Cours de tronc commun du DEA d'informatique d'Evry

Fondements logiques pour les méthodes formelles

Année 1999-2000

Marc Aiguier, Gilles Bernot

Chapitre 1 : Introduction

- Logiques et méthodes formelles
- Syntaxe, sémantique et preuves
- Correction et complétude

Chapitre 2 : Définitions générales

- Les logiques générales
  - Sémantique : les institutions
  - Preuves : les systèmes d'inférence
  - Propriétés classiques
- Les systèmes formels
  - Définitions
  - Preuves, déductions et leurs propriétés
  - Arbre de preuve
  - Propriétés classiques
- Un exemple simple

Chapitre 3 : La logique propositionnelle

- Syntaxe, sémantique et calcul de Hilbert
  - Théorème de la déduction
  - Correction du calcul de Hilbert
- Complétude du calcul de Hilbert
- Déduction naturelle et Calcul des séquents

Chapitre 4 : La logique des prédicats

- Motivation et utilité pour les méthodes formelles
- Syntaxe et sémantique
- Calculs de Hilbert, de la déduction naturelle et des séquents
- Correction et complétude (survol)

Chapitre 5 : Logiques finiment engendrées

- Considérations générales
- Motivation
- Syntaxe et sémantique
- Induction structurelle (la règle poly-signatures)

– Correction, complétude sur les formules closes, non-complétude dans le cas général (survol et exemples)

Chapitre 6 : Logiques modales

– Logique modale propositionnelle de base
– Syntaxe et sémantique générales
– Calculs de style Hilbert : K, T, S4 et S5
– Autres logiques : temporelles et autres

Chapitre 1 : Introduction

Parmi les soucis majeurs de l'informatique, la *correction* des systèmes développés se trouve en bonne place, particulièrement lorsque les « bugs » peuvent avoir des conséquences graves. Les *méthodes formelles* sont de plus en plus imposées parmi les éléments majeurs des normes de qualité pour les composants logiciels dit « critiques ». Elles ont pour objectif de démontrer par des techniques mathématiquement fondées et vérifiables par ordinateur que les composants logiciels concernés répondent logiquement à leur spécifications.

Le seul fait de vouloir appliquer des méthodes formelles dans cet objectif a des conséquences dont il est important d'être bien conscient :

1. Il faut expliciter les propriétés que l'on veut établir par ces méthodes formelles. Pour que le procédé de validation de ces propriétés soit vérifiable par ordinateur, il faut que leur énoncé lui-même soit mis sous forme directement « compréhensible » par un ordinateur, c'est-à-dire sous forme symbolique. La collection de tous les énoncés à établir est généralement appelée une *spécification formelle*. La définition des suites de symboles reconnaissables pour une méthode formelle donnée relève de la *syntaxe* et la définition de syntaxes correctes conduit tout naturellement à définir des *langages de spécification*. L'énoncé d'une propriété selon un langage de spécification donné est une *formule*.
2. L'énoncé symbolique des propriétés ne se suffit pas à lui seul, puisqu'il n'est qu'une suite de caractères et de symboles. Il faut être à même de lui donner un sens correspondant « dans le monde réel » au phénomène que l'on veut garantir par cette propriété. Cela conduit nécessairement à représenter le monde réel (en fait seulement la partie qui nous intéresse pour un système donné) sous une forme rigoureuse, c'est-à-dire un (ou des) modèle(s) mathématique(s), afin que cette propriété symbolique puisse porter rigoureusement sur quelque chose. Il faut ensuite être apte à dire si une certaine propriété est, ou n'est pas, vérifiée dans un modèle. Cette partie de la question s'appelle la *sémantique*.
3. Enfin on peut alors envisager d'appliquer des méthodes formelles. La question qui se pose est alors « *comment aboutir par ordinateur à démontrer la suite de symboles qu'est une propriété ?* » et la seule chose que sache faire un ordinateur étant justement de manipuler des suites de symboles, la réponse ne devrait pas poser de difficultés majeures. Il suffit de définir les conditions de manipulations des formules et comment elles peuvent conduire à obtenir une propriété. On appelle généralement ces manipulations des suites d'inférences, et on parle alors de *système d'inférence* ou encore de *calcul*.

Ces trois domaines (syntaxe, sémantique et calcul) sont loins d'être indépendants. La syntaxe définit en grande partie le domaine dans lequel travaillent la sémantique d'une part et le calcul d'autre part. Le point le plus important est que le calcul, qui n'est *a priori* qu'une suite de transformations syntaxiques sans aucun sens, soit compatible avec, et reflète aussi fidèlement que possible, « le monde réel ». Faute de pouvoir mixer directement monde réel et manipulations syntaxiques, on demande tout naturellement que la syntaxe soit en accord avec une vision mathématique du monde réel : la sémantique.

Cet accord se traduit naturellement en deux propriétés fondamentales, la première absolument nécessaire et la seconde seulement souhaitable :

1. Tout ce que le calcul permet de prouver doit être vrai « dans le monde réel », c'est-à-dire que

toute formule que les règles d'inférence permettent de produire doivent être satisfaites par les modèles de la sémantique. Cette propriété est la *correction* du calcul.

2. En pratique, on sait bien qu'il existe des propriétés vraies dans le monde réel qui ne sont pas démontrables formellement, mais si l'on s'intéresse à une vision du monde suffisamment pauvre pour que la sémantique le permette, il arrive parfois que tout ce qui est vrai dans tous les modèles soit démontrable par le calcul. Cette propriété est la *complétude* du calcul.

Dans tous les cas, il faut bien retenir que proposer un calcul sans définir en quel sens il est correct (i.e. ne pas fournir de sémantique) enlève *de facto* toute crédibilité aux méthodes formelles. Même un singe pourrait écrire des transformations symboliques au hasard pour former un calