_n)$;

alors par u.g., $\vdash_{\mathcal{T}^{**}} \forall x_1 \forall x_2 ... \forall x_n A(x_1, x_2, ..., x_n)$;

$\forall x_1 \forall x_2 ... \forall x_n A(x_1, x_2, ..., x_n)$ est une formule close, et on a:

$vv(\forall x_1 \forall x_2 ... \forall x_n A(x_1, x_2, ..., x_n)) = Vrai$;

donc $\forall\phi\ vv_\phi(A(x_1, x_2, ..., x_n)) =$ Vrai, ce qui signifie que $A(x_1, x_2, ..., x_n)$ est Vraie dans $\mathfrak{U}$ au sens où sa clôture universelle est Vraie dans $\mathfrak{U}$.

On a ainsi prouvé que $\mathfrak{T}$ qui est une théorie consistante V(*)-basée a un modèle,
puisque $\mathfrak{U}$ est un modèle de $\mathfrak{T}^{**}$ et $\mathfrak{T} \subseteq \mathfrak{T}^{**}$.

On a ainsi prouvé que $\models_T A$ implique $\vdash A$ pour toute formule close.

-Si A n'est pas close, alors $\models_T A$ implique $\models_T \forall A$ et donc $\vdash \forall A$
où $\forall A$ désigne la clôture universelle de A;

$\vdash \forall x_1 \forall x_2 ... \forall x_n A(x_1, x_2, ..., x_n)$ implique $\vdash A(y_1, y_2, ..., y_n)$ avec l'axiome B_4 (avec y_i n'apparaissant pas dans A) et le théorème de déduction.
$\vdash A(y_1, y_2, ..., y_n)$ implique $\vdash A(x_1, x_2, ..., x_n)$ en renommant les variables de la preuve.

Par conséquent, si A n'est pas close, alors $\models_T A$ implique $\vdash A$. ♦

Corollaire 4.5.2 *L'ensemble des tautologies du langage L, en calcul des prédicats, ayant pour seuls connecteurs $\neg$, $\rightarrow$, et pour quantificateur $\exists$ est récursivement énumérable.*

Par contre, on peut montrer qu'il n'est pas récursif:

Proposition 4.5.1 *L'ensemble des tautologies trivaluées du langage L, en calcul des prédicats, ayant pour seuls connecteurs $\neg$, $\rightarrow$, et pour quantificateur $\exists$ n'est pas récursif.*

Démonstration On considère l'ensemble $\mathfrak{F}$ des formules bivaluées closes et on construit un bijection α de $\mathfrak{F}$ sur $\alpha(\mathfrak{F})$ de la façon suivante:

Si f est une formule bivaluée close (ayant ses symboles parmi $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow, \exists, \forall$), comportant les prédicats $P_1, ..., P_n$ avec les variables libres, $x_1, ..., x_q$, on pose:

$$\alpha(F) = [(\forall x_1 ... \forall x_q\ (P_1(x_1, ..., x_q) \vee \neg P_1(x_1, ..., x_q)))\wedge \\ ...\wedge(\forall x_1 ... \forall x_q\ (P_1(x_1, ..., x_q) \vee \neg P_1(x_1, ..., x_q)))] \rightarrow F.$$

Cette formule devient une formule trivaluée puisqu'elle comporte le connecteur $\rightarrow$.

On a alors l'équivalence suivante:

F est une tautologie bivaluée $\Leftrightarrow$ $\alpha(F)$ est une tautologie trivaluée.

Montrons que F est une tautologie bivaluée implique que $\alpha(F)$ est une tautologie trivaluée.

Soit i une interprétation trivaluée.

Deux cas se présentent:

- $\forall j \in \{1, ..., n\}, \forall d_1...d_q \in D^q, vv_i(P_j(d_1, ..., d_q)) = V$ ou F. Dans ce cas, on fabrique une interprétation bivaluée i' qui prend la même valeur que i sur F, puisque seules comptent les valeurs de i' sur les symboles de prédicats intervenant dans F. On a alors $vv_{i'}(F) = V$ puisque F est une tautologie bivaluée; donc $vv_i(F) = V$ puisque $vv_i(F) = vv_{i'}(F)$. Donc $vv_i(\alpha(F)) = V$.

- $\exists j \in \{1, ..., n\}, vv_i(\forall x_1...\forall x_q\ (P_j(x_1, ..., x_n) \vee \neg P_j(x_1, ..., x_n))) = I$. Dans ce cas, $vv_i(\alpha(F)) = V$.

Montrons que $\alpha(F)$ est une tautologie trivaluée implique que F est une tautologie bivaluée.

Si $\alpha(F)$ est une tautologie trivaluée, elle est vraie en particulier dans toute interprétation bivaluée (cas particulier d'une interprétation trivaluée).

Dans une interprétation bivaluée i, on a toujours $\forall j \in \{1, ..., n\}$, $vv_i(\forall x_1...\forall x_q\ (P_j(x_1, ..., x_n) \vee \neg P_j(x_1, ..., x_n))) = V$ car $\forall j \in \{1, ..., n\}$, $\forall d_1...d_q \in D^q$, $vv_i(P_j(d_1, ..., d_q)) = V$ ou F puisque i est bivaluée. Par conséquent $vv_i(F) = V$ et F est une tautologie bivaluée.

Ceci permet alors de conclure que l'ensemble T des tautologies trivaluées n'est pas récursif. En effet, s'il l'était, l'ensemble des tautologies trivaluées de la forme $\alpha(F)$ le serait comme intersection de l'ensemble des formules $\alpha(F)$ où F est une formule bivaluée qui est un ensemble récursif et de l'ensemble des tautologies trivaluées. L'ensemble des formules $\alpha(F)$ où F est une formule bivaluée est un ensemble récursif puisque la bijection α de l'ensemble $\mathcal{F}$ des formules bivaluées closes sur l'ensemble $\alpha(\mathcal{F})$ est récursive et bi-récursive. Par conséquent l'ensemble des tautologies bivaluées serait récursif, puisque F est une tautologie bivaluée $\Leftrightarrow$ $\alpha(F)$ est une tautologie trivaluée; ce qui contredit le théorème de Church.

CHAPITRE 5: PROGRAMMES TRIVALUES. SEMANTIQUE. OPERATEURS ASSOCIES AUX PROGRAMMES. INTERPRETEURS.

Dans ce chapitre et dans le suivant, nous nous intéressons directement à la logique trivaluée pour étudier les programmes avec négation.

Le plan du chapitre est le suivant:

Dans une première partie (5.1), nous donnons les définitions des programmes étudiés.

Puis dans la partie 5.2, nous généralisons la propriété de monotonie obtenue pour les programmes bivalués aux programmes étudiés. Nous donnons ensuite la conséquence directe de cette propriété pour l'intersection de modèles d'un programme trivalué.

La partie 5.3 introduit l'opérateur conséquence associé à un programme trivalué et fournit ainsi une première sémantique dénotationnelle d'un programme avec négation. Le plus petit point fixe de cet opérateur est en effet le plus petit modèle d'un programme consistant. Ce plus petit point fixe se calcule facilement dans le cas d'un opérateur défini puisque l'opérateur conséquence associé au programme est alors continu. D'autres sémantiques peut-être mieux adaptées à la logique trivaluée seront fournies au chapitre 6.

La partie 5.4 concerne les différentes transformations pouvant être faites sur un programme trivalué (mise sous forme définie pour un programme sans quantificateur, mise sous forme normale). Elle introduit les deux complétions d'un programme avec négation (la $\neg$-complétion et la $\rightarrow$-complétion).

Dans la partie 5.5, nous étudierons les liens entre les différents opérateurs associés à un programme. Pour faire le lien avec l'opérateur de Fitting, nous utilisons la $\neg$-complétion ainsi que la $\rightarrow$-complétion définies au paragraphe précédent.

Enfin dans la partie 5.6, nous donnons une sémantique opérationnelle des programmes avec négation, en étudiant les interpréteurs. Par des opérations purement syntaxiques sur les interprétations et les programmes, tout interpréteur bivalué pourra être transformé en un interpréteur trivalué. Ceci a déjà été fait pour des interpréteurs Prolog (Delahaye-Matthieu [8 bis]).

5.1 DEFINITIONS

Définition 5.1.1 Une règle trivaluée est une formule de la forme:

lit ou **lit** $\leftarrow$ **for**

où for est une formule du langage L contenant les symboles $\forall$, $\exists$, $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$ et lit un littéral (définition 2.2.1) ne contenant pas le prédicat égalité.
Une règle est positive si elle comporte un atome (définition 2.2.1) en tête de règle.

Remarque: $\forall$, $\exists$, $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$ sont monotones
(cf. § 5.2.1 ci-dessous.).

Définition 5.1.2 Un programme trivalué est un ensemble fini de règles trivaluées.

Remarque: La notion de programme dans Lloyd [22] correspond à la notion à la notion de programme trivalué ne contenant que des règles positives.

Exemple 5.1.1 Soit Pr le programme trivalué suivant, comportant les symboles de prédicats plus, mult, et = et les symboles de fonctions su (d'arité 1), ze (constante).

plus(ze,x,x)
plus(su(x),y,su(z)) $\leftarrow$ plus(x, y, z)
mult(ze, x, ze)
mult(su(x),y,z) $\leftarrow$ mult(x, y, t) $\wedge$ plus(t, y, z)
$\neg$plus(x, y, z) $\leftarrow$ plus(x, y, t) $\wedge$ $\neg$(z = t)

$\neg$mult(x, y, z) $\leftarrow$ mult(x, y, t) $\wedge$ $\neg$(z = t)

Définition 5.1.3 Une règle définie trivaluée est une formule de la forme:

$$\textbf{lit} \text{ ou } \textbf{lit} \leftarrow \textbf{lit}_1 \wedge \textbf{lit}_2 \wedge \ldots \wedge \textbf{lit}_n$$

où lit est un littéral ne contenant pas le prédicat égalité et les lit_i sont des littéraux.

Définition 5.1.4 Un programme défini trivalué est un ensemble fini de règles définies trivaluées.

Remarque: La notion de programme normal dans Lloyd [22] correspond à la notion à la notion de programme défini trivalué ne contenant que des règles positives.

Définition 5.1.5 Une règle bivaluée est une formule de la forme:

$$\textbf{ato} \text{ ou } \textbf{ato} \leftarrow \textbf{for}$$

où for est une formule contenant les symboles $\forall$, $\exists$, $\wedge$, $\vee$ et ato est un atome ne contenant pas le prédicat égalité.

Remarque: $\forall$, $\exists$, $\wedge$, $\vee$ sont monotones (cf. § 5.2.1 ci-dessous.)

Définition 5.1.6 Un programme bivalué est un ensemble fini de règles bivaluées.

Définition 5.1.7 Une règle définie bivaluée est une formule de la forme:

$$\textbf{ato} \text{ ou } \textbf{ato} \leftarrow \textbf{ato}_1 \wedge \textbf{ato}_2 \wedge \ldots \wedge \textbf{ato}_n$$

où ato est un atome ne contenant pas le prédicat égalité et les ato_i sont des atomes.

Définition 5.1.8 Un programme défini bivalué est un ensemble fini de règles définies bivaluées.

Remarque: La notion de programme défini dans Lloyd [22] correspond à la notion à la notion de programme défini bivalué.

5.2 EXTENSION DE LA PROPRIETE DE MONOTONIE. INTERSECTION DE MODELES D'UN PROGRAMME TRIVALUE.

5.2.1 Monotonie.

Définition 5.2.1.1 On ordonne l'ensemble {V, F, I} par l'ordre induit par:

$$I \leq F, I \leq V.$$

On ordonne l'ensemble {V, F} par l'ordre induit par:

$$F \leq V.$$

On ordonne l'ensemble $IHT^{\infty}(L)$ par la relation d'ordre suivante:

$\forall$ i, j $\in$ IHT(L), i $\leq$ j $\Leftrightarrow$ i $\subseteq$ j; $\forall$ i$\in IHT^{\infty}(L)$, i $\leq$ Contra.

On ordonne l'ensemble IHB(L) par la relation d'ordre suivante:

$\forall$i, j $\in$ IHB(L), i $\leq$ j $\Leftrightarrow$ pos(i) $\subseteq$ pos(j);

Remarques: - L'élément Contra est rajouté à IHT(L) pour étudier les cas de programmes inconsistants.
- L'ordre sur {V, F, I} est celui qui correspond à l'ordre d'inclusion sur IHT(L): il rend la bijection ϕ: IHT(L) $\rightarrow$ {Fonctions de her(L) dans {V, F, I}}, i $\rightarrow$ vv_i, monotone.
- De même l'ordre sur {V, F} est celui qui correspond à l'ordre d'inclusion sur $2^{her(L)}$: il rend la bijection ϕ: $2^{her(L)} \rightarrow$ {Fonctions de her(L) dans {V, F}}, i $\rightarrow$ vv_i, monotone.

Proposition 5.2.1.1 $\neg$*: {V, F, I}* $\rightarrow$ *{V, F, I}*

$\wedge, \vee, \Rightarrow, \Leftrightarrow$*: $\{V, F, I\}^2 \rightarrow \{V, F, I\}$*

$\bigwedge$*: $\{V, F, I\}^{UNI(L)} \longrightarrow \{V, F, I\}$ et*

$\bigvee$*: $\{V, F, I\}^{UNI(L)} \longrightarrow \{V, F, I\}$*

avec $\bigwedge(h) = V$ (resp. $\bigvee(h) = V$) si $\forall t \in UNI(L)$ (resp. $\exists t \in UNI(L)$), $h(t) = V$, $\bigwedge(h) = F$ (resp. $\bigvee(h) = V$) si $\exists t \in UNI(L)$ (resp. $\forall t \in UNI(L)$), $h(t) = F$, $\bigwedge(h) = I$ (resp. $\bigvee(h) = I$) sinon, *où h est définie par: h: UNI(L) $\rightarrow$ {V, F, I}, h(t) = $vv_i(f(t))$, si $\wedge(h)$ = $vv_i(\forall x f)$, (resp. $\vee(h) = vv_i(\exists x f)$), sont monotones.*

$\leftrightarrow$, $\leftarrow\rightarrow$, $\rightarrow$ ne sont pas monotones de $\{V, F, I\}^2$ vers {V, F, I}. C'est pourquoi le connecteur $\rightarrow$ s'appelle aussi implication non monotone.

Démonstration $\rightarrow$ (I, F) = Vrai et $\rightarrow$ (V, F) = Faux;
$\leftarrow\rightarrow$ (I, V) = Faux and $\leftarrow\rightarrow$ (V, V) = Vrai;
$\leftrightarrow$ (I, I) = Vrai and $\leftrightarrow$ (I, F) = Faux;

Montrons par exemple que $\wedge$ est monotone.

Soient f et g deux éléments de $\{V, F, I\}^{UNI(L)}$.

$f \leq g$ si et seulement si $\forall t \in$ UNI(L), $f(t) \leq g(t)$.

$\wedge(f) = I$ implique $\wedge(f) \leq \wedge(g)$.

$\wedge(f) = V$ implique $\forall t \in$ UNI(L), f(t) = V et donc $\wedge(g) = V$.

$\wedge(f) = F$ implique $\exists t \in$ UNI(L), f(t) = F et donc $\wedge(g) = F$. ♦

Si For(L) désigne les formules qui constituent les corps des règles des programmes étudiés, i.e les formules du langage comportant les symboles $\forall$, $\exists$, $\neg$, $\vee$, $\wedge$, $\Rightarrow$, $\Leftrightarrow$, on peut étendre la propriété de monotonie obtenue en logique classique à ces formules.

Proposition 5.2.1.2 *L'application ϕ: IHT(L) $\rightarrow$ {Fonctions de For(L) vers {V, F, I}}, telle que $\phi(i) = vv_i$ est monotone pour l'ordre d'inclusion sur IHT(L) et l'ordre induit par $I \leq V$, $I \leq F$ sur l'ensemble des fonctions de For(L) vers {V, F, I}.*
Ce qui signifie: si f une formule comportant les quantificateurs $\forall$, $\exists$ et les connecteurs $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$,

$$\forall i, j \in IHT(L),\ i \subseteq j \text{ implique que } vv_i(f) \leq vv_j(f).$$

Démonstration Par induction sur la complexité de f.
- Si f = ato,
  - soit $vv_i(f) = I$ et $vv_i(f) \leq vv_j(f)$ ou $vv_i(f) \neq I$;
  - soit $vv_i(f) \neq I$, $vv_i(f) = vv_j(f)$ car $i \subseteq j$.

- Si f = $\neg g$, $h \wedge g$, $h \vee g$, $h \Rightarrow g$, $h \Leftrightarrow g$, on a le résultat par monotonie de $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$ (proposition 5.2.1.1) et par induction.

- De même si $f = \forall x\ g$ ou $\exists x\ g$ par monotonie de $\wedge$ et $\vee$, et par induction.
♦

Remarques: - En logique classique, on a aussi une propriété de monotonie: si F(L) désigne les formules qui constituent les corps des règles des programmes sans négation, i.e les formules du langage comportant les symboles $\forall$, $\exists$, $\vee$, $\wedge$, l'application:
ϕ: $2^{\text{her}(L)} \longrightarrow$ {Fonctions de F(L) vers {V, F}}, telle que $\phi(i) = vv_i$ est monotone pour l'ordre d'inclusion sur $2^{\text{her}(L)}$ et l'ordre induit par $F \leq V$ sur l'ensemble des fonctions de F(L) vers {V, F}. Ce qui signifie:

Si f est une formule comportant les quantificateurs $\forall$, $\exists$ et les connecteurs $\wedge$, $\vee$,

$$\forall\ i, j \in IHB(L),\ pos(i) \subseteq pos(j) \text{ implique que } vv_i(f) \leq vv_j(f).$$

- Le connecteur d'implication n'est pas monotone aussi dans le sens où si on inclut dans For(L), les formules utilisant $\longrightarrow$, l'application:
ϕ: IHT(L) $\longrightarrow$ {Fonctions de For(L) vers {V, F, I}}, telle que $\phi(i) = vv_i$ n'est plus monotone.

- Cette propriété de monotonie signifie que si une formule for comportant les symboles $\wedge$, $\vee$, $\neg$, $\Rightarrow$, $\Leftrightarrow$, $\forall$, $\exists$, possède des atomes dont la la valeur de vérité n'est pas encore connue (Indéterminé), la valeur de vérité de cette formule ne peut changer (passer de Vrai à Faux ou réciproquement) mais seulement devenir connue (passer de Indéterminé à Vrai ou Faux).

Cette généralisation de la propriété de monotonie permet d'obtenir avec la table de vérité du connecteur d'implication, la propriété d'intersection des modèles.

5.2.2 Intersections de modèles de Herbrand de programmes trivalués.

Remarquons tout d'abord que pour un programme trivalué, les deux notions de modèle de Herbrand fort et de modèle de Herbrand faible coïncident.

Proposition 5.2.2.1 *Soit Pr un programme trivalué et $i \in IHT(L)$. L'interprétation i est un modèle fort de Pr $\Leftrightarrow$ i est un modèle faible de Pr.*

Démonstration On montre par induction sur la complexité de g comportant les symboles:

$\forall$, $\exists$, $\neg$, $\wedge$, $\vee$, $\leftrightarrow$, $\Rightarrow$, $\Leftrightarrow$, $\leftarrow\rightarrow$, $\leftarrow$, $\rightarrow$, que, si $f = g(f_1, f_2, ..., f_m)$ où $f_1, f_2, ..., f_m$ ont leur connecteur de plus haut niveau parmi ($\leftrightarrow$, $\leftarrow\rightarrow$, $\leftarrow$, $\rightarrow$), alors:

i est un modèle fort de f $\Leftrightarrow$ i est un modèle faible de f.

Alors comme: $\forall Pr \equiv_T \forall(lit_1 \leftarrow for_1 \wedge ... \wedge lit_n \leftarrow for_n)$, on aura le résultat.

Si g n'a pas de connecteur, alors $f = f_1$; comme $\leftrightarrow$, $\leftarrow\rightarrow$, $\leftarrow$, $\rightarrow$ ne prennent jamais la valeur de vérité Indéterminé, on a le résultat.

On utilise ensuite le fait que $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$, $\leftrightarrow$, $\leftarrow\rightarrow$, $\leftarrow$, $\rightarrow$ restreints à $\{V, F\}$ prennent leurs valeurs dans $\{V, F\}$ et l'hypothèse d'induction.

Si $g = \forall x\ F$, alors $vv_i(g) = I$ si et seulement si $\forall t \in UNI(L)$, $vv_i(F(t)) \neq F$ et $\exists t \in UNI(L)$, $vv_i(F(t)) = I$. De même pour $g = \exists x\ F$. ♦

A partir de maintenant, on ne considèrera que les modèles de Herbrand trivalués forts que l'on appellera simplement modèles de Herbrand trivalués.

On a alors la propriété d'intersection des modèles:

Proposition 5.2.2.2 *Si Pr est un programme trivalué et $(i_e)_{e\in E}$ une famille de modèles de Herbrand trivalués de Pr, alors:*

$\bigcap_{e\in E} i_e$ *est un modèle de Herbrand trivalué de Pr.*

Démonstration Soit r une règle de Pr. Pour toute instance close $lit \leftarrow for$ de r, $\forall e \in E$, $vv_{i_e}(lit \leftarrow for) = Vrai$ car i_e est un modèle de $\forall r$.

On montre que, si $i = \bigcap_{e\in E} i_e$, alors $vv_i(lit \leftarrow for) = Vrai$, pour toute instance close $lit \leftarrow for$ de r.

Le seul cas à envisager est $\underline{vv_i(for) = Vrai}$ à cause de la table de vérité de $\rightarrow$.

Puisque for ne contient que les symboles $\forall$, $\exists$, $\neg$, $\wedge$, $\vee$, $\Leftrightarrow$, $\Rightarrow$, $\forall e \in E$, $vv_i(for) \leq vv_{i_e}(for)$ par monotonie. Si $vv_i(for) = Vrai$, alors $\forall e\in E$, $vv_{i_e}(for) = Vrai$ et par conséquent $\forall e\in E$, $vv_{i_e}(lit) = Vrai$ puisque $\forall e \in E$, $vv_{i_e}(lit \leftarrow for) = Vrai$.
Comme lit est un littéral clos, $\forall e \in E$, $lit \in i_e$, par conséquent $lit \in i$ et $vv_i(lit \leftarrow for) = Vrai$, pour toute instance close $lit \leftarrow for$ de r. ♦

Remarques: - En logique bivaluée, la propriété d'intersection des modèles dans le formalisme correspondant est: si Pr est un programme bivalué et $(i_e)_{e\in E}$ une famille de modèles de Herbrand bivalués de Pr, alors $pos^{-1}(\bigcap_{e\in E} pos(i_e))$ est un modèle de Herbrand bivalué de Pr.

- $\bigcap_{e\in E} i_e$ est l'inf (borne inférieure) du sous-ensemble $\{i_e, e \in E\}$ de IHT(L) ordonné par $\leq$, défini dans la definition 5.2.1.1.

- $pos^{-1}(\bigcap_{e\in E} pos(i_e))$ est l'inf (borne inférieure) du sous-ensemble $\{i_e, e \in E\}$ de IHB(L) ordonné par $\leq$, défini dans la definition 5.2.1.1.

Proposition 5.2.2.3 *Soit Pr un programme trivalué.*
Soit Pr est inconsistant au sens de Herbrand, soit Pr possède un plus petit modèle de Herbrand trivalué, qui est l'intersection de tous ses modèles de Herbrand trivalués: c'est aussi l'ensemble de tous les littéraux clos, conséquences logiques de Pr. On le note ppmt(Pr).

Démonstration Si Pr est consistant, alors Pr possède un modèle trivalué.
L'intersection de tous les modèles de Herbrand trivalués de Pr est un modèle de Herbrand trivalué de Pr par la proposition 5.2.2.2. C'est le plus petit modèle de Herbrand trivalué au sens de l'inclusion.
lit $\in \bigcap i_e$, i_e modèle de Pr
$\Leftrightarrow$ Tout modèle de Herbrand de Pr est un modèle de Herbrand de lit (lit est un littéral clos)
$\Leftrightarrow$ lit est une conséquence logique de Pr. ♦

Remarque: Si Pr est un programme bivalué, Pr possède un plus petit modèle de Herbrand bivalué, dont la partie positive est l'ensemble de tous les atomes clos, conséquences logiques de Pr. On le note ppmb(Pr).

5.3 OPERATEUR CONSEQUENCE ASSOCIE A UN PROGRAMME AVEC NEGATION. UNE SEMANTIQUE DENOTATIONNELLE POUR LES PROGRAMMES AVEC NEGATION.

Puisque l'on admet des programmes ayant en tête de règle un littéral négatif, un programme trivalué peut être inconsistant. C'est pourquoi on a rajouté à IHT(L) l'élément Contra qui sert pour le cas des programmes inconsistants. On définit alors l'opérateur sur IHT(L) $\cup$ {Contra}. Dans le cas d'un programme inconsistant, l'ensemble des post-points fixes de cet opérateur sera réduit au singleton {Contra}.

Définition 5.3.1 Soit Pr un programme trivalué.
T_{Pr} est l'application suivante:

T_{Pr}: IHT(L) ∪ {Contra} → IHT(L) ∪ {Contra}.

Si i ∈ IHT(L), alors:

$$T_{Pr}(i) = \{\text{lit} \in \text{Her(L)}/ \text{ il existe une instance close lit} \leftarrow \text{for d'une règle de Pr telle que } vv_i(\text{for}) = \text{Vrai}\}$$

si cet ensemble ne contient pas un atome et sa négation,

$$T_{Pr}(i) = \text{Contra sinon,}$$

$$T_{Pr}(\text{Contra}) = \text{Contra}.$$

De la propriété de monotonie obtenue pour les programmes avec négation, résulte la monotonie de cet opérateur. Cet opérateur admet alors un plus petit point fixe d'après le théorème de Birkhoff-Tarski:

Théorème: *Si E est un ensemble partiellement ordonné ayant un plus petit élément ⊥ et tel que tout sous ensemble totalement ordonné admette une borne sup, alors toute application monotone T: E→E admet un plus petit point fixe, noté pppf(T).*

Cet opérateur fournit donc une sémantique dénotationnelle d'un programme Pr d'après le théorème suivant.

Théorème 5.3.1 *1) T_{Pr} est monotone et admet un plus petit point fixe noté pppf(T_{Pr}).*

2) Pr est consistant ⇔ pppf(T_{Pr}) ≠ Contra.
Dans ce cas:
a) une interprétation i ∈ IHT(L) est un modèle de Pr ⇔ $T_{Pr}(i) \subseteq i$.
b) pppf(T_{Pr}) = ppmt(Pr) = ∩(modèles de Pr).

Démonstration 1) Si i ∈ IHT^∞(L), i ⊆ Contra et $T_{Pr}(i)$ ⊆ Contra = T_{Pr}(Contra).
Soient i et j deux éléments de IHT(L).

$i \subseteq j$ implique que $vv_i(for) \leq vv_j(for)$ si la formule for ne comprend que les symboles $\exists, \forall, \wedge, \vee, \Rightarrow, \Leftrightarrow$, ce qui est le cas de toute formule constituant le corps d'une règle de Pr.
T_{Pr} est alors monotone. T_{Pr} a un plus petit point fixe car $IHT^{\infty}(L)$ satisfait les conditions du théorème de Birkhoff-Tarski, puisque l'ensemble vide $\emptyset$ est son plus petit élément et tout sous ensemble totalement ordonné $\{I_\alpha, \alpha \in A\}$ de $IHT^{\infty}(L)$ possède une borne sup qui est Contra s'il existe $\alpha \in A$ tel que I_α = Contra et $\cup_{\alpha \in A} I_\alpha$, sinon.

2) Pour montrer que Pr est consistant si et seulement si $pppf(T_{Pr}) \neq$ Contra, on montre d'abord que:

a) si $i \in IHT(L)$ est tel que $T_{Pr}(i) \subseteq i$, alors i est un modèle de Pr.

Soit lit $\leftarrow$ for une instance close d'une règle de Pr. Montrons que vv_i(lit $\leftarrow$ for) = Vrai, si $i \in IHT(L)$ est telle que $T_{Pr}(i) \subseteq i$. Le seul cas à examiner est celui où $vv_i(for)$ = Vrai.
Comme $i \in IHT(L)$ et $T_{Pr}(i) \subseteq i$, $T_{Pr}(i) \neq$ Contra et lit $\in T_{Pr}(i)$ par définition de T_{Pr}. Donc lit $\in$ i, et $vv_i(lit)$ = Vrai.

On montre ensuite que:

b) Si $i \in IHT(L)$ est un modèle de Pr, alors $T_{Pr}(i) \subseteq i$.
On a vv_i(lit $\leftarrow$ for) = Vrai, pour toute instance close d'une règle de Pr. Si lit $\in T_{Pr}(i)$, alors il existe une instance close lit $\leftarrow$ for d'une règle de Pr telle que $vv_i(for)$ = Vrai. Alors $vv_i(lit)$ = Vrai et lit $\in$ i.

c) On peut ensuite montrer par induction transfinie que:

Si $i \in IHT(L)$ est un modèle de Pr, alors $T_{Pr}\uparrow\alpha \subseteq i$, pour tout ordinal α.
$\alpha = 0$, $T_{Pr}\uparrow 0 = \emptyset$.
Si α est un ordinal successeur, $T_{Pr}(T_{Pr}\uparrow\alpha\text{-}1) \subseteq T_{Pr}(i)$, par monotonie de T_{Pr} et induction, et $T_{Pr}(i) \subseteq i$ d'après b) ci-dessus puisque $i \in IHT(L)$ est un modèle de Pr.
Si α est un ordinal limite, $T_{Pr}\uparrow\alpha = \sup\{T_{Pr}\uparrow\beta, \beta < \alpha\} \subseteq i$, par induction.
Comme $pppf(T_{Pr}) = T_{Pr}\uparrow\alpha$, pour un certain ordinal α, $pppf(T_{Pr}) \subseteq i$, pour tout modèle i de Pr.

Si Pr est consistant, il existe $i \in IHT(L)$ modèle de Pr, et $pppf(T_{Pr}) \subseteq i$ donc $pppf(T_{Pr}) \neq$ Contra.

Réciproquement, si $pppf(T_{Pr}) \neq$ Contra, $pppf(T_{Pr}) \in IHT(L)$ et $T_{Pr}(pppf(T_{Pr})) \subseteq pppf(T_{Pr})$ donc $pppf(T_{Pr})$ est un modèle de Pr qui est consistant.
On a montré que $pppf(T_{Pr})$ est un modèle de Pr s'il est différent de Contra.
Comme $pppf(T_{Pr}) \subseteq$ i,pour tout i modèle de Pr, si Pr est consistant, $pppf(T_{Pr}) = ppmt(Pr) = \cap$(modèles de Pr), d'après la proposition 5.2.2.3.
♦

Dans la proposition suivante, on connaît exactement l'ordinal α tel que $T_{Pr}\uparrow\alpha = pppf(T_{Pr})$ dans le cas où Pr est un programme défini (ayant des règles de la forme lit $\leftarrow$ $lit_1, ..., lit_n$, où lit est un littéral ne contenant pas le prédicat d'égalité et les lit_i sont des littéraux)

Proposition 5.3.1 *a) Si Pr est un programme défini trivalué, alors T_{Pr} est continu. T_{Pr} n'est pas continu en général.*

b) Si Pr est un programme défini trivalué, alors

$$ppmt(Pr) = pppf(T_{Pr}) = T_{Pr}\uparrow\omega = \cup_{m \in \mathcal{N}} T_{Pr}{}^{m}(\varnothing).$$

Démonstration a) Soit X un sous-ensemble dirigé de IHT(L), où L est le langage sous-jacent à Pr. ($\forall x, y \in X, \exists z \in X$ tel que $x \leq z$ et $y \leq z$).

On vérifie que $sup(T_{Pr}(X)) = T_{Pr}(sup(X))$ où (sup(X) est la borne supérieure de X).

lit $\in T_{Pr}(sup(X))$ $\Leftrightarrow$ lit $\leftarrow lit_1 \wedge lit_2 \wedge ... \wedge lit_n$ est une instance close d'une règle de Pr telle que: $vv_{sup(X)}(lit_1 \wedge lit_2 \wedge ... \wedge lit_n)$ = Vrai.

$\Leftrightarrow$ lit $\leftarrow lit_1 \wedge lit_2 \wedge ... \wedge lit_n$ est une instance close d'une règle de Pr telle que: $\forall i = 1, 2, ..., n$, $lit_i \in sup(X)$

$\Leftrightarrow$ lit $\leftarrow lit_1 \wedge lit_2 \wedge ... \wedge lit_n$ est une instance close d'une règle de Pr telle que: $\exists$ I $\in$ X tel que $\forall i = 1, 2, ..., n$, $lit_i \in I$ puisque X est un sous-ensemble dirigé de IHT(L)

$\Leftrightarrow$ $\exists I \in X$ tel que lit $\leftarrow lit_1 \wedge lit_2 \wedge ... \wedge lit_n$ soit une instance close d'une règle de Pr telle que: $vv_I(lit_1 \wedge lit_2 \wedge ... \wedge lit_n)$ = Vrai.

$\Leftrightarrow \quad \exists I \in X$ tel que lit $\in T_{Pr}(I)$

$\Leftrightarrow \quad$ lit $\in \sup(T_{Pr}(X))$.

- $T_{Pr}(\sup(X)) = \text{Contra}$

$\Leftrightarrow \quad$ ato $\leftarrow lit_1 \wedge lit_2 \wedge \ldots \wedge lit_n$ et
$\neg$ato $\leftarrow lit'_1 \wedge lit'_2 \wedge \ldots \wedge lit'_p$ sont des instances closes de Pr telles que:
$\forall i = 1, 2, \ldots, n,\ vv_{sup(X)}(lit_i) = \text{Vrai}$ et
$\forall j = 1, 2, \ldots, n,\ vv_{sup(X)}(lit'_j) = \text{Vrai}$.

$\Leftrightarrow \quad \exists I \in X$ tel que ato $\in T_{Pr}(I)$ et $\neg$ato $\in T_{Pr}(I)$ puisque X est dirigé.

$\Leftrightarrow \quad \exists I \in X$ tel que $T_{Pr}(I) = \text{Contra}$

$\Leftrightarrow \quad \sup(T_{Pr}(X)) = \text{Contra}$. ♦

Lorsque Pr comporte des quantificateurs et des symboles de fonctions, T_{Pr} n'est pas continu en général. Par exemple, si on prend pour Pr le programme:

$$P(a) \leftarrow \forall x\ Q(x)$$
$$Q(a) \leftarrow$$
$$Q(f(x)) \leftarrow Q(x)$$

Soit $I_0 = \emptyset$; $I_1 = \{Q(a)\}$; $I_2 = \{Q(a), Q(f(a))\}$;...; $I_n = I_{n-1} \cup \{Q(f^{n-1}(a))\}$

On obtient ainsi une suite croissante d'interprétations trivaluées.

$$T_{Pr}(\cup_{n\geq 0} I_n) = \{P(a)\} \cup \cup_{n\geq 0} I_n.$$
$$T_{Pr}(I_n) = I_{n+1}.$$
$$\cup_{n\geq 0} T_{Pr}(I_n) = \cup_{n\geq 0} I_n.$$

Remarques: - Si Pr est un programme sans quantificateur, alors $\exists$Pr' défini tel que: $\forall Pr \equiv_T \forall Pr'$. (prop. 5.4.1). T_{Pr} sera continu car on montrera que $T_{Pr} = T_{Pr'}$.

- Si Pr est un programme ne comportant pas de symbole de fonction, alors toute croissante d'interprétations est stationnaire et le problème de continuité ne se pose pas.

5.4 TRANSFORMATIONS DE PROGRAMMES

Nous allons montrer que les méthodes de transformations utilisées en calcul bivalué sont encore valables en calcul trivalué.

5.4.1 Transformation d'un programme trivalué sans quantificateur en un un programme défini trivalué.

Lemme 5.4.1 *Une formule sans quantificateur, n'utilisant que les symboles $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$ est logiquement équivalente à une formule en forme normale disjonctive.*

Démonstration On adapte ici les méthodes bivaluées classiques.
On remplace chaque occurrence de: $F \Rightarrow G$ par $\neg F \vee G$, $F \Leftrightarrow G$ par $(\neg F \vee G) \wedge (\neg G \vee F)$, $\neg(F \vee G)$ par $\neg F \wedge \neg G$, $\neg(F \wedge G)$ par $\neg F \vee \neg G$, $\neg\neg F$ par F et enfin chaque occurrence de $(F \vee G) \wedge H$ par $(F \wedge H) \vee (G \wedge H)$ et de $F \wedge (G \vee H)$ par $(F \wedge G) \vee (F \wedge H)$.♦

Définition 5.4.1 Soit Ax un ensemble de formules de L. On note $\forall$Ax, la formule obtenue en prenant la conjonction des clôtures universelles des formules de Ax.

Proposition 5.4.1 *Soit Pr un progamme trivalué sans quantificateur. Alors il existe un programme défini trivalué Pr' tel que: $\forall Pr \equiv_T \forall Pr'$.
De même pour le cas bivalué.*

Démonstration Pr est un ensemble de règles r de la forme:

$$\mathrm{lit}(t_1, ..., t_p) \leftarrow \mathrm{for}(z_1, ..., z_q).$$

$\{z_1, ..., z_q\}$ est l'ensemble des variables libres de for et $\{y_1, ..., y_k\}$ est l'ensemble des variables libres de $t_1, ..., t_p$.

Si $\{x_1, ..., x_n\} = \{z_1, ..., z_q\} \cup \{y_1, ..., y_k\}$, r est de la forme:

$$\mathrm{lit}(t_1(x_1, ..., x_n), ..., t_p(x_1, ..., x_n)) \leftarrow \mathrm{for}(x_1, ..., x_n).$$

D'après le lemme 5.4.1, $\mathrm{for}(x_1, ..., x_n) \equiv_T$

$$(\mathrm{lit}_{11} \wedge ... \wedge \mathrm{lit}_{1n_1}) \vee \\ ... \vee (\mathrm{lit}_{m1} \wedge ... \wedge \mathrm{lit}_{mn_m}).$$

en mettant for en forme normale disjonctive.

Par le lemme de substitution 2.2.9,

$$r \equiv_T lit \leftarrow [(lit_{11}\wedge...\wedge lit_{1n_1})\vee ...\vee(lit_{m1}\wedge...\wedge lit_{mn_m})].$$

On utilise ensuite le fait que: $A_1\vee A_2\vee...\vee A_n \rightarrow B \equiv_T (A_1 \rightarrow B)\wedge...\wedge(A_n \rightarrow B)$. (exemples 2.1.1)

On a donc:

$$r \equiv_T (lit \leftarrow (lit_{11}\wedge...\wedge lit_{1n_1}))\wedge ...\wedge(lit \leftarrow (lit_{m1}\wedge...\wedge lit_{mn_m}))$$

On a alors (lemme 2.2.6):

$$\forall r \equiv_T$$
$$\forall x_1\forall x_2...\forall x_n\, (lit \leftarrow (lit_{11}\wedge...\wedge lit_{1n_1}))\wedge...\wedge(lit \leftarrow (lit_{m1}\wedge...\wedge lit_{mn_m}))$$

On utilise enfin le fait que: $\forall x\, (F\wedge G) \equiv_T (\forall x\, F)\wedge(\forall x\, G)$. ♦

Par exemple, si Pr est le progamme:

$$A(x) \leftarrow (B(x)\vee C(x))\wedge D(x)$$
$$B(x) \leftarrow E(x)$$

$\forall Pr \equiv_T \forall Pr'$ avec Pr':

$$A(x) \leftarrow (B(x)\wedge D(x))$$
$$A(x) \leftarrow (C(x)\wedge D(x))$$
$$B(x) \leftarrow E(x)$$

Remarque: On en déduit alors que T_{Pr} est continu si Pr est un programme sans quantificateur. En effet: $T_{Pr}(i) = \{lit \in Her(L)/$ il existe une instance close $lit \leftarrow for$ d'une règle de Pr telle que: $vv_i(for) = Vrai\}$ et si Pr' est le programme défini tel que $\forall Pr \equiv_T \forall Pr'$, Pr' s'obtient en mettant for sous forme normale disjonctive.

D'après le lemme 5.4.1, $for(x_1, ..., x_n) \equiv_T (lit_{11}\wedge...\wedge lit_{1n_1})\vee ...\vee(lit_{m1}\wedge...\wedge lit_{mn_m})$.

Si lit $\in T_{Pr}(i)$, il existe une instance close lit $\leftarrow$ for d'une règle de Pr telle que: $vv_i(for)$ = Vrai, et l'une des disjonctions va être vraie donc lit $\in T_{Pr'}(i)$.
Réciproquement, si lit $\in T_{Pr'}(i)$, l'une des disjonctions est vraie et on aura $vv_i(for)$ = Vrai, par conséquent lit $\in T_{Pr}(i)$.

5.4.2 Mise sous forme normale.

Définition 5.4.2 Soient m règles positives $r_1, r_2, ..., r_m$ ayant en tête de règle des littéraux bâtis sur le même symbole de prédicat p;
r_i est de la forme:

$$p(t_{i1}, ..., t_{in}) \leftarrow for_i;\ \text{pour } i = 1, ..., m.$$

La forme normale du paquet $(r_1, r_2, ..., r_m)$ est la règle:

$$p(x_1, ..., x_n) \leftarrow for'_1 \vee ... \vee for'_m$$

où for'_i est la formule close:

$$\exists y_1 ... \exists y_p (x_1 = t_{i1} \wedge ... \wedge x_n = t_{in} \wedge for_i)$$

où $y_1, ..., y_p$ sont les variables libres de $for_i, t_{i1}, t_{i2}, ..., t_{in}$ et où les x_i sont des variables différentes des y_i.
On définit de même la forme normale d'un paquet de règles ayant en tête de règles des littéraux négatifs bâtis sur un même symbole de prédicat p.

Proposition 5.4.2 *Si r est la règle obtenue par mise sous forme normale du paquet:*

$$(r_1, r_2, ..., r_m),$$

on a:

$$\forall r \equiv_{TH} \forall r_1 \wedge ... \wedge \forall r_m.$$

Démonstration Soit r_i la règle:

$$p(t_{i1}, ..., t_{in}) \leftarrow for_i$$

où $y_1, ..., y_p$ sont les variables libres de $for_i, t_{i1}, t_{i2}, ..., t_{in}$.

Puisque for'_i est une formule sans variable libre, on a:

$$\forall r = \forall x_1 ... \forall x_n \, (p(x_1, ..., x_n) \leftarrow for'_1 \vee ... \vee for'_m)$$

Par le fait que:

$$A_1 \vee A_2 \vee ... \vee A_n \rightarrow B \equiv_T (A_1 \rightarrow B) \wedge ... \wedge (A_n \rightarrow B),$$

et que $\forall x \, (A(x) \wedge B(x)) \equiv_T (\forall x \, A(x) \wedge \forall x \, B(x))$

on a:

$$\begin{aligned} &\forall x_1 ... \forall x_n \, (p(x_1, ..., x_n) \leftarrow for'_1 \vee ... \vee for'_m) \equiv_T \\ &\forall x_1 ... \forall x_n \, (p(x_1, ..., x_n) \leftarrow for'_1) \wedge \\ &\quad ... \wedge \forall x_1 ... \forall x_n \, (p(x_1, ..., x_n) \leftarrow for'_m). \end{aligned}$$

En utilisant le fait que:

$$\exists x \, A(x) \rightarrow B \equiv_T \forall x \, (A(x) \rightarrow B)$$

si x n'est pas variable libre de B, on a:

$$\begin{aligned} &\forall x_1 ... \forall x_n \, (p(x_1, ..., x_n) \leftarrow for'_i) \equiv_T \\ &\forall x_1 ... \forall x_n \forall y_1 ... \forall y_p \, (p(x_1, ..., x_n) \leftarrow (x_1 = t_{i1} \wedge ... \wedge x_n = t_{in} \wedge for_i)) \end{aligned}$$

car $y_1, ..., y_p$ ne sont pas des variables libres de $p(x_1, ..., x_n)$ puisque $x_i \neq y_i$.

Or:

$$\begin{aligned} &\forall x_1 ... \forall x_n \forall y_1 ... \forall y_p \, (p(x_1, ..., x_n) \leftarrow (x_1 = t_{i1} \wedge ... \wedge x_n = t_{in} \wedge for_i)) \\ &\qquad \equiv_{TH} \forall y_1 ... \forall y_p \, (p(t_{i1}, ..., t_{in}) \leftarrow for_i). \end{aligned}$$

En effet dans toute interprétation $\mathcal{U}$ où l'égalité est interprétée par l'égalité syntaxique, on a:

$$\begin{aligned} &vv_{\mathcal{U}}(\forall x_1 ... \forall x_n \forall y_1 ... \forall y_p \, (p(x_1, ..., x_n) \leftarrow (x_1 = t_{i1} \wedge ... \wedge x_n = t_{in} \wedge for_i)) \\ &\qquad \leftrightarrow \forall y_1 ... \forall y_p \, (p(t_{i1}, ..., t_{in}) \leftarrow for_i)) = V. \end{aligned}$$

En effet, on démontre que si ϕ est une assignation, et $\mathcal{U}$ une interprétation où l'égalité est interprétée par l'égalité syntaxique,

$$vv_{\phi}(\forall x_1 ... \forall x_n \forall y_1 ... \forall y_p \, (p(x_1, ..., x_n) \leftarrow (x_1 = t_{i1} \wedge ... \wedge x_n = t_{in} \wedge for_i))$$
$$=$$
$$vv_{\phi}(\forall y_1 ... \forall y_p \, (p(t_{i1}, ..., t_{in}) \leftarrow for_i).$$

Comme les formules:

$\forall x_1 \ldots \forall x_n \forall y_1 \ldots \forall y_p\, (p(x_1, \ldots, x_n) \leftarrow (x_1 = t_{i1} \wedge \ldots \wedge x_n = t_{in} \wedge for_i)$ et

$\forall y_1 \ldots \forall y_p\, (p(t_{i1}, \ldots, t_{in}) \leftarrow for_i)$

ne prennent jamais la valeur I, il suffit de montrer que:

$$vv_\phi(\forall x_1 \ldots \forall x_n \forall y_1 \ldots \forall y_p\, (p(x_1, \ldots, x_n) \leftarrow (x_1 = t_{i1} \wedge \ldots \wedge x_n = t_{in} \wedge for_i)) = V$$

si et seulement si

$$vv_\phi(\forall y_1 \ldots \forall y_p\, (p(t_{i1}, \ldots, t_{in}) \leftarrow for_i) = V.$$

$$vv_\phi(\forall x_1 \ldots \forall x_n \forall y_1 \ldots \forall y_p\, (p(x_1 \ldots, x_n) \leftarrow (x_1 = t_{i1} \wedge \ldots \wedge x_n = t_{in} \wedge for_i)) = V$$

équivaut à (1):

$\forall (c_1, \ldots, c_n) \in D^n, \forall (d_1, \ldots, d_p) \in D^p,$

$[vv_{\phi(x_1 \leftarrow c_1)\ldots(x_n \leftarrow c_n)(y_1 \leftarrow d_1)\ldots(y_p \leftarrow d_p)}(x_1 = t_{i1} \wedge \ldots \wedge x_n = t_{in} \wedge for_i)) = V$

implique:

$vv_{\phi(x_1 \leftarrow c_1)\ldots(x_n \leftarrow c_n)(y_1 \leftarrow d_1)\ldots(y_p \leftarrow d_p)}(p(x_1, \ldots, x_n)) = V]$

De même, $vv_\phi(\forall y_1 \ldots \forall y_p\, (p(t_{i1}, \ldots, t_{in}) \leftarrow for_i) = V$

équivaut à (2):

$\forall (d_1, \ldots, d_p) \in D^p$, $[vv_{\phi(y_1 \leftarrow d_1)\ldots(y_p \leftarrow d_p)}(for_i) = V$ implique:

$$vv_{\phi(y_1 \leftarrow d_1)\ldots(y_p \leftarrow d_p)}(p(t_{i1}, \ldots, t_{in})) = V].$$

Supposons que (1) soit vérifié et que l'on ait (3):

$$vv_{\phi(y_1 \leftarrow d_1)\ldots(y_p \leftarrow d_p)}(for_i) = V.$$

On choisit les c_j tels que $c_j = \phi_{(y_1 \leftarrow d_1)\ldots(y_p \leftarrow d_p)}(t_{ij}(y_1, \ldots, y_p))$, ($\phi_{(y_1 \leftarrow d_1)\ldots(y_p \leftarrow d_p)}$ est le prolongement de $\phi_{(y_1 \leftarrow d_1)\ldots(y_p \leftarrow d_p)}$ défini en 2.2.4) alors:

$$vv\phi(x_1 \leftarrow c_1)...(x_n \leftarrow c_n)(y_1 \leftarrow d_1)...(y_p \leftarrow d_p)$$
$$(x_1 = t_{i1} \wedge ... \wedge x_n = t_{in} \wedge for_i)) = V,$$

puisque l'égalité est interprétée par l'égalité syntaxique et que l'on a (3).

Donc par (1), on a:

$$vv\phi(x_1 \leftarrow c_1)...(x_n \leftarrow c_n)(y_1 \leftarrow d_1)...(y_p \leftarrow d_p)(p(x_1, ..., x_n)) = V$$

Mais $vv\phi(x_1 \leftarrow c_1)...(x_n \leftarrow c_n)(y_1 \leftarrow d_1)...(y_p \leftarrow d_p)(p(x_1, ..., x_n)) =$

$$vv\phi(x_1 \leftarrow c_1)...(x_n \leftarrow c_n)(p(x_1, ..., x_n)) =$$

$$vv\phi(y_1 \leftarrow d_1)...(y_p \leftarrow d_p)(p(t_{i1}, ..., t_{in})) = V$$
car $c_j = \phi(y_1 \leftarrow d_1)...(y_p \leftarrow d_p)(t_{ij}(y_1, ..., y_p))$.

On a donc (2).

Supposons que (2) est vérifié et que l'on ait (4):

$$vv\phi(x_1 \leftarrow c_1)...(x_n \leftarrow c_n)(y_1 \leftarrow d_1)...(y_p \leftarrow d_p)$$
$$(x_1 = t_{i1} \wedge ... \wedge x_n = t_{in} \wedge for_i)) = V$$

Nécessairement, on a $c_j = \phi(y_1 \leftarrow d_1)...(y_p \leftarrow d_p)(t_{ij}(y_1, ..., y_p))$ car l'égalité est interprétée par l'égalité syntaxique.

On a aussi $vv\phi(x_1 \leftarrow c_1)...(x_n \leftarrow c_n)(y_1 \leftarrow d_1)...(y_p \leftarrow d_p)(for_i) =$

$vv\phi(y_1 \leftarrow d_1)...(y_p \leftarrow d_p)(for_i) = V$.

Donc, comme (2) est vérifié,

$vv\phi(y_1 \leftarrow d_1)...(y_p \leftarrow d_p)(p(t_{i1}, ..., t_{in})) = V$ car les x_i ne sont pas des variables libres de for_i puisqu'elles sont différentes des y_j.

Mais $vv\phi(x_1 \leftarrow c_1)...(x_n \leftarrow c_n)(p(x_1, ..., x_n)) =$

$vv\phi(y_1 \leftarrow d_1)...(y_p \leftarrow d_p)(p(t_{i1}, t_{i2}, ..., t_{in})) = V$ car
$c_j = \phi(y_1 \leftarrow d_1)...(y_p \leftarrow d_p)(t_{ij}(y_1, ..., y_p))$.

Donc on a (1). ♦

Proposition 5.4.3 *Si Pr est un programme trivalué et Pr' sa forme normale, on a:*

$$\forall Pr \equiv_{TH} \forall Pr'$$

Démonstration On utilise la proposition 5.4.2. ♦

5.4.3 Complétion de règles et de programmes.

Définition 5.4.3 Soit $(r_1, r_2, ..., r_m)$ un paquet de règles ayant en tête de règle des litteraux bâtis sur le même symbole de prédicat p;

La règle de ¬–completion du paquet de règles $(r_1, r_2, ..., r_m)$ est la règle:

$$\neg\text{lit} \leftarrow \neg\text{for}$$

où lit ← for est la forme normale du paquet $(r_1, r_2, ..., r_m)$.

La règle de →–completion du paquet $(r_1, r_2, ..., r_m)$ est la règle:

$$\text{lit} \rightarrow \text{for}$$

où lit ← for est la forme normale du paquet $(r_1, r_2, ..., r_m)$.

Ce n'est pas une règle trivaluée en général.

On fait la même chose pour les paquets de règles négatives en utilisant le fait que:

$$\neg\neg\text{ato} \equiv_T \text{ato}.$$

Définition 5.4.4 Le ¬–complété d'un programme trivalué Pr ne comportant que des règles positives est le programme obtenu à partir de Pr en remplaçant tous les paquets de règles par leurs formes normales et en ajoutant la ¬–complétion de ces paquets. On introduit dans le ¬–complété d'un programme trivalué Pr, le littéral $\neg p(x_1, ..., x_n)$ si p est un symbole de prédicat apparaissant dans Pr sans être en tête de règle.

Le programme obtenu se note:

$$\text{Comp}(\neg, \text{Pr}).$$

De même pour le →–complété de Pr que l'on note:

$$\mathrm{Comp}(\rightarrow, \mathrm{Pr}).$$

Remarques: - La complétion d'un programme Pr n'ayant que des règles positives au sens de Fitting est le $\rightarrow$-complété du $\neg$-complété de notre programme. En effet la formule $A \cong B$ est logiquement équivalente à: $(A \rightarrow B)\wedge(B \rightarrow A)\wedge(\neg A \rightarrow \neg B)\wedge(\neg B \rightarrow \neg A)$, et Fitting complète un programme en mettant en forme normale le paquet de règles ayant dans leurs têtes le même symbole de prédicat p.

La forme normale du paquet de règles $(r_1, r_2, ..., r_m)$ construites sur le prédicat p est la règle:

$$p(x_1, ..., x_n) \cong for'_1 \vee ... \vee for'_m$$

où for'_i est la formule close:

$$\exists y_1 ... \exists y_p (x_1 = t_{i,1} \wedge ... \wedge x_n = t_{i,n} \wedge for_i)$$

et r_i la règle: $p(t_{i,1}, ..., t_{i,n}) \leftarrow for_i$; for $i = 1, ..., m$, $y_1, ..., y_p$ étant les variables libres de for_i, $t_{i,1}, ..., t_{i,n}$ et $x_1, ..., x_n$ des variables différentes des y_j.

Fitting rajoute aux règles $p(x_1, ..., x_n) \cong for'_1 \vee ... \vee for'_m$, les littéraux $\neg q(x_1, ..., x_n)$ où q est un symbole de prédicat n'apparaissant pas en tête de règle.

- On pourrait ne pas introduire dans le $\neg$–complété d'un programme trivalué Pr, le littéral $\neg p(x_1, ..., x_n)$ si p est un symbole de prédicat apparaissant dans Pr sans être en tête de règle. Mais la notion de $\neg$–complétion nous sert à faire le lien avec la théorie de Fitting et nous en aurons donc besoin.

- En logique bivaluée, on a une seule complétion d'après la proposition suivante:

Proposition 5.4.4 *Soit Pr un progamme bivalué comportant des règles de la forme ato $\leftarrow$ for, où for est une formule n'utilisant que les symboles $\forall$, $\exists$, $\wedge$, $\vee$, alors on a:*

$$\forall Comp(\neg, Pr) \equiv_B \forall Comp(\rightarrow, Pr)$$

Démonstration $\neg A \leftarrow \neg B \equiv_B A \rightarrow B$. ♦

Par exemple, si Pr est le programme suivant:

$$r_1\text{: pair(ze)}$$

r_2: pair(s(s(x))) $\leftarrow$ pair(x)

La forme normale de r_1 et de r_2 est:

r: pair(y) $\leftarrow$ $\exists$x (y = ze$\vee$(y = s(s(x))$\wedge$pair(x)))

Le $\neg$–complété de Pr est le programme obtenu en ajoutant à r

$\neg$pair(y) $\leftarrow$ $\forall$x ($\neg$(y = ze) $\wedge$($\neg$(y = s(s(x)))$\vee\neg$pair(x))).

On aurait pu définir le prédicat $\neg$pair sans passer par le $\neg$–complété de Pr:

$\neg$pair(s(y)) $\leftarrow$ pair(y).

Dans le paragraphe suivant, on fait le lien entre notre opérateur "consequence" , celui de Van Emden et Kowalski pour les programmes sans négation, et celui de Fitting pour les programmes n'ayant que des règles positives.

5.5. LES RELATIONS ENTRE L'OPERATEUR T_{Pr}, CELUI DE VAN EMDEN ET KOWALSKI, ET CELUI DE FITTING

On rappelle la définition de l'opérateur de Van Emden et Kowalski [43] dans notre formalisme.

Definition 5.5.1 Si Pr est un programme sans négation, ayant des règles de la forme ato $\leftarrow$ for, où ato est un atome et for une formule utilisant les symboles $\forall$, $\exists$, $\wedge$, $\vee$, alors on associe l'opérateur de Van Emden et Kowalski [43] suivant à Pr:

B_{Pr}: IHB(L) $\rightarrow$ IHB(L), est défini par:

Si i $\in$ IHB(L), $B_{Pr}(i) = pos^{-1}\{$ato $\in$ her(L)/ il existe une instance close ato $\leftarrow$ for d'une règle de Pr telle que vv_i(for) = Vrai$\}$

Proposition 5.5.1 *Soit Pr un programme bivalué (ne comportant que des règles de la forme: ato $\leftarrow$for, for utilisant les symboles $\forall$, $\exists$, $\wedge$, $\vee$), et i $\in$ IHB(L).*

a) B_{Pr} est monotone sur IHB(L) et admet un plus petit point fixe noté pppf(B_{Pr}). Si Pr est un programme défini bivalué, ayant des règles de la forme ato $\leftarrow$ $ato_1 \wedge ato_2 \wedge ... \wedge ato_n$, alors B_{Pr} est continu et pppf(B_{Pr}) = $B_{Pr}\uparrow\omega$.

b) $B_{Pr}(i) \leq i \Leftrightarrow$ i est un modèle de Herbrand bivalué de Pr.

c) ppmb(Pr) = pppf(B_{Pr}).
(ppmb est le plus petit modèle de Herbrand bivalué).

Démonstration a) B_{Pr} est monotone sur IHB(L) pour l'ordre $i \leq j \Leftrightarrow$ pos(i) $\subseteq$ pos(j). Comme IHB(L) satisfait les conditions du théorème de Birkhoff-Tarski, il admet un plus petit point fixe, noté pppf(B_{Pr}). On démontre que si Pr est un programme défini bivalué et X est une partie dirigée de IHB(L), où L est le langage sous-jacent à Pr, on a B_{Pr}(sup(X)) = sup(B_{Pr}(X)). On a dans ce cas pppf(B_{Pr}) = $B_{Pr}\uparrow\omega$ =

$$\cup_{m \in \mathbb{N}} B_{Pr}{}^{m}(\emptyset).$$

Si on a démontré b), on a immédiatement c).

c) Le plus petit point fixe de B_{Pr} est aussi le plus petit modèle de Herbrand bivalué de Pr, puisque les modèles de Pr sont exactement les post-points fixes de B_{Pr} ($B_{Pr}(i) \leq i$) et le plus petit point fixe d'un opérateur monotone est son plus petit post-point fixe (par induction transfinie, $B_{Pr}\uparrow\alpha \leq i$, pour tout modèle i de Pr (post-point fixe de B_{Pr}) et tout ordinal α).

b) Soit i telle que $B_{Pr}(i) \leq i$ et

$$\text{ato} \leftarrow \text{for}$$

une instance close d'une règle de Pr.

Montrons que vv_i(for) = Vrai implique que vv_i(ato) = Vrai.

vv_i(for) = Vrai implique que ato $\in B_{Pr}(i)$ par définition de B_{Pr}, donc ato $\in$ pos(B_{Pr}(i)) et ato$\in$ pos(i) puisque $B_{Pr}(i) \leq i$.

Par conséquent vv_i(ato) = Vrai.

- Réciproquement, soit i un modèle de Herbrand bivalué de Pr.

Soit ato $\in$ pos(B_{Pr}(i)). Montrons que ato $\in$ pos(i).

ato $\in$ pos(B_{Pr}(i)) $\Leftrightarrow$ Il existe une instance close d'une règle de Pr:

$$\text{ato} \leftarrow \text{for}$$

telle que vv_i(for) = Vrai.

Puisque i est un modèle bivalué de Pr, on a: vv_i(ato $\leftarrow$ for) = Vrai;
par conséquent, vv_i(ato) = Vrai et ato $\in$ pos(i). ♦

La proposition suivante fait le lien entre notre opérateur et celui de Van Emden et Kowalski quand Pr est un programme comportant des règles de la forme ato $\leftarrow$ for, for comportant les symboles $\forall$, $\exists$, $\wedge$, $\vee$.

Proposition 5.5.2 *a)* $B_{Pr} = pos^{-1} o T_{Pr} o pos$

b) $pppf(B_{Pr}) = pos^{-1}(pppf(T_{Pr})) = ppmb(Pr) = pos^{-1}(ppmt(Pr))$

c) Si Pr est un programme défini bivalué, $pppf(B_{Pr}) = pos^{-1}(T_{Pr}\uparrow\omega)$

$$= (\cup_{m \in \mathcal{N}} pos^{-1}(T_{Pr}{}^{m}(\emptyset)).$$

Démonstration a) On utilise le fait que si for comprend les symboles $\forall$, $\exists$, $\wedge$, $\vee$, alors vv_i(for) = Vrai $\Leftrightarrow$ $vv_{pos(i)}$(for) = Vrai, par induction sur la formule for.
Pr est un programme bivalué dont les règles sont de la forme:

ato $\leftarrow$ for, avec for ne contenant que les symboles ($\forall$, $\exists$, $\wedge$, $\vee$).

T_{Pr}(pos(i)) = {ato $\in$ Her(L)/ il existe une instance close ato $\leftarrow$ for d'une règle de Pr telle que $vv_{pos(i)}$(for) = Vrai}.

Si $vv_{pos(i)}$(for) = Vrai, alors vv_i(for) = Vrai par monotonie puisque pos(i) $\subseteq$ i.
Si vv_i(for) = Vrai, alors $vv_{pos(i)}$(for) = Vrai par induction sur for:
- si for = ato, alors vv_i(ato) = Vrai $\Leftrightarrow$ ato $\in$ i $\Leftrightarrow$ ato $\in$ pos(i);
- de même si for = f$\wedge$h, f$\vee$h, $\exists$x f, $\forall$x f, par induction.

b) On montre que: $B_{Pr}\uparrow\alpha = pos^{-1}(T_{Pr}\uparrow\alpha)$ pour tout ordinal α, par induction transfinie avec a).

$-\ \alpha = 0$; $B_{Pr}(\emptyset) = pos^{-1}(T_{Pr}(\emptyset))$.

- Si α est un ordinal successeur, alors $B_{Pr}\uparrow\alpha = B_{Pr}(pos^{-1}(T_{Pr}\uparrow\alpha-1))$ par induction.

Donc $B_{Pr}\uparrow\alpha = pos^{-1}(\{ato \in her(L)/$ il existe une instance close $ato \leftarrow for$ d'une règle de Pr telle que $vv_{pos^{-1}(T_{Pr}\uparrow\alpha-1)}(for) = Vrai\})$

$B_{Pr}\uparrow\alpha = pos^{-1}(\{ato \in her(L)/$ il existe une instance close $ato \leftarrow for$ d'une règle de Pr telle que $vv_{(T_{Pr}\uparrow\alpha-1)}(for) = Vrai\})$,

car $vv_{pos^{-1}(T_{Pr}\uparrow\alpha-1)}(for) = Vrai \Leftrightarrow vv_{(T_{Pr}\uparrow\alpha-1)}(for) = Vrai$ puisque for comprend les symboles $\forall$, $\exists$, $\wedge$, $\vee$.

Par conséquent $B_{Pr}\uparrow\alpha = pos^{-1}(T_{Pr}\uparrow\alpha)$ puisque Pr est un programme bivalué.

- Si α est un ordinal limite, $\alpha = \sup\{\beta, \beta<\alpha\}$, alors:

$B_{Pr}\uparrow\alpha = \sup_{IHB(L)}\{pos^{-1}(T_{Pr}\uparrow\beta), \beta<\alpha\}$ par induction.

Comme Pr est un programme bivalué, $T_{Pr}\uparrow\beta$ ne contient que des atomes positifs, et l'on a $\sup_{IHB(L)}\{pos^{-1}(T_{Pr}\uparrow\beta), \beta<\alpha\} = pos^{-1}(\sup_{IHT(L)}\{T_{Pr}\uparrow\beta, \beta<\alpha\}$.

$B_{Pr}\uparrow\alpha = pos^{-1}(\sup_{IHT(L)}\{T_{Pr}\uparrow\beta, \beta<\alpha\}) = pos^{-1}(T_{Pr}\uparrow\alpha)$.

Donc $pppf(B_{Pr}) = pos^{-1}(pppf(T_{Pr}))$ puisqu'il existe α tel que $B_{Pr}\uparrow\alpha = pppf(B_{Pr})$ et β tel que $T_{Pr}\uparrow\beta = pppf(T_{Pr})$. ♦

c) Si Pr est défini bivalué, il est défini trivalué et T_{Pr} est continu donc $T_{Pr}\uparrow\omega = pppf(T_{Pr})$ et $pppf(B_{Pr}) = pos^{-1}(T_{Pr}\uparrow\omega) =$
$pos^{-1}(\cup_{m\in\mathbb{N}} (T_{Pr}{}^{m}(\emptyset))$.

Mais $pppf(B_{Pr}) = \cup_{m\in\mathbb{N}} B_{Pr}{}^{m}(i)$, avec $i \in IHB(L)$ telle que $pos(i) = \emptyset$.
Comme $B_{Pr} = pos^{-1} \circ T_{Pr} \circ pos$, $B_{Pr}{}^{m}(i) = pos^{-1}(T_{Pr}{}^{m}(\emptyset))$ et $pppf(B_{Pr}) = \cup_{m\in\mathbb{N}}\ pos^{-1}(T_{Pr}{}^{m}(\emptyset))$.

On étudie maintenant les relations entre notre opérateur et celui de Fitting.

Pour cela, on donne la définition de l'opérateur introduit par Fitting [9] dans notre formalisme:

Définition 5.5.2 Soit Pr un programme trivalué n'ayant que des règles positives de la forme ato $\leftarrow$ for, où for comprend les symboles ($\forall$, $\exists$, $\neg$, $\wedge$, $\vee$, $\Leftrightarrow$, $\Rightarrow$).

F_{Pr} est l'application suivante:

F_{Pr}: IHT(L) $\rightarrow$ IHT(L).

Si i $\in$ IHT(L), alors:

> $F_{Pr}(i)$ = {ato $\in$ her(L)/ il existe une instance close ato $\leftarrow$ for d'une règle de Pr telle que vv_i(for) = Vrai} $\cup$
>
> {$\neg$ato $\in$ $\neg$her(L)/ pour toute instance close ato $\leftarrow$ for de Pr, vv_i(for) = Faux}.

Remarque: Dans $F_{Pr}(i)$ figure toutes les instances closes des littéraux $\neg p(x_1, ..., x_n)$, où p est un symbole de prédicat figurant dans Pr sans être en tête de règle. C'est pourquoi dans le $\neg$-complété de Pr, doit figurer aussi ce littéral $\neg p(x_1, ..., x_n)$, où p est un symbole de prédicat figurant dans Pr sans être en tête de règle pour obtenir les points fixes de F_{Pr} comme modèles du complété.

Proposition 5.5.3 *F_{Pr} est monotone sur IHT(L) pour l'ordre $i \subseteq j$. Comme IHT(L) satisfait les conditions du théorème de Birkhoff-Tarski, F_{Pr} admet un plus petit point fixe noté pppf(F_{Pr}).*

Démonstration résulte de la propriété de monotonie (proposition 5.2.1.1)

La proposition suivante fait le lien entre notre opérateur et celui de Fitting:

Proposition 5.5.4 *1) $T_{Comp(\neg, Pr)} = F_{Pr}$.*

2) a) Une interprétation de Herbrand trivaluée i est un modèle de Comp($\neg$, Pr) si et seulement si $F_{Pr}(i) \subseteq i$.

b) Une interprétation de Herbrand trivaluée i est un modèle de Comp($\rightarrow$, Comp($\neg$, Pr)) si et seulement si $F_{Pr}(i) = i$.

3) pppf(F_{Pr}) = ppmt(Comp(→, Comp(¬, Pr))) = ppmt(Comp(¬, Pr)).

Démonstration 1) On obtient, si Comp(¬, Pr) désigne le ¬-complété de Pr,

$$T_{Comp(\neg, Pr)} = F_{Pr}$$

En effet:

$T_{Comp(\neg, Pr)}(i)$ = {lit ∈ Her(L)/ il existe une instance close lit ← for d'une règle de Comp(¬, Pr) telle que vv_i(for) = Vrai}.

si cet ensemble ne contient pas un atome et sa négation,

$T_{Comp(\neg, Pr)}(i)$ = Contra sinon.

$F_{Pr}(i)$ = {ato ∈ her(L)/ il existe une instance close ato ← for d'une règle de Pr telle que vv_i(for) = Vrai}

∪ {¬ato ∈ ¬her(L)/ pour toute instance close ato ← for de Pr, vv_i(for) = Faux}

=

$T_{Pr}(i)$ ∪ {¬ato ∈ ¬her(L)/ pour toute instance close ato ← for de Pr, vv_i(for) = Faux}

car Pr n'a que des règles positives.

- On commence par démontrer le lemme 5.5.1 suivant:

Lemme 5.5.1 *Si Pr' est la forme normale d'un programme trivalué Pr, alors on a $T_{Pr} = T_{Pr'}$.*

Démonstration a) Soit Pr un progamme trivalué.

Si Pr' est la forme normale de Pr, alors ∀Pr $\equiv_{TH}$∀Pr' (Proposition 5.4.3). Soit $(\alpha_1, ..., \alpha_n) \in UNI(L)^n$ tel que $lit(\alpha_1, ..., \alpha_n) \in T_{Pr}(i)$.

Alors il existe $(t_1, ..., t_p) \in UNI(L)^p$ tel que:

$$lit(\alpha_1, ..., \alpha_n) \leftarrow for_i(t_1, ..., t_p)$$

soit une instance close de Pr avec $vv_i(for_i(t_1, ..., t_p))$ = Vrai.

Cette instance close provient d'une règle de Pr de la forme:

$lit(t_{i1}(y_1, ..., y_p), ..., t_{in}(y_1, ..., y_p)) \leftarrow for_i(y_1, ..., y_p)$

avec $\alpha_j = t_{ij}(t_1, ..., t_p)$.

Par conséquent:

$$lit(x_1, ..., x_n) \leftarrow \vee_i(\exists y_1 ... \exists y_p\, (x_1 = t_{i1}(y_1, ..., y_p) \wedge \\ ... \wedge x_n = t_{in}(y_1, ..., y_p) \wedge for_i(y_1, ..., y_p)))$$

est une règle de Pr', forme normale de Pr et il existe une instance close de cette règle:

$$lit(\alpha_1, ..., \alpha_n) \leftarrow \vee_i(\exists y_1 ... \exists y_p\, (\alpha_1 = t_{i1}(y_1, ..., y_p) \wedge \\ ... \wedge \alpha_n = t_{in}(y_1, ..., y_p) \wedge for_i(y_1, ..., y_p)))$$

telle que:

$$vv_i(\exists y_1 ... \exists y_p\, (\alpha_1 = t_{i1}(y_1, ..., y_p) \wedge ... \wedge \alpha_n = t_{in}(y_1, ..., y_p) \wedge \\ for_i(y_1, ..., y_p))) = Vrai$$

puisqu'il existe $(t_1, ..., t_p) \in UNI(L)^p$ tel que $\alpha_j = t_{ij}(t_1, ..., t_p)$ et $vv_i(for_i(t_1, ..., t_p)) = Vrai$.

Par conséquent $lit(\alpha_1, ..., \alpha_n) \in T_{Pr'}(i)$.

De même, $lit(\alpha_1, ..., \alpha_n) \in T_{Pr'}(i)$ implique que $lit(\alpha_1, ..., \alpha_n) \in T_{Pr}(i)$.

Par conséquent, $lit(\alpha_1, ..., \alpha_n) \in T_{Pr}(i) \Leftrightarrow lit(\alpha_1, ..., \alpha_n) \in T_{Pr'}(i)$. (*)

Donc $T_{Pr}(i) = T_{Pr'}(i)$, puisque le cas $T_{Pr}(i) = Contra$ se résout avec (*).

Par conséquent:

$$T_{Pr} = T_{Pr'}$$

Remarque: On a de même: $B_{Pr} = B_{Pr'}$, si Pr est un programme bivalué et Pr' sa forme normale. ♦

- On démontre ensuite le lemme 5.5.2 suivant:

Lemme 5.5.2 *$T_{Comp(\neg, Pr)}(i) \neq Contra$ si $i \in IHT(L)$.*

Démonstration Comp(¬, Pr) s'obtient en ajoutant aux règles de la forme normale de Pr telles que:

$$ato(x_1, ..., x_n) \leftarrow \vee_i(\exists y_1...\exists y_p\,(x_1 = t_{i1}\wedge...\wedge x_n = t_{in}\wedge for_i))$$

les règles:

$$\neg ato(x_1, ..., x_n) \leftarrow \neg(\vee_i(\exists y_1...\exists y_p\,(x_1 = t_{i1}\wedge...\wedge x_n = t_{in}\wedge for_i)))$$

S'il existe $(\alpha_1, ..., \alpha_n) \in UNI(L)^n$ tel que $ato(\alpha_1, ..., \alpha_n) \in T_{Comp(\neg, Pr)}(i)$ et $\neg ato(\alpha_1, ..., \alpha_n) \in T_{Comp(\neg, Pr)}(i)$, alors si

$$for' = \vee_i(\exists y_1...\exists y_p\,(\alpha_1 = t_{i1}\wedge...\wedge\alpha_n = t_{in}\wedge for_i)),$$

on aurait à la fois: $vv_i(for')$ = Vrai et Faux ce qui est impossible. ♦

Pour les symboles $\neg ato(x_1, ..., x_n)$ dans Comp(¬, Pr) où $ato(x_1, ..., x_n)$ ne figure pas en tête de règle, les instances closes de $\neg ato(x_1, ..., x_n)$ appartiennent à {lit ∈ Her(L)/ il existe une instance close lit ← for d'une règle de Comp(¬, Pr) telle que $vv_i(for)$ = Vrai}, mais il ne peut y avoir dans cet ensemble une instance close de $ato(x_1, ..., x_n)$ et sa négation puisque ato ne figure pas en tête de règle.

On est maintenant en mesure de prouver que $T_{Comp(\neg, Pr)} = F_{Pr}$.

Soit lit $\in T_{Comp(\neg, Pr)}(i)$.

- Ou bien lit = ato et lit $\in T_{Pr'}(i) = T_{Pr}(i)$ d'après le lemme 5.5.1, ce qui entraîne que lit $\in F_{Pr}(i)$.

- Ou bien lit = $\neg ato(\alpha_1, ..., \alpha_n) \in T_{Comp(\neg, Pr)}(i)$ pour un $(\alpha_1, ..., \alpha_n) \in UNI(L)^n$.

- Soit il existe une instance close de Comp(¬, Pr):

$$\neg ato(\alpha_1, ..., \alpha_n) \leftarrow \neg for'$$

telle que $vv_i(for')$ = Faux, avec for' =

$$\vee_i(\exists y_1...\exists y_p\,(\alpha_1 = t_{i1}(y_1,, ..., y_p) \wedge ... \wedge \alpha_n = t_{in}(y_1, ..., y_p) \wedge$$
$$for_i(y_1, ..., y_p))).$$

Par conséquent $\forall$ i = 1, ..., m et $\forall(u_1, ..., u_p) \in UNI(L)^p$,
si $\alpha_j = t_{ij}(u_1, ..., u_p)$, pour tout j = 1, ..., n, alors $vv_i(for_i(u_1, ..., u_p))$ = Faux.

Soit $ato(\alpha_1, ..., \alpha_n)$ la tête d'une instance close d'une règle de Pr; elle provient de:

$$ato(t_{i1}(y_1, ..., y_p), ..., t_{in}(y_1, ..., y_p)) \leftarrow for_i(y_1, ..., y_p).$$

Soit $(u_1, ..., u_p) \in UNI(L)^p$ tel que:

$$t_{ij}(u_1, ..., u_p) = \alpha_j, \text{ pour tout } j = 1, ..., n.$$

Alors $vv_i(for_i(u_1, ..., u_p))$ = Faux, puisque $vv_i(for')$ = Faux; par conséquent:

$$\neg ato(\alpha_1, ..., \alpha_n) \in F_{Pr}(i).$$

- Soit $\neg ato(x_1, ..., x_n) \in Comp(\neg, Pr)$ car ato est un symbole intervenant dans le programme sans figurer en tête de règle.

Dans ce cas, $\neg ato(\alpha_1, ..., \alpha_n) \in F_{Pr}(i) = \{ato \in her(L)/$ il existe une instance close ato $\leftarrow$ for d'une règle de Pr telle que $vv_i(for)$ = Vrai$\} \cup \{\neg ato \in \neg her(L)/$ pour toute instance close ato $\leftarrow$ for de Pr, $vv_i(for)$ = Faux$\}$ car $\neg ato(\alpha_1, ..., \alpha_n) \in \{\neg ato \in \neg her(L)/$ pour toute instance close ato $\leftarrow$ for de Pr, $vv_i(for)$ = Faux$\}$.

Réciproquement, soit lit $\in F_{Pr}(i)$. Alors:

- soit lit $\in T_{Pr}(i) = T_{Pr'}(i) \subseteq T_{Comp(\neg, Pr)}(i)$;

- soit lit = $\neg ato(\alpha_1, ..., \alpha_n)$ pour un $(\alpha_1, ..., \alpha_n) \in UNI(L)^n$;

alors pour toute instance close:

$$ato(t_{i1}(u_1, ..., u_p), ..., t_{in}(u_1, ..., u_p)) \leftarrow for_i(u_1, ..., u_p)$$

telle que: $t_{ij}(u_1, ..., u_p) = \alpha_j$, pour tout j = 1, ..., n,

on a $vv_i(for_i(u_1, ..., u_p))$ = Faux;

par conséquent:

$$vv_i(\vee_i(\exists y_1 ... \exists y_p\,(\alpha_1 = t_{i1}(y_1, ..., y_p) \wedge ... \wedge \alpha_n = t_{in}(y_1, ..., y_p) \wedge \\ for_i(y_1, ..., y_p)))) = \text{Faux}$$

et:

$\neg ato(\alpha_1, ..., \alpha_n) \in T_{Comp(\neg, Pr)}(i)$. ♦

2) a) Une interprétation de Herbrand trivaluée i est un modèle de Comp(¬, Pr) $\Leftrightarrow F_{Pr}(i) \subseteq i$ est une conséquence immédiate de 1) et du théorème 5.3.1 (2a).
En effet:
$F_{Pr}(i) \subseteq i \quad \Leftrightarrow T_{Comp(\neg, Pr)}(i) \subseteq i$ (1) ci-dessus)
$\Leftrightarrow$ i est un modèle de Comp(¬, Pr) (Théorème 5.3.1 (2a)).

b) Une interprétation de Herbrand trivaluée i est un modèle de Comp(→, Comp(¬, Pr)) $\Leftrightarrow F_{Pr}(i) = i$ peut se démontrer de deux manières.

La première utilise l'argument de Fitting:
i est un modèle de Pr (ce qui signifie un modèle de son complété) si et seulement si i est un point fixe de F_{Pr}
et le fait que le complété au sens de Fitting d'un programme s'obtient en prenant le →-complété du ¬-complété du programme (Remarque définition 5.4.4).

On obtient alors que:
$i \in IHT(L)$ est un modèle de Comp(→, Comp(¬, Pr)) $\Leftrightarrow F_{Pr}(i) = i$.

La deuxième manière résultera de la proposition 5.5.5.

3) $pppf(F_{Pr}) = ppmt(Comp(\rightarrow, Comp(\neg, Pr)))$ est une conséquence de 2) b) de la proposition 5.5.4.
$pppf(F_{Pr}) = ppmt(Comp(\neg, Pr))$ est une conséquences de 2) a) et du fait que $F_{Pr}\uparrow\alpha \subseteq i$ pour tout modèle i de Comp(¬, Pr) pour tout ordinal α (post-point fixe de F_{Pr}), par induction transfinie. ♦

La proposition suivante permet de démontrer d'une autre manière le 2)b).

Proposition 5.5.5 Si Pr est consistant, alors Comp(→, Pr) est consistant et l'on a:
1) Une interprétation $i \in IHT(L)$ est un modèle trivalué de Comp(→, Pr) $\Leftrightarrow T_{Pr}(i) = i$.
2) $ppmt(Pr) = ppmt(Comp(\rightarrow, Pr)) = pppf(T_{Pr})$.

Démonstration On remarque d'abord que le 2) de la proposition est en partie une conséquence du 1): ppmt(Comp($\rightarrow$, Pr)) = pppf(T_{Pr}) et ensuite une conséquence du théorème 5.3.1 2) b): pppf(T_{Pr}) = ppmt(Pr).

On remarque aussi que le 2) b) de la proposition 5.5.4 peut être prouvé avec le 1) de cette proposition 5.5.5:

$F_{Pr}(i) = i \Leftrightarrow T_{Comp(\neg,\ Pr)}(i) = i$ (proposition 5.5 4))
$\Leftrightarrow$ i est un modèle de Comp($\rightarrow$, Comp($\neg$, Pr)) (par le 1) de la proposition 5.5.5).

Il reste à montrer le 1) de la proposition:

$i \neq$ Contra et $T_{Pr}(i) = i \Leftrightarrow$ i est un modèle de Comp($\rightarrow$, Pr).

Ceci prouvera aussi que si Pr est consistant, alors Comp($\rightarrow$, Pr) l'est aussi puisque si Pr est consistant alors pppf(T_{Pr}) $\neq$ Contra selon le théorème 5.3.1 2), et pppf(T_{Pr}) sera aussi un modèle de Comp($\rightarrow$, Pr).

Montrons que $i \neq$ Contra et $T_{Pr}(i) = i$ implique que i est un modèle de Comp($\rightarrow$, Pr).

Soit r une règle de Comp($\rightarrow$, Pr).

- Ou bien r est de la forme:

$$\text{lit} \leftarrow \text{for}$$

et r est une règle de Pr'.

Comme $T_{Pr}(i) = T_{Pr'}(i)$, $T_{Pr'}(i) \subseteq i$, donc i est un modèle de Pr' et enfin $vv_i(\text{lit} \leftarrow \text{for})$ = Vrai.

- Ou bien r est de la forme:

$$\text{lit} \rightarrow \text{for}.$$

On montre que: $vv_i(\text{lit} \rightarrow \text{for})$ = Vrai pour toute instance close de r; r est de la forme:

$$\text{lit}(x_1, \ldots, x_n) \rightarrow \vee_i(\exists y_1 \ldots \exists y_p\, (x_1 = t_{i1}(y_1, \ldots, y_p) \wedge \ldots \wedge x_n = t_{in}(y_1, \ldots, y_p) \wedge \text{for}_i(y_1, \ldots, y_p))).$$

Si $(t_1, ..., t_n) \in \mathrm{UNI}(L)^n$, alors:

$$\mathrm{lit}(t_1, ..., t_n) \rightarrow \vee_i(\exists y_1 ... \exists y_p (t_1 = t_{i1}(y_1, ..., y_p) \wedge ... \wedge t_n = t_{in}(y_1, ..., y_p) \wedge \mathrm{for}_i(y_1, ..., y_p)))$$

est une instance close de r.

Le seul cas à envisager est $vv_i(\mathrm{lit}(t_1, ..., t_n)) = \mathrm{Vrai}$, puisque $i \subseteq T_{Pr}(i) = T_{Pr'}(i)$, on a:

$$\mathrm{lit}(t_1, ..., t_n) \in T_{Pr'}(i).$$

Par conséquent, il existe une instance close:

$$\mathrm{lit}(t_1, ..., t_n) \leftarrow \vee_i(\exists y_1 ... \exists y_p (t_1 = t_{i1}(y_1, ..., y_p) \wedge ... \wedge t_n = t_{in}(y_1, ..., y_p) \wedge \mathrm{for}_i(y_1, ..., y_p)))$$

d'une règle de Pr' telle que $vv_i(\mathrm{for}') = \mathrm{Vrai}$ avec:

$$\mathrm{for}' = \vee_i(\exists y_1 ... \exists y_p (t_1 = t_{i1}(y_1, ..., y_p) \wedge ... \wedge t_n = t_{in}(y_1, ..., y_p) \wedge \mathrm{for}_i(y_1, ..., y_p))).$$

Par conséquent, $vv_i(\mathrm{lit}(t_1, ..., t_n) \rightarrow \mathrm{for}') = \mathrm{Vrai}$.

Si $vv_i(\mathrm{lit}(t_1, ..., t_n)) \neq \mathrm{Vrai}$, alors $vv_i(\mathrm{lit}(t_1, ..., t_n) \rightarrow \mathrm{for}') = \mathrm{Vrai}$.

Réciproquement, si i est un modèle de Herbrand de Comp($\rightarrow$, Pr), alors i est un modèle de Herbrand de Pr' si Pr' est la forme normale de Pr et un modèle de Pr puisque $\forall Pr \equiv_{TH} \forall Pr'$.

Par conséquent $T_{Pr}(i) \subseteq i$ et $i \neq \mathrm{Contra}$.

Par ailleurs, $vv_i(\mathrm{lit} \rightarrow \mathrm{for}') = \mathrm{Vrai}$ pour toute instance close $\mathrm{lit} \rightarrow \mathrm{for}'$ d'une règle de Comp($\rightarrow$, Pr).

Donc, si $\mathrm{lit} \in i$, $vv_i(\mathrm{for}') = \mathrm{Vrai}$ et $\mathrm{lit} \in T_{Pr'}(i) = T_{Pr}(i)$ puisqu'il existe une instance close:

$$\mathrm{lit} \leftarrow \mathrm{for}'$$

de Pr' telle que $vv_i(for') = Vrai$ et $T_{Pr'}(i) \neq Contra$ puisque $T_{Pr'}(i) = T_{Pr}(i) \subseteq i \neq Contra$.

Donc $i \subseteq T_{Pr}(i)$.

Si Pr est consistant au sens de Herbrand, $pppf(T_{Pr}) \neq Contra$ est un modèle de Comp(→, Pr) qui est donc aussi consistant.

On a:

$pppf(T_{Pr}) \subseteq i$, $\forall i$ modèle de Pr, donc $pppf(T_{Pr}) \subseteq i$, $\forall i$ modèle de Comp(→, Pr) et:

$$\underline{pppf(T_{Pr}) = ppmt(Pr) = ppmt(Comp(\rightarrow, Pr)).}\ \blacklozenge$$

On termine ce paragraphe par deux propositions concernant les programmes bivalués.

Proposition 5.5.6 *Soit Pr un programme bivalué (ne comportant que des règles de la forme ato ← for, for utilisant les symboles ∀, ∃, ∧, ∨, i ∈ IHB(L).*

a) $B_{Pr}(i) = i \Leftrightarrow$ *i est un modèle bivalué de Comp(¬, Pr).*
$B_{Pr}(i) = i \Leftrightarrow$ *i est un modèle bivalué de Comp(→, Pr).*

b) $ppmb(Pr) = ppmb(Comp(\neg, Pr)) = ppmb(Comp(\rightarrow, Pr)) =$
$$pppf(B_{Pr}) = pos^{-1}(pppf(T_{Pr})) = pos^{-1}(ppmt(Pr)).$$
(ppmb est le plus petit modèle de Herbrand bivalué).

Remarque: Pour cette proposition, on doit supprimer du ¬-complété Comp(¬, Pr), $\neg ato(x_1, ..., x_n)$ où ato est un symbole de prédicat de Pr n'apparaissant pas en tête de règle de Pr.

Démonstration a) Comme $\forall Comp(\neg, Pr) \equiv_B \forall Comp(\rightarrow, Pr)$ puisque
$$ato \rightarrow for \equiv_B \neg ato \leftarrow \neg for,$$

montrons que: i est un modèle bivalué de Comp(¬, Pr) $\Leftrightarrow B_{Pr}(i) = i$.

- $B_{Pr}(i) = i$ implique que $B_{Pr}(i) \leq i$, ce qui implique que i est un modèle bivalué de Pr' si Pr' est la forme normale de Pr, puisque $B_{Pr} = B_{Pr'}$.

Montrons que i est un modèle bivalué des instances closes:

$$\neg ato \leftarrow \neg for$$

des règles de Comp($\neg$, Pr) n'étant pas dans Pr'.

Le seul cas à envisager est $vv_i(\neg for) =$ Vrai. On doit alors montrer que $vv_i(\neg ato) =$ Vrai. Si $vv_i(\neg ato) =$ Faux, alors $vv_i(ato) =$ Vrai et comme $B_{Pr}(i) = i$, ato $\in B_{Pr}(i)$. Les règles de Comp($\neg$, Pr) proviennent de:

$$ato(x_1, ..., x_n) \leftarrow \vee_i(\exists y_1 ... \exists y_p\, (x_1 = t_{i1}(y_1, ..., y_p) \wedge \\ ... \wedge x_n = t_{in}(y_1, ..., y_p) \wedge for_i(y_1, ..., y_p)));$$

Donc:

$$\neg ato(\alpha_1, ..., \alpha_n) \leftarrow \neg(\vee_i(\exists y_1 ... \exists y_p\, (\alpha_1 = t_{i1}(y_1, ..., y_p) \wedge ... \\ \wedge \alpha_n = t_{in}(y_1, ..., y_p) \wedge for_i(y_1, ..., y_p))))$$

est l'instance close considérée, avec for =

$$\vee_i(\exists y_1 ... \exists y_p\, (x_1 = t_{i1}(y_1, ..., y_p) \wedge \\ ... \wedge x_n = t_{in}(y_1, ..., y_p) \wedge for_i(y_1, ..., y_p)));$$

on aurait à la fois $vv_i(for(\alpha_1, ..., \alpha_n)) =$ Vrai et $vv_i(for(\alpha_1, ..., \alpha_n)) =$ Faux ce qui est impossible.

Donc $B_{Pr}(i) = i$ implique que i est un modèle bivalué de Comp($\neg$, Pr).

- Réciproquement, si i est un modèle de Herbrand bivalué de Comp($\neg$, Pr), alors i est un modèle bivalué de Pr', donc de Pr puisque $\forall Pr \equiv_{BH} \forall Pr'$, donc $B_{Pr}(i) \leq i$ et i est un modèle bivalué de chaque instance close:

$$\neg ato \leftarrow \neg for$$

d'une règle de Comp($\neg$, Pr) qui n'est pas dans Pr'.

On veut montrer que $pos(i) \subseteq pos(B_{Pr}(i))$, pour avoir $i \leq B_{Pr}(i)$.

Soit ato $\notin pos(B_{Pr}(i))$; alors pour toute instance close:

$$ato \leftarrow for$$

ayant ato en tête de règle, $vv_i(for) =$ Faux;

puisque $vv_i(\neg ato \leftarrow \neg for)$ = Vrai, $vv_i(\neg ato)$ = Vrai et ato $\notin$ pos(i);

b) $pppf(B_{Pr}) \subseteq i$, pour tout modèle bivalué i de Pr parce qu'on peut montrer par induction transfinie que $B_{Pr}\uparrow\alpha \subseteq i$, pour tout modèle bivalué i de Pr. Donc en particulier, $pppf(B_{Pr}) \subseteq i$, pour tout modèle bivalué i de Comp(¬, Pr).

Donc $pppf(B_{Pr})$ = ppmb(Pr) = ppmb(Comp(¬, Pr))

= ppmb(Comp(→, Pr)) = $pos^{-1}(pppf(T_{Pr}))$ (proposition 5.5.2 b))

= $pos^{-1}(ppmt(Pr))$ (théorème 5.3.1 b)). ♦

Proposition 5.5.7 *Soit Pr un programme défini bivalué;.*

ppmt(Comp(¬, Pr)) = pos(ppmb(Pr)) ∪ ¬neg(pgmb(Comp(¬, Pr))).

(pgmb est le plus grand modèle bivalué).

Remarque: Pour cette proposition, on doit avoir dans le ¬-complété Comp(¬, Pr), $\neg ato(x_1, ..., x_n)$ où ato est un symbole de prédicat de Pr n'apparaissant pas en tête de règle de Pr.

Démonstration Il revient au même de montrer que:

$$pppf(T_{Comp(\neg, Pr)}) = pos(pppf(B_{Pr})) \cup \neg neg(pgpf(B_{Pr})) \text{ (1)},$$

où: ¬neg: IHT(L) → IHT(L), i → ¬neg(i) = {¬ato ∈ i},
pos: IHT(L) → IHT(L), i → pos(i) = {ato ∈ i}.
pgpf est le plus grand point fixe.

car $pgpf(B_{Pr})$ = pgmb(Comp(¬, Pr)) selon la proposition 5.5.6 (a), $pppf(T_{Comp(\neg, Pr)})$ = ppmt(Comp(¬, Pr)), (théorème 5.3.1 (2 b))) et $pppf(B_{Pr})$ = ppmb(Pr), (proposition 5.5.1 (c)).

On définit pos^{-1}: pos(IHT(L)) → IHB(L),
pos(i) → pos(i) ∪ {¬ato, ato $\notin$ pos(i)}, et:
$\neg neg^{-1}$: ¬neg(IHT(L)) → IHB(L),
¬neg(i) → ¬neg(i) ∪ {ato/ ¬ato $\notin$ ¬neg(i)}.

Pour montrer (1), on montre que pour tout ordinal α,

$$T_{Comp(\neg, Pr)}\uparrow\alpha = pos(B_{Pr}\uparrow\alpha) \cup \neg neg(B_{Pr}\downarrow\alpha) \text{ (2)}.$$

Puisqu'il existe un ordinal α tel que:
$pppf(T_{Comp(\neg, Pr)}) = T_{Comp(\neg, Pr)}\uparrow\alpha$, un ordinal β tel que:
$pppf(B_{Pr}) = B_{Pr}\uparrow\beta$ et un ordinal γ tel que:
$pgpf(B_{Pr}) = B_{Pr}\downarrow\gamma$, on aura le résultat en prenant le plus grand des trois ordinaux.

Pour prouver (2), on commence par établir les trois lemmes suivants:

Lemme 5.5.3 *Soient i $\in$ IHT(L) et Pr un programme défini bivalué. On a alors:*

a) $pos(T_{Comp(\neg, Pr)}(i)) = pos(T_{Comp(\neg, Pr)}(pos(i)))$
b) $\neg neg(T_{Comp(\neg, Pr)}(i)) = \neg neg(T_{Comp(\neg, Pr)}(\neg neg(i)))$

Démonstration a) Si Pr' est la forme normale de Pr, alors

$$\begin{aligned}
pos(T_{Comp(\neg, Pr)}(i)) &= \{ato/ \text{ il existe une instance close de Pr'},\\
&\qquad ato \leftarrow for \text{ telle que } vv_i(for) = Vrai\}\\
&= T_{Pr'}(i) \text{ (puisque Pr est défini bivalué)}\\
&= T_{Pr}(i) \text{ (puisque } T_{Pr'}(i) = T_{Pr}(i))\\
&= \{ato/ \text{ il existe une instance close de Pr},\\
&\qquad ato \leftarrow for \text{ telle que } vv_i(for) = Vrai\}.
\end{aligned}$$

Comme Pr est défini bivalué, $vv_i(for) = Vrai \Leftrightarrow vv_{pos(i)}(for) = Vrai$, (par induction sur for).

Donc $T_{Pr}(i) = T_{Pr}(pos(i)) = T_{Pr'}(pos(i)) = pos(T_{Comp(\neg, Pr)}(pos(i)))$.

$$\begin{aligned}
\neg neg(T_{Comp(\neg, Pr)}(i)) = \{&\neg ato/ \text{ il existe une instance close d'une règle}\\
&\text{de } Comp(\neg, Pr), \neg ato \leftarrow \neg for' \text{ avec } vv_i(\neg for') = Vrai\}.
\end{aligned}$$

On a: $vv_i(for') = Faux \Leftrightarrow vv_{\neg neg(i)}(for') = Faux$, en effet $for'(x_1, ..., x_n)$ est de la forme:

$$\vee_i(\exists y_1...\exists y_p\,(x_1 = t_{i1}(y_1, ..., y_p)\wedge...\wedge x_n = t_{in}(y_1, ..., y_p)\wedge for_i(y_1, ..., y_p))).$$

$\exists(\alpha_1, ..., \alpha_n) \in UNI(L)^n$ tel que $vv_i(for'(\alpha_1, ..., \alpha_n)) = Faux \Leftrightarrow$

$\exists(\alpha_1, ..., \alpha_n) \in UNI(L)^n$ tel que $\forall i \in \{1, ..., m\}$, $\forall(u_1, ..., u_p) \in UNI(L)^p$,

$vv_i(\alpha_1 = t_{i1}(u_1, ..., u_p)) = Faux \vee ... \vee vv_i(\alpha_n = t_{in}(u_1, ..., u_p)) = Faux$

$$\vee vv_i(for_i(u_1, ..., u_p)) = \text{Faux}.$$

On a:

$$vv_i(\alpha_j = t_{ij}(u_1, ..., u_p)) = \text{Faux} \Leftrightarrow$$
$$vv_{\neg neg(i)}(\alpha_j = t_{ij}(u_1, ..., u_p)) = \text{Faux}$$

puisque la valeur de vérité de cette formule ne dépend pas de l'interprétation choisie.

De plus, $vv_i(for) = \text{Faux} \Leftrightarrow vv_{\neg neg(i)}(for) = \text{Faux}$ car:

$for = ato_1 \wedge ato_2 \wedge ... \wedge ato_n$ (par induction sur for).

Donc $\neg neg(T_{Comp(\neg, Pr)}(i)) = \neg neg(T_{Comp(\neg, Pr)}(\neg neg(i)))$. ♦

Lemme 5.5.4 *Soit $i \in IHT(L)$ et Pr un programme défini bivalué. On a alors:*

a) $pos(T_{Comp(\neg, Pr)}(pos(i))) = pos(B_{Pr}(pos^{-1}(pos(i))))$
b) $\neg neg(T_{Comp(\neg, Pr)}(\neg neg(i))) = \neg neg(B_{Pr}(\neg neg^{-1}(\neg neg(i))))$

Démonstration $B_{Pr}(pos^{-1}(pos(i))) = B_{Pr'}(pos^{-1}(pos(i))) =$
$pos^{-1}(\{ato/$ il existe une instance close de Pr', $ato \leftarrow for'$ telle que:
$vv_{pos^{-1}pos(i)}(for') = \text{Vrai}\})$.

$vv_{pos^{-1}pos(i)}(for') = \text{Vrai} \Leftrightarrow vv_{pos(i)}(for') = \text{Vrai}$ car $for' =$

$$\vee_i(\exists y_1 ... \exists y_p\, (x_1 = t_{i1}(y_1, ..., y_p) \wedge ... \wedge x_n = t_{in}(y_1, ..., y_p) \wedge for_i(y_1, ..., y_p)))$$

et $for_i = ato_1 \wedge ato_2 \wedge ... \wedge ato_n$.

Donc $pos(B_{Pr}(pos^{-1}(pos(i)))) = \{ato/$ il existe une instance close de Pr', $ato \leftarrow for'$ telle que $vv_{pos(i)}(for') = \text{Vrai}\}$
$= pos(T_{Comp(\neg, Pr)}(pos(i)))$.

$\neg neg(B_{Pr}(\neg neg^{-1}(\neg neg(i)))) = \neg neg(B_{Pr'}(\neg neg^{-1}(\neg neg(i))))$
$= \{\neg ato/\ ato \notin pos(B_{Pr'}(\neg neg^{-1}(\neg neg(i))))\}$
$= \{\neg ato/$ pour toute instance close $ato \leftarrow for'$ de Pr',

$vv_{\neg neg^{-1}(\neg neg(i))}(for') \neq$ Vrai (= Faux puisque $\neg neg^{-1}(\neg neg(i))$ est une interpretation bivaluée)}

$\cup$ {$\neg$ato/ ato ne soit pas en tête d'une règle de Pr'}.

$\neg neg(T_{Comp(\neg, Pr)}(\neg neg(i))) = \{\neg ato \in T_{Comp(\neg, Pr)}(\neg neg(i))\}$
= {$\neg$ato/ il existe une instance close $\neg ato \leftarrow \neg for$ de Comp($\neg$, Pr) telle que $vv_{\neg neg(i)}(\neg for)$ = Vrai}

$\cup$ {$\neg$ato/ ato ne soit pas en tête d'une règle de Pr}.

Or, pour toute instance close ato $\leftarrow$ for' de Pr', on a l'équivalence:

$vv_{\neg neg^{-1}(\neg neg(i))}(for') = Faux \Leftrightarrow vv_{\neg neg(i)}(for') = Faux.$

On a de plus l'équivalence suivante:

Pour toute instance close ato $\leftarrow$ for' de Pr', $vv_{\neg neg(i)}(for')$ = Faux $\Leftrightarrow$
Il existe une instance close ato $\leftarrow$ for' de Pr', $vv_{\neg neg(i)}(for')$ = Faux.

Par conséquent, $\neg neg(B_{Pr}(\neg neg^{-1}(\neg neg(i)))) =$
$\neg neg(T_{Comp(\neg, Pr)}(\neg neg(i)))$. ♦

Lemme 5.5.5 *Soient $i \in IHT(L)$ et Pr un programme défini bivalué. On a alors:*

a) $pos(T_{Comp(\neg, Pr)}\uparrow\alpha) = pos(B_{Pr}\uparrow\alpha)$.
b) $\neg neg(T_{Comp(\neg, Pr)}\uparrow\alpha) = \neg neg(B_{Pr}\downarrow\alpha)$.

Démonstration Par induction transfinie.

a) - $pos(T_{Comp(\neg, Pr)}\uparrow 0) = pos(\emptyset) = pos(B_{Pr}\uparrow 0)$.

- Si α est un ordinal successeur, alors:

$pos(T_{Comp(\neg, Pr)}(T_{Comp(\neg, Pr)}\uparrow\alpha\text{-}1))$
$= pos(T_{Comp(\neg, Pr)}(pos(T_{Comp(\neg, Pr)}\uparrow\alpha\text{-}1)))$ (Lemme 5.5.3)
$= pos(T_{Comp(\neg, Pr)}(pos(B_{Pr}\uparrow\alpha\text{-}1)))$ (induction)
$= pos(B_{Pr}(pos^{-1}(pos(B_{Pr}\uparrow\alpha\text{-}1))))$ (Lemme 5.5.4)
$= pos(B_{Pr}\uparrow\alpha)$, puisque $B_{Pr}\uparrow\alpha\text{-}1$ est une interprétation bivaluée.

- Si α est un ordinal limite, alors $\alpha = \sup\{\beta, \beta<\alpha\}$ et:

$pos(T_{Comp(\neg, Pr)}\uparrow\alpha)$
$= pos(\sup\{T_{Comp(\neg, Pr)}\uparrow\beta, \beta<\alpha\})$
$= \sup\{pos(T_{Comp(\neg, Pr)}\uparrow\beta), \beta<\alpha\}$
$= \sup\{pos(B_{Pr}\uparrow\beta), \beta<\alpha\}$ (par induction)
$= pos(B_{Pr}\uparrow\alpha)$.

b) - $\neg neg(T_{Comp(\neg, Pr)}\uparrow 0) = \neg neg(\varnothing) = \neg neg(B_{Pr}\downarrow 0) .= \varnothing$.
En effet, $B_{Pr}\downarrow 0$ = max IHB(L) = l'ensemble de tous les atomes positifs clos.

- Si α est un ordinal successeur, alors:

$\neg neg(T_{Comp(\neg, Pr)}(T_{Comp(\neg, Pr)}\uparrow\alpha\text{-}1))$
$= \neg neg(T_{Comp(\neg, Pr)}(\neg neg(T_{Comp(\neg, Pr)}\uparrow\alpha\text{-}1)))$ (lemme 5.5.3)
$= \neg neg(T_{Comp(\neg, Pr)}(\neg neg(B_{Pr}\downarrow\alpha\text{-}1)))$ (par induction)
$= \neg neg(B_{Pr}(\neg neg^{-1}(\neg neg(B_{Pr}\downarrow\alpha\text{-}1)))$ (lemme 5.5.4)
$= \neg neg(B_{Pr}\downarrow\alpha)$.

- Si α est un ordinal limite, alors:

$\neg neg(\sup\{T_{Comp(\neg, Pr)}\uparrow\beta, \beta<\alpha\})$
$= \sup\{\neg neg(T_{Comp(\neg, Pr)}\uparrow\beta), \beta<\alpha\}$
$= \sup\{\neg neg(B_{Pr}\downarrow\beta), \beta<\alpha\}$ (par induction)
$= \neg neg(\inf\{B_{Pr}\downarrow\beta, \beta<\alpha\}$
$= \neg neg(B_{Pr}\downarrow\alpha)$,

où sup est la borne sup dans IHT(L) muni de l'ordre $i \le j \Leftrightarrow i \subseteq j$ et inf est la borne inf dans IHB(L) muni de l'ordre $i \le j \Leftrightarrow pos(i) \subseteq pos(j)$. ♦

On a alors le résultat en écrivant que:
$T_{Comp(\neg, Pr)}\uparrow\alpha = pos(T_{Comp(\neg, Pr)}\uparrow\alpha) \cup \neg neg(T_{Comp(\neg, Pr)}\uparrow\alpha)$.
♦

Dans le paragraphe suivant, nous donnons une sémantique opérationnelle pour les programmes trivalués, ou encore un moyen siple d'obtenir le plus petit modèle trivalué d'un programme trivalué à partir de ceux déjà utilisés.

5.6. INTERPRETEURS

Le sens d'un programme est donné par son plus petit modèle modèle. On cherche maintenant des algorithmes qui calculent ce plus petit modèle ou donnent des informations sur lui.
On définit ainsi trois sortes d'interpréteurs:

- **Les interpréteurs réponse fermés**:
qui répondent si un atome clos (resp. littéral) appartient ou non à ppmb(Pr) (resp. ppmt(Pr)).

- **Les interpréteurs réponse ouverts**:
qui donnent, pour un atome donné (resp. littéral) avec variables, $ato(x_1, x_2, ..., x_n)$, (resp. $lit(x_1, x_2, ..., x_n)$), une suite de substitutions $\theta_1, \theta_2, ...$, telles que toutes les instances closes de $\theta_i(ato(x_1, x_2, ..., x_n))$ (resp. $\theta_i(lit(x_1, x_2, ..., x_n))$) soient exactement les instances closes de $ato(x_1, x_2, ..., x_n)$ (resp. $lit(x_1, x_2, ..., x_n)$) qui sont dans ppmb(Pr) (resp. ppmt(Pr)).

- **Les interpréteurs saturateurs**:
qui donnent une représentation finie de ppmb(Pr) (resp. ppmt(Pr)).

5.6.1 Il est facile de transformer un interpréteur trivalué pour un programme bivalué en un interpréteur bivalué pour ce programme:
puisque $ppmb(Pr) = pos^{-1}(ppmt(Pr))$. (proposition 5.5.2 (b))

5.6.2 Il est intéressant de voir comment obtenir un interpréteur trivalué à partir d'un bivalué. On commence par transformer un programme trivalué en un bivalué avec les transformations suivantes:
Soit Pr un programme trivalué sur un langage logique du premier ordre L.

$\sigma(L)$ est le langage logique du premier ordre obtenu à partir de L en lui ajoutant les symboles de prédicats, non-p, pour chaque symbole de prédicat p de L.

Soit $\sigma(Pr)$ le programme bivalué sur $\sigma(L)$ obtenu à partir de Pr par les transformations suivantes:

1) On supprime $\Leftrightarrow$ et $\Rightarrow$ en utilisant:
$A \Rightarrow B \equiv_T (\neg A \vee B)$
$A \Leftrightarrow B \equiv_T (A \wedge B) \vee (\neg A \wedge \neg B)$.

2) On ramène le connecteur $\neg$ juste devant les atomes en utilisant:
$\neg(A \vee B) \equiv_T (\neg A \wedge \neg B)$.
$\neg(A \wedge B) \equiv_T (\neg A \vee \neg B)$.

$\neg(\forall x\ A(x)) \equiv_T \exists x\ (\neg A(x))$.
$\neg(\exists x\ A(x)) \equiv_T \forall x\ (\neg A(x))$.
$\neg\neg A \equiv_T A$.

3) On remplace tous les $\neg$ato par non-ato.

Définition 5.6.2.1 Si $i \in$ IHT(L), alors $\sigma(i)$ = pos(i) $\cup$ {non-p($t_1, t_2, ..., t_n$)/ $\neg$p($t_1, t_2, ..., t_n$) $\in$ i}. L'interprétation $\sigma(i)$ appartient à IHT(σ(L)).
σ est une bijection entre IHT(L) et pos(IHB(σ(L)).

On obtient un interpréteur trivalué à partir d'un bivalué par la proposition suivante:

Proposition 5.6.2.1 *Si Pr est un programme trivalué consistant,*
σ(ppmt(Pr)) = pos(ppmb(σ(Pr)));
donc ppmt(Pr) = σ^{-1}(pos(ppmb(σ(Pr)))).

Démonstration On montre par induction transfinie que:
$\sigma(T_{Pr}\uparrow\alpha) = \text{pos}(B_{\sigma(Pr)}\uparrow\alpha)$ pour tout ordinal α.

Comme ppmt($\forall$Pr) = $T_{Pr}\uparrow\alpha$ et ppmb(σ(Pr)) = $B_{\sigma(Pr)}\uparrow\beta$, on a le résultat en prenant le plus grand des deux ordinaux.

On montre que, $\forall i \in$ IHT(L), $\sigma(T_{Pr}(i)) = (\text{pos} \circ B_{\sigma(Pr)} \circ \text{pos}^{-1})(\sigma(i))$ (*).

On utilise le fait que $B_{\sigma(Pr)} = \text{pos}^{-1} \circ T_{\sigma(Pr)} \circ \text{pos}$ puisque σ(Pr) est un programme bivalué, d'après la proposition 5.5.2 (a).

On doit donc montrer que: $\sigma(T_{Pr}(i)) = T_{\sigma(Pr)}(\sigma(i))$.

$\sigma(T_{Pr}(i)) = \text{pos}(T_{Pr}(i)) \cup$ {non-p($t_1, t_2, ..., t_n$) tel que $\neg$p($t_1, t_2, ..., t_n$) $\in T_{Pr}(i)$}.

$T_{\sigma(Pr)}(\sigma(i))$ = {ato $\in \sigma$(L) / il existe une instance close ato $\leftarrow$ for de σ(Pr) telle que $vv_{\sigma(i)}$(for) = Vrai}.

Si lit $\in \sigma(T_{Pr}(i))$, soit lit = ato avec ato $\in T_{Pr}(i)$ ou lit = non-ato avec $\neg$ato $\in T_{Pr}(i)$ et on a le résultat car $vv_{\sigma(i)}(\sigma(\text{for}))$ = Vrai $\Leftrightarrow$ vv_i(for) = Vrai.

On a alors $\sigma(T_{Pr}\uparrow\alpha) = pos(B_{\sigma(Pr)}\uparrow\alpha)$ pour tout ordinal α, par induction transfinie:

- $\alpha = 0$, $T_{Pr}\uparrow 0 = \varnothing = pos(\varnothing)$.

- Si α est un ordinal successeur, alors:

$$\begin{aligned}
&\sigma(T_{Pr}(T_{Pr}\uparrow\alpha -1)) \\
&\quad = (pos \circ B_{\sigma(Pr)} \circ pos^{-1})(\sigma(T_{Pr}\uparrow\alpha -1)) \text{ (selon (*))} \\
&\quad = (pos \circ B_{\sigma(Pr)} \circ pos^{-1})(pos(B_{\sigma(Pr)}\uparrow\alpha -1)) \text{ par induction,} \\
&\quad = pos(B_{\sigma(Pr)}\uparrow\alpha).
\end{aligned}$$

- Si α est un ordinal limite, alors:

$$\begin{aligned}
&\sigma(T_{Pr}\uparrow\alpha) \\
&\quad = \sigma(sup_{IHT(L)}\{T_{Pr}\uparrow\beta, \beta < \alpha\}) \\
&\quad = \sigma(sup_{IHT(L)}\{pos(B_{\sigma(Pr)}\uparrow\beta), \beta < \alpha\}) \text{ par induction;}
\end{aligned}$$

on a $sup_{IHT(L)}\{pos(i), i \in I \subseteq IHB(L)\} = pos(sup_{IHB(L)}\{i, i \in I \subseteq IHB(L)\})$; donc:

$$\sigma(T_{Pr}\uparrow\alpha) = pos(B_{\sigma(Pr)}\uparrow\alpha).$$

Donc:

$$\begin{aligned}
&\sigma(ppmt(Pr)) = pos(ppmb(\sigma(Pr))) \text{ et} \\
&\qquad ppmt(Pr) = \sigma^{-1}(pos(ppmb(\sigma(Pr)))). \ \blacklozenge
\end{aligned}$$

Exemple 5.6.2.1 Soit Pr le programme (A, B, C, D, E sont des atomes clos):

$$\begin{aligned}
&A, \neg B \leftarrow \\
&C \leftarrow A \\
&\neg D \leftarrow \neg B, C \\
&E \leftarrow C, \neg D
\end{aligned}$$

$\sigma(Pr)$ est le programme:

$$\begin{aligned}
&A, \text{non-}B \leftarrow \\
&C \leftarrow A \\
&\text{non-}D \leftarrow \text{non-}B, C \\
&E \leftarrow C, \text{non-}D
\end{aligned}$$

$ppmb(\sigma(Pr)) = \{A, non\text{-}B, C, non\text{-}D, E, \neg non\text{-}A, \neg non\text{-}C, \neg non\text{-}E, \neg B, \neg D\}$

$pos(ppmb(\sigma(Pr))) = \{A, non\text{-}B, C, non\text{-}D, E\}$.

$\sigma^{-1}(pos(ppmb(\sigma(Pr)))) = \{A, \neg B, C, \neg D, E\} = ppmt(Pr)$.

CHAPITRE 6: MODELES OPTIMAUX

6.1 INTRODUCTION

L'ensemble des interprétations de Herbrand trivaluées est en bijection avec l'ensemble des fonctions partielles de her(L) dans {V, F}, par l'application ϕ qui à une interprétation i associe la fonction vv_i, non définie sur les atomes où elle prend la valeur Indéterminé. La théorie de Manna et Shamir [28] concernant les fonctionnelles monotones sur un ensemble de fonctions partielles peut donc être adaptée à notre opérateur conséquence T_{Pr}. Cependant plutôt que de choisir le point fixe optimal de T_{Pr}, comme autre modèle désigné d'un programme trivalué que le plus petit point fixe de T_{Pr}, on préfèrera la notion de modèle optimal de Pr.

En effet, les modèles d'un programme trivalué ne sont plus les points fixes de T_{Pr} mais ses post-points fixes; on va donc substituer la notion de modèle optimal à celle de point fixe optimal. L'ensemble des interprétations de Herbrand trivaluées n'a plus une stucture de treillis complet puisqu'une union de deux interprétations de Herbrand trivaluées n'est pas nécessairement une interprétation de Herbrand trivaluée. Il possède seulement une structure de semi-treillis complet: l'intersection de deux interprétations de Herbrand en est une ainsi qu'une union consistante d'interprétations de Herbrand. Il n'y a donc plus de plus grand modèle mais une série de modèles maximaux dont l'intersection, qui est encore un modèle du programme coïncide avec le plus grand des modèles consistants avec tous les modèles.

Le modèle optimal de Pr ne coïncide pas forcément avec son plus petit modèle trivalué et contient donc plus d'informations. Il va permettre de pallier un peu la faiblesse de la logique trivaluée. En effet, il fournit une autre sémantique dénotationnelle que le plus petit point fixe de l'opérateur conséquence pour un programme trivalué. Le plus petit point fixe de l'opérateur T_{Pr}, coïncide avec l'ensemble de tous les littéraux clos, conséquences logiques du programme. Or il semble que la logique trivaluée calcule trop peu de conséquences logiques:

Si Pr est le programme suivant:

$$q(x) \leftarrow p(x)$$
$$\neg q(a)$$

le littéral $\neg p(a)$ n'est pas conséquence logique de Pr, il n'est donc pas dans le modèle désigné de Pr (plus petit modèle ppmt(Pr)). Cela provient essentiellement du fait que la formule $q(x) \leftarrow p(x)$ n'est pas logiquement équivalente à la formule $\neg p(x) \leftarrow \neg q(x)$ en logique trivaluée. En effet, l'ensemble $PF(T_{Pr})$ se réduit au singleton $\{\neg q(a)\}$, alors que l'ensemble MOD(Pr) des modèles de Pr comprend les ensembles $\{\neg q(a)\}$, et $\{\neg p(a), \neg q(a)\}$. Le modèle optimal de Pr sera $\{\neg p(a), \neg q(a)\}$ et comprendra donc le littéral $\neg p(a)$.

La théorie du modèle optimal est plus simple que celle du point fixe optimal puisque une intersection de modèles du programme est encore un modèle alors qu'une intersection de points fixes de l'opérateur conséquence n'est plus nécessairement un point fixe de T_{Pr}.

Le plan du chapitre est le suivant:

Dans une première partie, on rappelle les différentes notions définies dans Manna et Shamir [28]; dans une deuxième partie, on applique ces différentes notions à l'opérateur de Lassez et Maher [21]; enfin dans une troisième partie on introduit notre terminologie de modèles consistants, maximaux, optimaux et on démontre des propriétés structurelles de l'ensemble des post-points fixes de T_{Pr}, c'est-à-dire des modèles de Pr.

On démontre l'existence d'un unique modèle optimal et on étudie certaines de ses propriétés. Le modèle optimal coïncide avec le plus grand modèle consistant et l'intersection de tous les modèles maximaux.

Dans une quatrième partie, on compare toutes les notions étudiées sur des exemples de programmes trivalués. On étudie les correspondances entre le plus petit point fixe de T_{Pr}, son plus petit post-point fixe, son point fixe optimal et son post-point fixe optimal.

On verra sur des exemples, que contrairement à l'opérateur τ introduit dans [21], le plus petit point fixe de T_{Pr} et son point fixe optimal ne coïncident pas.

De même, les notions de plus petit modèle de Pr (en fait égal au plus petit point fixe de T_{Pr}) et de modèle optimal ne coïncident pas pour l'opérateur T_{Pr} des programmes trivalués.

Ces notions théoriquement bien définies présentent donc un intérêt intrinsèque, et peuvent fournir plus d'informations que celle de plus petit point fixe. Le problème est plutôt d'ordre pratique et réside dans leur calculabilité.

6.2. RAPPEL DES NOTIONS DEFINIES DANS [28]

6.2.1 Notations.

On désigne par $D^+ = D \cup \{\omega\}$, l'ensemble D auquel on a adjoint un élément ω.
FP(A,D) désigne l'ensemble des fonctions partielles de A dans D, c'est-à-dire qui prennent la valeur ω aux points où elles ne sont pas définies:

C'est l'ensemble des fonctions de A dans D^+.

On munit D^+ de l'ordre partiel suivant:

$$x \leq y \Leftrightarrow x = y \text{ ou } x = \omega.$$

L'ordre partiel $\leq$ sur FP(A,D) associé à $\leq$ (l'ordre produit associé) est le suivant:

$$f \leq g \Leftrightarrow \forall x \in A, f(x) \leq g(x).$$

Définitions 6.2.1 Deux éléments f et g de FP(A,D) sont consistants si:

$$\forall x \in A, f(x) \neq \omega \text{ et } g(x) \neq \omega \Rightarrow f(x) = g(x).$$

Un sous ensemble de FP(A,D) est consistant si ses éléments sont consistants deux à deux.

Remarque: En particulier un sous ensemble totalement ordonné de FP(A,D) est consistant.

Lemme 6.2.1 *a) FP(A,D) admet un plus petit élément $\perp$, la fonction valant partout ω.*

b) Tout sous ensemble non vide de FP(A,D) admet une borne sup si et ssi il est consistant.

Donc tout sous ensemble non vide totalement ordonné de FP(A,D) admet une borne sup.

c) Tout sous ensemble non vide de FP(A,D) admet une borne inf.

Démonstration $\forall x \in A, \omega \leq x$.

Soit S un sous ensemble non vide et consistant de FP(A,D).

On définit la borne sup α de S de la façon suivante:
Si $\forall s \in S$, $s(x) = \omega$, $\alpha(x) = \omega$; si $\exists s \in S$, $s(x) \neq \omega, \forall s' \in S$, $s'(x) \neq \omega$, $s'(x) = s(x)$ car S est consistant et $\alpha(x) = s(x)$.
On voit immédiatement que α est un majorant: $\forall s \in S$, $s(x) \leq \alpha(x)$, et c'est le plus petit.

Tout sous ensemble totalement ordonné est consistant.

Soit S un sous ensemble non vide de FP(A,D);

On définit la borne inf β de S de la façon suivante:
Soit $x \in A$, si $\exists(s,s') \in S^2$, $s(x) \neq s'(x)$, $\beta(x) = \omega$; si $\forall(s,s') \in S^2$, $s(x) = s'(x)$, $\beta(x) = s(x)$. ♦

Nous utiliserons le théorème du point fixe suivant:

Théorème 6.2.1 *Soit $(S, \leq)$ un ensemble partiellement ordonné ayant un plus petit élément $\perp$, tel que tout sous ensemble ordonné non vide de S admette une borne sup, τ une fonctionnelle monotone de S dans S, alors $\exists\alpha$ ordinal tel que τ admette un plus petit point fixe, $\tau\uparrow\alpha$.*

Définition 6.2.2 Si τ est une fonctionnelle de FP(A,D) dans lui-même, on définit l'ensemble de ses points fixes, de ses points fixes consistants, de ses points fixes maximaux, de ses pré-points fixes et de ses post-points fixes de la manière suivante:

L'ensemble des points fixes de τ est:
$$PF(\tau) = \{f \in FP(A,D), f = \tau(f)\}.$$

L'ensemble des points fixes de τ consistants avec tous les autres points fixes de τ est:
$$PFC(\tau) = \{f \in PF(\tau), f \text{ est consistant avec tout point fixe de } \tau\}.$$

L'ensemble des points fixes maximaux de τ est:
$$MAX(\tau) = \{f \in PF(\tau), \forall g \in PF(\tau), f \leq g \Rightarrow f = g\}.$$

L'ensemble des pré-points fixes de τ est:
$$PRE(\tau) = \{f \in FP(A,D), f \leq \tau(f)\}.$$

L'ensemble des post-points fixes de τ est:
$$POST(\tau) = \{f \in FP(A,D), \tau(f) \leq f\}.$$

Dans toute la suite τ désigne une fonctionnelle monotone de FP(A,D) dans lui-même.

Lemme 6.2.2 *a) τ admet un plus petit élément noté pppf(τ) dans FP(A,D).*

b) Si $f \in PRE(\tau)$, l'ensemble $\{f' \in PF(\tau), f \leq f'\}$ admet un plus petit élément.

c) Si $f \in POST(\tau)$, l'ensemble $\{f' \in PF(\tau), f' \leq f\}$ admet un plus grand élément.

Démonstration

a) FP(A,D) vérifie les conditions du théorème du point fixe (Lemme 2.1).

b) Soit $S = \{f' \in FP(A,D), f \leq f'\}$;

S est stable par τ: $f \leq \tau(f) \leq \tau(f')$, par monotonie de τ et parce que $f \in PRE(\tau)$; S admet un plus petit élément f; S vérifie donc les conditions du théorème du point fixe; $\exists$ un plus petit point fixe de τ dans S;

c) On choisit $S' = \{f' \in FP(A,D), f' \leq f\}$; on le munit de l'ordre $\leq_1$:
$g \leq_1 h \Leftrightarrow h \leq g$ et on réapplique le raisonnement précédent. ♦

Théorème 6.2.2 *PFC(τ) admet un plus grand élément appelé le point fixe optimal de τ et noté opt(τ).*

Démonstration $PFC(\tau) \neq \emptyset$, $pppf(\tau) \in PFC(\tau)$ puisque $\forall f \in PF(\tau)$, $pppf(\tau) \leq f$.

Donc il admet une borne sup puisque c'est un ensemble consistant.
Soit α cette borne sup; on va montrer que $\alpha \in PFC(\tau)$.

α vérifie: $\forall f \in PFC(\tau), f \leq \alpha$ et $f = \tau(f) \leq \tau(\alpha)$, donc $\alpha \leq \tau(\alpha)$, par définition de la borne sup. Donc $\alpha \in PRE(\tau)$ et $\{f' \in PF(\tau), \alpha \leq f'\}$ admet un plus petit élément f_2.

On montre que $f_2 \in PFC(\tau)$, ce qui entraînera $f_2 \leq \alpha \leq f_2$ donc $\alpha = f_2$, et $\alpha \in PFC(\tau)$.

Soit $g \in PF(\tau)$, on montre que f_2 et g sont consistants en montrant que $\exists f_3 \in PF(\tau)$, que $f_2 \leq f_3$ et $g \leq f_3$.

$S = PFC(\tau) \cup \{g\}$ est consistant, $\neq \varnothing$, donc il admet une borne sup ϕ dans FP(A,D). ϕ vérifie: $\forall f \in PFC(\tau), f \leq \phi, f = \tau(f) \leq \tau(\phi)$, et $g = \tau(g) \leq \tau(\phi)$, donc $\phi \leq \tau(\phi)$, par définition de la borne sup. Donc $\phi \in PRE(\tau)$ et $\{f' \in PF(\tau), \phi \leq f'\}$ admet un plus petit élément f_3. On a $\sup(PFC(\tau)) \leq f_3$, $g \leq \phi \leq f_3$. Comme f_2 est le plus petit élément de $PF(\tau)$ tel que $\alpha = \sup(PFC(\tau)) \leq f_2$, on a $f_2 \leq f_3$ et par conséquent, $f_2 \in PFC(\tau)$. ♦

On considère maintenant les points fixes maximaux et le lien entre les deux notions.

Lemme 6.2.3 *$\forall f \in PRE(\tau), \exists g \in MAX(\tau), f \leq g$.*

Démonstration $S_f = \{f' \in PF(\tau), f \leq f'\}$, comme $f \in PRE(\tau)$, S_f admet un plus petit élément et est donc non vide.

Tout sous-ensemble S de S_f totalement ordonné admet un majorant dans S_f:

Soit $S \subseteq S_f$ totalement ordonné, S est consistant, donc admet une borne sup α dans FP(A,D); $\alpha \in PRE(\tau)$, car $\forall f' \in S$, $f' = \tau(f') \leq \tau(\alpha)$, donc $\alpha \leq \tau(\alpha)$, par définition de la borne sup.

Donc $\{f' \in PF(\tau), \alpha \leq f'\}$ admet un plus petit élément f_1. $\forall f' \in S$, $f \leq f' \leq \alpha \leq f_1$. Comme $f_1 \in PF(\tau)$, $f_1 \in S_f$ et est un majorant de S dans S_f.

Donc S_f vérifie les conditions du lemme de Zorn et admet un élément maximal qui est alors un point fixe maximal de τ, supérieur ou égal à f. ♦

Corollaire 6.2.1 *Pour toute fonctionnelle monotone τ de FP(A,D) dans lui-même, $MAX(\tau) \neq \varnothing$.*

Démonstration $PRE(\tau) \neq \varnothing$ puisque $pppf(\tau) \in PRE(\tau)$. ♦

Lemme 6.2.4 *Si $g \in PRE(\tau)$ et $f \in MAX(\tau)$, soit $g \leq f$, soit f et g ne sont pas consistants.*

Démonstration Supposons que non($g \leq f$) et que f et g soient consistants. Alors $\sup(f,g) = \alpha$ existe dans FP(A,D) et est distinct de f puisque $g \leq \sup(f,g)$ et non($g \leq f$); $\alpha \in PRE(\tau)$ car $\tau(f) = f \leq \tau(\alpha)$ et $g \leq \tau(g) \leq \tau(\alpha)$, donc $\alpha \leq \tau(\alpha)$; par conséquent, l'ensemble $\{f' \in PF(\tau), \alpha \leq f'\}$ admet un plus petit élément f_1. Comme $f \leq \sup(f,g) \leq f_1$, f étant distinct de α, est distinct de f_1, et cela contredit le fait que f est un point fixe maximal. ♦

Corollaire 6.2.2 *Deux points fixes maximaux distincts ne sont pas consistants.*

Définition 6.2.3 Comme $MAX(\tau) \neq \varnothing$, il admet une borne inf notée $lmax(\tau)$.

Théorème 6.2.2 *a) $lmax(\tau) \in POST(\tau)$ et $\{f \in PF(\tau), f \leq lmax(\tau)\}$ admet un plus grand élément.*

b) Ce plus grand élément coïncide avec $max(PFC(\tau)) = opt(\tau)$.

Démonstration a) $lmax(\tau) = \inf(MAX(\tau))$; donc $\forall g \in MAX(\tau)$, $lmax(\tau) \leq g$, $\tau(lmax(\tau)) \leq \tau(g) = g$, donc $\tau(lmax(\tau)) \leq lmax(\tau)$ par définition de la borne inf.

Par conséquent, $\{f \in PF(\tau), f \leq lmax(\tau)\}$ admet un plus grand élément f_2.

b) $opt(\tau)$ est consistant avec tout point fixe de τ donc $\forall g \in MAX(\tau)$, $opt(\tau) \leq g$, d'après le corollaire 6.2.2. Donc $opt(\tau) \leq lmax(\tau)$ par définition de la borne inf.

Donc $opt(\tau) \leq f_2$.

Montrons que $f_2 \leq opt(\tau)$ en montrant que f_2 est un point fixe consistant de τ.

Soit f un point fixe de τ. D'après le lemme 6.2.3, $\exists g \in MAX(\tau), f \leq g$.
Or f_2 est le plus grand élément de l'ensemble $\{f \in PF(\tau), f \leq lmax(\tau)\}$.
Donc $f_2 \leq lmax(\tau) \leq g$;

$f \leq g$ et $f_2 \leq g \Rightarrow$ f et f_2 sont consistants. ♦

Remarques: - On a donc deux façons d'obtenir $opt(\tau)$, l'une en "descendant" des points fixes maximaux, l'autre en "montant" des points fixes consistants.

- Si pppf(τ) est une fonction totale, i.e. $\neq \omega$ partout, alors pppf(τ) = opt(τ), puisque pppf(τ) $\leq$ opt(τ).

- Si opt(τ) est maximal, comme $\forall g \in$ MAX(τ), opt(τ) $\leq$ lmax(τ) $\leq$ g, on a opt(τ) = g et il n'y a qu'un seul point fixe maximal, opt(τ).

- Comme pppf(τ) $\leq$ opt(τ), le point fixe optimal peut être plus intéressant au sens où il contient plus d'information, étant plus défini que le plus petit point fixe de τ.
Cependant, le point fixe optimal et le plus petit point fixe peuvent coïncider comme dans l'exemple de Lassez et Maher [21] que nous étudierons dans le paragraphe suivant.

Nous prendrons D = {V,F}, ω = I et A = her(L), c'est-à-dire, nous prendrons pour FP(A,D) l'ensemble des interprétations de Herbrand trivaluées qui est en bijection avec l'ensemble des fonctions de her(L) dans {V,F,I}, i.e. l'ensemble des fonctions partielles de her(L) dans {V,F} (prenant la valeur I, là où elles ne sont pas définies).
L'ordre sur IHT(L) est l'ordre associé à I $\leq$ V, I $\leq$ F. C'est l'ordre qui rend la bijection ϕ de IHT(L) vers l'ensemble des fonctions de her(L) dans {V,F,I}, qui à une interprétation i associe vv_i, monotone.

Si nous reprenons l'exemple de programme présenté au début du chapitre:

$$q(x) \leftarrow p(x)$$
$$\neg q(a)$$

le plus petit point fixe et le point fixe optimal de T_{Pr} coïncident; ce dernier ne contient toujours pas $\neg p(a)$, qui appartiendra par contre au modèle optimal de Pr.

6.3 UN CAS OU pppf(τ) ET opt(τ) COINCIDENT: τ EST L'OPERATEUR DE LASSEZ ET MAHER [21]

Les programmes considérés sont des programmes définis sans négation: c'est-à-dire des ensembles de clauses $A \leftarrow B_1, ..., B_n$

On choisit comme ensemble FP(A,D) l'ensemble des fonctions de l'ensemble des atomes de Herbrand vers {V,F,I}.

Le connecteur $\rightarrow$ a la table de vérité suivante:

$A \rightarrow B$ n'est pas Vrai si A vaut V et B vaut I ($A \rightarrow B$ vaut I), si A vaut V et B vaut F ($A \rightarrow B$ vaut F), si A vaut I et B vaut F ($A \rightarrow B$ vaut I).

(c'est le connecteur de Lukasiewicz).

L'opérateur τ est défini de la manière suivante:

Si il existe une instance close d'une règle de P ayant A en tête de règle, $A \leftarrow B_1, ..., B_n$, telle que $f(B_1 \wedge ... \wedge B_n) = V$, $\tau(f)(A) = V$.

Si pour toute instance close des règles de P ayant A en tête de règle, $A \leftarrow B_1, ..., B_n$, $f(B_1 \wedge ... \wedge B_n) = F$, $\tau(f)(A) = f(A)$.

Si pour toute instance close des règles de P ayant A en tête de règle, $A \leftarrow B_1, ..., B_n$, $f(B_1 \wedge ... \wedge B_n) \neq V$, et il existe une instance close d'une règle de P ayant A en tête de règle, $A \leftarrow B_1, ..., B_n$, telle que $f(B_1 \wedge ... \wedge B_n) = I$, $\tau(f)(A) = \inf(V, f(A))$.

Pour le cas où A ne figure pas en tête d'une règle de P, $\tau(f)(A) = f(A)$.

Dans l'article sont démontrées les propositions suivantes:

Proposition 6.3.1 *$\tau(f) = f \Leftrightarrow f$ est un modèle de P.*

Démonstration f est un modèle $\Rightarrow \tau(f) = f$.

1) Pour toute instance close des règles de P ayant A en tête de règle, $A \leftarrow B_1, ..., B_n$, $\exists i\ f(B_i) = F$; alors $f(A \leftarrow B_1, ..., B_n) = V$ et $\tau(f)(A) = f(A)$.

2) Il existe instance close des règles de P ayant A en tête de règle, $A \leftarrow B_1, ..., B_n$, $\forall i\ f(B_i) \neq F$;

a) $\exists$ instance close des règles de P ayant A en tête de règle, $A \leftarrow B_1, ..., B_n$, $\forall i\ f(B_i) = V$; Alors $f(A) = V$, donc $f(A) = \tau(f)(A)$;

b) $\exists$ instance close des règles de P ayant A en tête de règle, $A \leftarrow B_1, ..., B_n$, $\forall i\ f(B_i) \neq F$; $\forall$ instance close des règles de P ayant A en tête de règle, $A \leftarrow B_1, ..., B_n$, $\exists i\ f(B_i) \neq V$; alors $f(B_1 \wedge ... \wedge B_n) = I$ et $\tau(f)(A) = \inf(V, f(A))$.
Comme f est un modèle de P, $f(A) = V$ ou I et donc $f(A) = \tau(f)(A)$.

- Réciproquement $f = \tau(f) \Rightarrow f$ est un modèle de P.

Soit $A \leftarrow B_1, ..., B_n$, une instance d'une règle de P.

Si $\forall i$ $f(B_i) = V$, alors $f(A) = \tau(f)(A) = V$ et $f(A \leftarrow B_1, ..., B_n) = V$.

Si $\forall i$ $f(B_i) \neq F$ et $\exists i$ $f(B_i) \neq V$, alors $\exists i$ $f(B_i) = I$ et $\tau(f)(A) = \inf(f(A),V) = f(A)$ donc $f(A) = V$ ou I et $f(A \leftarrow B_1, ..., B_n) = V$.

Si $\exists i$ $f(B_i) = F$, alors $f(A \leftarrow B_1, ..., B_n) = V$. ♦

La fonctionnelle τ ainsi définie est monotone et non continue en général; on montre cependant la proposition suivante (toujours vrai lorsque τ est continue):

Proposition 6.3.2 *$pppf(\tau) = \tau\uparrow\omega = \bigcup_{n\in\mathcal{N}} \tau^n(\emptyset)$.*

On peut dans ce cas construire effectivement $pppf(\tau)$ et $opt(\tau)$.

Soit $\zeta(A) = V$ si A est V dans tout modèle de P.(1)
$\zeta(A) = F$ si A est F dans tout modèle de P.(2)
$\zeta(A) = I$ sinon (3)

Soit $\zeta'(A) = V$ si A est V dans un modèle de P et F dans aucun autre. (1)
$\zeta'(A) = F$ si A est F dans un modèle de P et V dans aucun autre. (2)
$\zeta'(A) = I$ sinon. (3)

En fait, les cas (2) de ζ et ζ' ne se produisent jamais:

En effet l'interprétation f constamment égale à V est un modèle de P, puisque $f = \tau(f)$.

Proposition 6.3.3 *$\zeta = pppf(\tau)$ et $\zeta' = opt(\tau)$.*

Démonstration On a $\zeta(A) = \inf\{f(A), f$ modèle de $P\}$; on a donc:
$\zeta(A) = (\inf\{f, f$ modèle de $P\})(A) = pppf(\tau)(A)$;

On montre que $\zeta'(A) = \max\{f(A), f \in PFC(\tau)\}$; on aura alors:
$\zeta(A) = (\max\{f, f \in PFC(\tau)\})(A) = opt(\tau)(A)$;

Pour cela, on montre que s'il existe un modèle M de P dans lequel A est V et A n'est F dans aucun autre, alors il existe un modèle consistant M' de P dans lequel A est V.

Soit M' défini de la façon suivante:

> M'(B) = V si B est V dans M et B n'est F dans aucun autre modèle.
> M'(B) = pppf(τ)(B) sinon.

M'(B) $\in$ {I,V} car pppf(τ)(B) $\in$ {I,V} puisque pppf(τ) $\leq$ f où f est l'interprétation constamment égale à V, qui est un modèle de P.

On montre que τ(M') = M'.

> τ(M')(B) = V si $\exists$ B $\leftarrow$ B_1, ..., B_n avec $\forall$i M'(B_i) = V. (1)
> τ(M')(B) = inf(M'(B),V) si le premier cas ne se produit pas et si $\exists$ B $\leftarrow$ B_1, ..., B_n avec $\forall$i M'(B_i) $\neq$ F. (2)
> τ(M')(B) = M'(B) si $\forall$ B $\leftarrow$ B_1, ..., B_n, $\exists$i M'(B_i) = F. (3)

1) $\forall$i M'(B_i) = V $\Rightarrow$ B_i est V dans M et F dans aucun autre modèle.
Dans ce cas B est V dans M car M(B $\leftarrow$ B_1, ..., B_n) = V.
Dans un autre modèle M" de P, on a M"(B_i) = V ou I; si M"($B_1 \wedge ... \wedge B_n$) = V, M"(B) = V; si M"($B_1 \wedge ... \wedge B_n$) = I, M"(B) = V ou I;
donc M'(B) = V = τ(M')(B).

2) τ(M')(B) = inf(M'(B),V) si le premier cas ne se produit pas et si il existe:
B $\leftarrow$ B_1, ..., B_n avec $\forall$i M'(B_i) $\neq$ F.
si M'(B) = V, τ(M')(B) = V = M'(B);
si M'(B) = I, τ(M')(B) = I = M'(B);
il n'y a pas d'autre cas à envisager car M'(B) $\neq$ F;

3) τ(M')(B) = M'(B) si $\forall$ B $\leftarrow$ B_1, ..., B_n, $\exists$i M'(B_i) = F. (3)

Donc M' est un modèle de P.

Montrons que M' est consistant.

Soit M" un modèle de P; si M"(B) $\neq$ I et M'(B) $\neq$ I, alors B est V dans M et F dans aucun autre modèle, donc M"(B) $\neq$ F, et M"(B) = M'(B).

Le cas 2) ne se produit jamais pour ζ';

Le cas 3):

Soit B est indéterminé dans tous les modèles: auquel cas il est indéterminé dans tous les modèles consistants;
Soit B est V dans un modèle, F dans un autre, auquel cas il est indéterminé dans tous les modèles consistants;

Donc $\zeta'(B) = opt(\tau)$. ♦

On démontre enfin:

Proposition 6.3.4 *$pppf(\tau) = opt(\tau)$.*

Démonstration Soit v l'interprétation constamment égale à V.

Soit f l'interprétation $pppf(\tau)$, f ne prend jamais la valeur F puisque v est un point fixe de τ et $f \leq v$.

Dans la définition de $\tau(f)(A)$,

$\tau(f)(A) = V$ si $\exists\ A \leftarrow A_1, ..., A_n$ avec $\forall i\ f(A_i) = V$. (1)
$\tau(f)(A) = inf(f(A),V)$ si le premier cas ne se produit pas
et si $\exists\ A \leftarrow A_1, ..., A_n$ telle que: $\forall i\ f(A_i) \neq F$. (2)
$\tau(f)(A) = f(A)$ si $\forall\ A \leftarrow A_1, ..., A_n, \exists i\ f(A_i) = F$. (3)

le troisième cas ne se produit donc jamais.

Soit f' tel que: $f'(A) = V$ si $f(A) = V$, $f'(A) = F$ si $f(A) = I$.

Si le cas 1) de la définition de $\tau(f)(A)$ se produit pour f, il se produit pour f', et $\tau(f')(A) = V = \tau(f)(A) = f(A) = f'(A)$.

Si le cas 2) se produit pour f, $\forall\ A \leftarrow A_1, ..., A_n, \exists i\ f(A_i) \neq V$, dans ce cas $\forall\ A \leftarrow A_1, ..., A_n, \exists i\ f'(A_i) = F$, et le cas 3) se produit pour f' et $\tau(f')(A) = f'(A)$.

Donc $\tau(f') = f'$ et f' est un modèle de P.

Puisque f' et v ne prennent jamais la valeur I, ce sont des points fixes maximaux de τ, et d'autre part $f = inf(v,f')$.

On a:

$f \leq opt(\tau) = max\{g \in PF(\tau), g \leq inf(MAX(\tau))\} \leq inf(MAX(\tau)) \leq inf(v,f') = f$.

Donc $pppf(\tau) = opt(\tau)$. ♦

6.4 LES MODELES OPTIMAUX

Dans tout ce paragraphe, on se place dans le cas d'un programme trivalué consistant.

On rappelle la fonctionnelle utilisée pour les programmes trivalués avec négation:

T_{Pr}: IHT(L) $\cup$ {Contra} $\rightarrow$ IHT(L) $\cup$ {Contra }.

Si i $\in$ IHT(L), alors:

T_{Pr}(i) = {lit $\in$ Her(L)/ il existe une instance fermée lit $\leftarrow$ for d'une règle de Pr telle que: vv_i(for) = Vrai},

si cet ensemble ne contient pas un atome et sa négation,

T_{Pr}(i) = Contra sinon;

T_{Pr}(Contra) = Contra.

L'ensemble considéré est l'ensemble: IHT(L) $\cup$ {Contra}; IHT(L) étant l'ensemble des fonctions de her(L) dans {V,F,I}, i.e. des fonctions partielles de her(L) dans {V,F} (prenant la valeur I, là où elles ne sont pas définies), pouvant aussi être étendues à des fonctions partielles de For(L) dans {V,F} (la valeur de vérité d'une formule étant définie à partir de l'interprétation trivaluée).
L'ordre sur IHT(L) étant l'ordre associé à I $\leq$ V, I $\leq$ F, l'ordre sur IHT(L) $\cup$ {Contra} est celui sur IHT(L) avec d'autre part, $\forall$i $\in$ IHT(L), i $\leq$ Contra.

Lemme 6.4.1 *a) Deux interprétations i et j de IHT(L) sont consistantes si $\forall$lit $\in$ Her(L), i(lit) $\neq$ I et j(lit) $\neq$ I $\Rightarrow$ i(lit) = j(lit).*

b)Tout sous ensemble I non vide et consistant de IHT(L) admet une borne sup: $\bigcup_{i\in I} i$.

c)Tout sous ensemble I non vide de IHT(L) admet une borne inf: $\bigcap_{i\in I} i$.

Démonstration a) Par définition.

b) Si I est consistant, $\bigcup_{i \in I} i$ est une interprétation car cet ensemble ne contient pas un atome et sa négation.
c) Clair. ♦

On redéfinit les notions du paragraphe 6.2, en se restreignant à l'ensemble IHT(L) puisque l'on est dans le cas d'un programme consistant et on introduit les notions analogues liées aux modèles d'un programme trivalué.

Définitions 6.4.1
L'ensemble des points fixes de T_{Pr} est:

$$PF(T_{Pr}) = \{f \in IHT(L) / f = T_{Pr}(f)\}.$$

L'ensemble des points fixes de T_{Pr} consistants avec tous les points fixes de T_{Pr} est:

$$PFC(T_{Pr}) = \{f \in PF(T_{Pr}) / f \text{ est consistant avec tout point fixe de } T_{Pr}\}.$$

L'ensemble des points fixes maximaux de T_{Pr} est:

$$MAX(T_{Pr}) = \{f \in PF(T_{Pr}) / \forall g \in PF(T_{Pr}), f \leq g \Rightarrow f = g\}.$$

L'ensemble des pré-points fixes de T_{Pr} est:

$$PRE(T_{Pr}) = \{f \in IHT(L) / f \leq T_{Pr}(f)\}.$$

L'ensemble des post-points fixes de T_{Pr} est:

$$POST(T_{Pr}) = \{f \in IHT(L) / T_{Pr}(f) \leq f\}.$$

Lemme 6.4.2 *Pr étant consistant, tous ces ensembles sont non vides.*

Démonstration Pr est consistant $\Leftrightarrow$ $pppf(T_{Pr}) \neq$ Contra, donc $PF(T_{Pr})$, $PFC(T_{Pr})$, $PRE(T_{Pr})$, $POST(T_{Pr})$ sont non vides, car $pppf(T_{Pr})$ est élément de tous ces ensembles.

Comme $PRE(T_{Pr}) \neq \emptyset$, $MAX(T_{Pr}) \neq \emptyset$, d'après le lemme 6.2.3. ♦

On a vu précédemment que:

$$f \text{ est un modèle de Pr} \Leftrightarrow f \in \mathrm{IHT}(L) \text{ et } T_{Pr}(f) \leq f$$

Ce qui nous amène à définir de manière analogue, les notions de modèles maximaux, consistants et optimal.

Définitions 6.4.2
L'ensemble des modèles de Pr est:

$$\mathrm{MOD}(Pr) = \{f \in \mathrm{IHT}(L) / \ T_{Pr}(f) \leq f\} = \mathrm{POST}(T_{Pr}).$$

L'ensemble des modèles de Pr consistants avec tous les modèles de Pr est:

$$\mathrm{MODC}(Pr) = \{f \in \mathrm{POST}(T_{Pr}) / \ f \text{ est consistant avec tout post-point fixe de } T_{Pr}\}.$$

L'ensemble des modèles maximaux de Pr est:

$$\mathrm{MODMAX}(Pr) = \{f \in \mathrm{POST}(T_{Pr}) / \ \forall g \in \mathrm{POST}(T_{Pr}), f \leq g \Rightarrow f = g\}.$$

Lemme 6.4.2 *MOD(Pr) et MODC(Pr) admettent un plus petit élément égal à pppf(T_{Pr}), et sont par conséquent non vides.*

Démonstration Il suffit de démontrer que le plus petit post-point fixe de T_{Pr} est égal à son pppf et ce sera terminé car alors $\forall g \in \mathrm{POST}(T_{Pr})$, $\mathrm{pppf}(T_{Pr}) \leq g$ et donc $\mathrm{pppf}(T_{Pr})$ sera consistant avec tout post-point fixe.

Or on a le lemme suivant:

Lemme 6.4.3 *Si S est un ensemble vérifiant les conditions du théorème du point fixe, et τ une fonctionnelle monotone de S dans S, alors τ admet un plus petit post-point fixe égal à son pppf.*

Démonstration $\mathrm{pppf}(\tau) = \tau\uparrow\alpha$ pour un certain ordinal α.
$\tau\uparrow 0 = \perp$ le plus petit élément de S.

Or $\forall i$ post-point fixe de τ , on a $\forall\alpha$ ordinal, $\tau\uparrow\alpha \subseteq i$, par induction transfinie.

$\alpha = 0$, c'est vrai.

Si α est un ordinal successeur, $\tau\uparrow\alpha = \tau(\tau\uparrow\alpha\text{-}1) \subseteq \tau(i) \subseteq i$, par monotonie de τ, induction et parce que i est un post-point fixe.

Si α est un ordinal limite, $\tau\uparrow\alpha = \sup\{\tau\uparrow\gamma, \gamma < \alpha\} \subseteq i$, par induction.

Comme $\mathrm{pppf}(\tau) = \tau\uparrow\alpha$ pour un certain ordinal α, $\tau\uparrow\alpha$ est un point fixe, c'est un post-point fixe, et c'est donc le plus petit. ♦

Pour montrer que MODMAX(Pr) $\neq \varnothing$, on utilisera la proposition suivante:

Proposition 6.4.1 *$\forall M \in MOD(Pr)$, $\exists M' \in MODMAX(Pr)$, $M \subseteq M'$.*

Démonstration Soit $M \in$ MOD(Pr).
Soit $A_M = \{M' \in MOD(Pr), M \subseteq M'\}$, $A_M \neq \varnothing$, car $M \in A_M$.

Tout sous ensemble S totalement ordonné de A_M admet un majorant dans A_M, $\cup_{M' \in S} M'$.

En effet $\cup_{M' \in S} M' \in IHT(L)$ car S est consistant, c'est un modèle de Pr car

$T_{Pr}(\cup_{M' \in S} M') = \cup_{M' \in S} T_{Pr}(M') \subseteq \cup_{M' \in S} M'$. De plus $M \subseteq \cup_{M' \in S} M'$.

Donc A_M admet un élément maximal par le lemme de Zorn. Cet élément maximal est un modèle maximal de Pr qui contient M. ♦

Corollaire 6.4.1 *MODMAX(Pr) $\neq \varnothing$ et admet une borne inf lmax(Pr) qui est un modèle de Pr.*

Démonstration Pr étant consistant, MOD(Pr) $\neq \varnothing$ et d'après la proposition ci-dessus, MODMAX(Pr) $\neq \varnothing$.

Comme MODMAX(Pr) $\neq \varnothing$, MODMAX(Pr) admet une borne inf ($\cap_{M \in MODMAX(Pr)} M$, lemme 4.1 c)).

On peut voir de deux manières que lmax(Pr) $\in$ MOD(Pr):

- Soit en utilisant la définition de la borne inf:
$\forall g \in$ MODMAX(Pr), lmax(Pr) $\leq g$.
Donc $\forall g \in$ MODMAX(Pr), $T_{Pr}(\mathrm{lmax}(Pr)) \leq T_{Pr}(g).\leq g$, par monotonie de T_{Pr} et parce que g est un post-point fixe de T_{Pr}.
Donc $T_{Pr}(\mathrm{lmax}(Pr)) \leq \mathrm{lmax}(Pr)$, par définition de la borne inf.

- Soit en utilisant directement le fait qu'une intersection quelconque de modèles d'un programme trivalué est encore un modèle de ce programme. ♦

Lemme 6.4.4 *MODC(Pr) admet un plus grand élément égal à $\cup_{M \in MODC(Pr)} M$ et appelé opt(Pr).*

Démonstration En effet $\cup_{M \in MODC(Pr)} M \in$ IHT(L) car MODC(Pr) est consistant, c'est un modèle de Pr car:

$T_{Pr}(\cup_{M \in MODC(Pr)} M) = \cup_{M \in MODC(Pr)} T_{Pr}(M) \subseteq \cup_{M \in MODC(Pr)} M.$
Il est consistant.

En effet, soit f un modèle de Pr, si $\cup_{M \in MODC(Pr)} M(lit) \neq I$ et $f(lit) \neq I$,

$\exists M_0 \in$ MODC(Pr) tel que $M_0(lit) \neq I$ et $M_0(lit) = \cup_{M \in MODC(Pr)} M(lit)$ par définition de la borne sup.

Comme M_0 est consistant, $M_0(lit) = f(lit) = \cup_{M \in MODC(Pr)} M(lit)$. ♦

Lemme 6.4.5 *Si $g \in$ MODMAX(Pr) et $f \in$ MOD(Pr), soit $f \subseteq g$, soit f et g ne sont pas consistants.*

Démonstration Si f et g sont consistants, sup(f,g) existe et vaut $f \cup g$; c'est un modèle comme union de deux modèles consistants. Si non($f \subseteq g$), $f \cup g \neq g$ et g ne serait pas maximal. ♦

Proposition 6.4.2 *opt(Pr) = max(MODC(Pr)) = inf(MODMAX(Pr)) = lmax(Pr).*

Démonstration On démontre d'abord que opt(Pr) $\subseteq$ lmax(Pr);

opt(Pr) est un modèle de Pr consistant avec n'importe quel modèle de Pr et d'après le lemme 6.4.5, on a: $\forall g \in$ MODMAX(Pr), opt(Pr) $\subseteq$ g, donc opt(Pr) $\subseteq$ lmax(Pr) par définition de inf(MODMAX(Pr)).

Pour montrer que lmax(Pr) $\subseteq$ opt(Pr), on montre que lmax(Pr) est consistant, ce qui termine la démonstration puisque lmax(Pr) est un modèle de Pr et opt(Pr) est le plus grand modèle consistant.

Soit f un modèle de Pr, alors $\exists g \in$ MODMAX(Pr), $f \subseteq g$, d'après la proposition 6.4.1. Comme lmax(Pr) $\subseteq$ g, f et lmax(Pr) sont consistants. ♦

6.5 LIENS ENTRE TOUTES CES NOTIONS POUR LES PROGRAMMES TRIVALUES

On se pose les questions suivantes que l'on résout par des contre-exemples.

1) A-t-on opt(Pr) = ppmt(Pr) (= pppf(T_{Pr}) = pppostpf(T_{Pr}) (plus petit post point fixe de T_{Pr}))?

2) A-t-on MAX(T_{Pr}) $\subseteq$ MODMAX(Pr)?

3) A-t-on inf(MAX(T_{Pr})) = inf(MODMAX(Pr))?

4) A-t-on max(PFC(T_{Pr})) = max(MODC(Pr))?

Remarque: Ces deux questions ne sont pas semblables car:
inf(MAX(T_{Pr})) $\neq$ max(PFC(T_{Pr})) puisque inf(MAX(T_{Pr})) n'est pas toujours un point fixe de T_{Pr} et l'on a: max(PFC(T_{Pr})) =
max{f $\in$ PF(T_{Pr}), f $\subseteq$ inf(MAX(T_{Pr}))} alors que inf(MODMAX(Pr)) = max(MODC(Pr)).

5) Les modèles maximaux sont ils des modèles bivalués?

6) A-t-on opt(T_{Pr}) = pppf(T_{Pr})?

7) A-t-on PFC(T_{Pr}) = MODC(Pr) $\cap$ PF(T_{Pr})?

8) A-t-on MAX(T_{Pr}) = MODMAX(Pr) $\cap$ PF(T_{Pr})?

- On n'a pas opt(Pr) = ppmt(Pr).

Dans toute la suite, p, q, et r désignent des symboles de prédicats, x, y, ..., des variables et a une constante.

Soit Pr_1 le programme trivalué suivant:

$$\begin{array}{l} p(x) \leftarrow \neg q(x) \\ p(a) \leftarrow q(x) \\ q(x) \leftarrow \neg p(x) \\ \neg q(x) \leftarrow \neg p(a) \end{array}$$

L'ensemble des modèles trivalués de Pr_1 ou l'ensemble des post-points fixes de T_{Pr_1} est le suivant:

$MOD(Pr_1) = \{\varnothing, \{p(a)\}, \{p(a),q(a)\}, \{p(a),\neg q(a)\}\}$.

L'ensemble des points fixes de T_{Pr_1} se réduit au singleton ensemble vide.
$PF(T_{Pr_1}) = \{\varnothing\}$.

En effet, les points fixes de T_{Pr_1} sont en particulier des post-points fixes et: $T_{Pr_1}(\varnothing) = \varnothing$, $T_{Pr_1}(\{p(a)\}) = \varnothing$, $T_{Pr_1}(\{p(a),q(a)\}) = \{p(a)\}$,
$T_{Pr_1}(\{p(a),\neg q(a)\}) = \{p(a)\}$.

L'ensemble des modèles maximaux de Pr_1 est:
$MODMAX(Pr_1) = \{\{p(a), q(a)\}, \{p(a),\neg q(a)\}\}$.

$opt(Pr_1) = \{p(a)\}$ alors que $ppmt(Pr_1) = pppf(T_{Pr_1}) = \varnothing$.

De plus $inf(MODMAX(Pr_1)) = \{p(a)\}$ alors que $inf(MAX(T_{Pr_1})) = \varnothing$. On a donc répondu à 3).

$MAX(T_{Pr_1}) = \{\varnothing\}$ n'est pas inclus dans $MODMAX(Pr_1)$. On a donc répondu à 2) et à 8).

De plus on a $max(PFC(T_{Pr_1})) = \varnothing$ alors que $max(MODC(Pr_1)) = \{p(a)\}$; on a donc répondu à 4).

Le treillis des modèles de Pr_1 est le suivant:

$\{p(a), q(a)\}$ $\qquad\qquad$ $\{p(a), \neg q(a)\}$

$\{p(a)\}$

$\varnothing$

- Pour répondre à la question 5), il suffit de considérer le programme Pr_2 suivant:

$$p(x) \leftarrow \neg p(x)$$

Pr_2 possède un seul modèle trivalué $\varnothing$ qui est donc maximal, alors qu'il n'est pas bivalué.

- Pour répondre à la question 6), il suffit de considérer le programme Pr_3 suivant:

$p(x) \leftarrow \neg q(x)$
$p(a) \leftarrow q(x)$
$q(x) \leftarrow \neg p(x)$
$\neg q(a) \leftarrow \neg p(a)$
$p(x) \leftarrow p(x)$

Le treillis des modèles de Pr_3 est le suivant:

$\{p(a), q(a)\}$ $\{p(a), \neg q(a)\}$

$\{p(a)\}$

$\varnothing$

On a: $MOD(Pr_3) = \{\varnothing, \{p(a)\}, \{p(a), q(a)\}, \{p(a), \neg q(a)\}\}$.
$PF(T_{Pr_3}) = \{\varnothing, \{p(a)\}\}$.
$opt(Pr_3) = \{p(a)\}$.
$MODC(Pr_3) = \{\varnothing, \{p(a)\}\}$.
$PFC(T_{Pr_3}) = \{\varnothing, \{p(a)\}\}$.
$MODMAX(Pr_3) = \{\{p(a), q(a)\}, \{p(a), \neg q(a)\}\}$.
$opt(T_{Pr_3}) = \{p(a)\} \neq pppf(T_{Pr_3}) = \varnothing$.
Cet exemple répond aussi à la question 8).

$PF(T_{Pr_3}) \cap MODMAX(Pr_3) = \varnothing$. Or $MAX(T_{Pr_3}) = \{\{p(a)\}\}$.

Par contre, on a toujours l'inclusion:

$MODMAX(Pr) \cap PF(T_{Pr}) \subseteq MAX(T_{Pr})$.

En effet, si $f \in MODMAX(Pr) \cap PF(T_{Pr})$, f est un point fixe de T_{Pr} et si $g \in PF(T_{Pr})$, est tel que $f \leq g$, comme $g \in POST(T_{Pr})$, $f = g$.

La réponse à la question 7) est non.

On a bien l'inclusion $MODC(Pr) \cap PF(T_{Pr}) \subseteq PFC(T_{Pr})$ car si f est consistant avec n'importe quel post-point fixe, il l'est avec n'importe quel point fixe.
On n'a pas l'inclusion réciproque. En effet, si $f \in PFC(T_{Pr})$, $f \in PF(T_{Pr})$ et si $g \in POST(T_{Pr})$, l'ensemble $\{\alpha \in PF(T_{Pr}), \alpha \leq g\}$ admet un plus

grand élément h, avec lequel f est consistant. Mais g(lit) $\neq$ I et f(lit) $\neq$ I n'implique pas h(lit) $\neq$ I donc on ne peut pas conclure.

Sur l'exemple suivant, on voit que cette inclusion est fausse.

Soit Pr_4 le programme suivant:

$$\begin{array}{l} p(x) \leftarrow \neg q(x) \\ p(x) \leftarrow q(x) \\ q(x) \leftarrow \neg p(x) \\ \neg q(x) \leftarrow \neg p(x) \\ r(x) \leftarrow r(x) \end{array}$$

Le treillis des modèles de Pr_4 est le suivant:

{p(a), ¬r(a), ¬q(a)} {p(a), r(a), ¬q(a)}

{p(a), r(a), q(a)} {p(a), ¬r(a), q(a)}

{p(a), ¬q(a)}   {p(a), r(a)}   {p(a), ¬r(a)}   {p(a), q(a)}

{r(a)}   {p(a)}   {¬r(a)}

∅

On a: $MOD(Pr_4)$ = {∅, {p(a)}, {r(a)}, {¬r(a)}, {p(a), q(a)}, {p(a), ¬q(a)}, {p(a), r(a)}, {p(a), ¬r(a)}, {p(a), q(a), r(a)}, {p(a), ¬q(a), r(a)}, {p(a), ¬q(a), ¬r(a)}, {p(a), q(a), ¬r(a)}}.

$PF(T_{Pr_4})$ = {∅, {r(a)}}.

$opt(Pr_4)$ = {p(a)}.

$MODC(Pr_4)$ = {∅, {p(a)}}.
$PFC(T_{Pr_4})$ = {∅, {r(a)}}.
$MODMAX(Pr_4)$ = {{p(a), q(a), r(a)}, {p(a), ¬q(a), r(a)},
{p(a), ¬q(a), ¬r(a)}, {p(a), q(a), ¬r(a)}}.
$opt(T_{Pr_4})$ = {r(a)} $\neq$ $pppf(T_{Pr_4})$ = ∅.

Donc $PFC(T_{Pr_4}) = \{\varnothing, \{r(a)\}\}$ n'est pas inclus dans $MODC(Pr_4) \cap PF(T_{Pr_4}) = \{\varnothing\}$.

Sur cet exemple, on a aussi: $opt(T_{Pr_4}) = \{r(a)\} \neq pppf(T_{Pr_4}) = \varnothing$.
Sur cet exemple, on voit aussi que $opt(T_{Pr_4}) = \{r(a)\}$ n'est pas inclus dans $opt(Pr_4) = \{p(a)\}$.

BIBLIOGRAPHIE

[1] K. Apt, H.Blair and A.Walker, Towards a theory of declarative knowledge, in: J. Minker, ed, *Foundations of Deductive Databases and Logic Programming* (Morgan Kaufman, Los Altos, 1988), 89-142.

[2] K. Apt and M. Van Emden, Contribution to the theory of logic programming, *J. ACM*, **29-3**, (1982), 841-842.

[3] M. Ben Jacob and M. Fitting, Stratified and Three-valued Logic Programming Semantics, in: R.A. Kowalski and A. Bowen, ed., *Proceedings of the Fifth International Conference and symposium on Logic Programming* (MIT Press, 1988) 1055-1069.

[4] H.Blair, The recursion theoretic complexity of the semantics of predicate logic as a programming language, *Information and control*, **54**, (1982), 25-47.

[5] S. Blamey, Partial Logic, in: D. Gabbay and F. Guenthner, ed., *Handbook of Philosophical logic*, **vol 3**, (1986), 1-70.

[6] D. Bochvar, On three-valued logical calculus and its application to the analysis of contradictions, *Matematiceskij Sbornik*, **4**, (1939), 353-369.

[7] K.L. Clark, Negation as failure, in: Gallaire and Minker ed., *Logic and databases* (Plenum Press, New York,1978).293-324.

[8] J.P. Delahaye: Programmation en logique trivaluée, publication interne I.T n°115, Université des Sciences et Techniques de Lille, 1987.

[8 Bis] J.P. Delahaye et P. Matthieu : Logique partielle et Prolog, publication interne I.T n°166, Université des Sciences et Techniques de Lille, 1987.

[9] M. Fitting, A Kripke-Kleene semantic for logic programs, *J. Logic Programming*, **4** (1985) 295-312.

[10] M. Fitting, *Computability theory, Semantics and Logic Programming*, (Oxford University Press, New York, 1987).

[11] M. Fitting, Logic Programming on a topological bilattice, *Fundamenta Informatica*, 1987.

[12] M. Fitting, Logic Programming semantics using a compact data structure.

[13] M. Fitting, Notes on the mathematical aspects of Kripke 's theory of truth, *Notre Dame Journal of Formal Logic*, vol.**27**, n°**1** (1986) 75-88.

[14] M. Fitting, Partial models and logic programming, *Theoretical Computer Science*, **48**, (1987), 229-255.

[15] M. Fitting, Pseudo Boolean valued Prolog, *Studia Logica*, (1987).

[16] D.M. Gabay and J. Sergot, Negation as Inconsistency, *Journal of Logic Programming* **1** (1986) 1-35.

[17] S. Haack, *Philosophy of Logics* (Cambridge University Press, Cambridge, 1978).

[18] S. Kripke, Outline a thery of truth, *Journal of Philosophy*, **vol 72** (1975) 690-716.

[19] K. Kunen, Negation in Logic Programming, *Journal of Logic Programming* (1987) 289-308.

[20] K. Kunen, Some remarks on the completed databases, 979-991.

[21] J.L. Lassez et M.J. Maher, Optimal fixpoints of logic programs, *Theoretical Computer Science* **39** (1985) 15-25.

[22] J.W. Lloyd, *Foundations of Logic Programming* (second edition, Springer, Berlin, 1987).

[23] J. Lukasiewicz, Many valued systems of propositional logic, in: Mc Call ed., *S. Polish Logic* (Oxford University Press, 1967).

[24] J. Mc Carthy, Circumscription- a form of non monotonic reasoning, *Journal of Artificial Intelligence*, **13** (1980) 27-39.

[25] M.J. Maher, Equivalences of logic programs, *ICLP*, (1987) 410-423.

[26] P. Mancarella, S. Martini and D. Pedreschi, Complete Logic Programs with Domain-Closure axiom, *Journal of Logic Programming* **5** (1988) 263-276.

[27] Z. Manna and A. Shamir,The optimal approach to recursive programs, *ACM* vol **20**, n° **11** (1977) 824-831.

[28] Z. Manna and A. Shamir, The theoretical aspects of the optimal fixpoints, *Siam J. comput.*, vol **5**, n°**3** (1976).414-426.

[29] Miller, Nadhatur and Scedrov, Hereditary Harrop Formulas and uniform proof system, *LICS*, (1987).

[30] A. Mycroft, Logic programs and many-valued logic,in: *STACS 84*, Lecture Notes in Computer Science **166** (Springer, Berlin, 1984) 274-286.

[31] T. Przymunsinski, Non monotonic formalisms and Logic Programming, *ICLP*, (1989) 655-674.

[32] T. Przymunsinski, On the declarative semantics of deductive databases and logic programs, in: J. Minker, ed, *Foundations of Deductive Databases and Logic Programming* (Morgan Kaufman, Los Altos, 1988).

[33] J.B. Rosser and A.R. Turquette, *Many-valued logic*, (North Holland, Amsterdam,1958).

[34] J.C. Shepherdson, Negation in Logic Programming, in: J. Minker, ed, *Foundations of Deductive Databases and Logic Programming* (Morgan Kaufman, Los Altos, 1988) 19-87.

[35] J.C. Shepherdson, A sound and complete semantics for negation asfailure,1-42.

[36] V. Thibau, Expressivité des connecteurs en logique trivaluée, Publication interne I.T à paraitre, Université des Sciences et Techniques de Lille, (1989).

[37] V. Thibau, Modèles optimaux en logique trivaluée, Publication interne I.T à paraitre, Université des Sciences et Techniques de Lille, (1989).

[38] V. Thibau, 3 théorèmes de complétude en logique trivaluée, Publication interne I.T à paraitre, Université des Sciences et Techniques de Lille, (1989).

[39] L. Thorne Mac Carty, Fixedpoints semantics, *Journal of Logic Programming*, (1988) 3-31.

[39] R. Turner, *Logiques pour l'intelligence artificielle*, Masson, Paris, (1987).

[40] A. Urquhart, Many valued logic Logic, in: D. Gabbay and F. Guenthner, ed., *Handbook of Philosophical logic*, **vol 3**, (1986), 71-115.

[42] M.H. Van Emden, Quantitative deduction and its fixpoints theory, *Journal of Logic Programming* **1** (1986) 37-53.

[43] M.H. Van Emden, R.A. Kowalski, The semantics of predicate logic as a programming language, *J. ACM*, **23** (1976), 733-742.

[44] A. Van Gelder, Negation as failure using tight derivations, in: J. Minker, ed, *Foundations of Deductive Databases and Logic Programming* (Morgan Kaufman, Los Altos, 1988).

[45] A. Yasuhara, *Theory of recursive functions*. (Academic Press, New York, 1971).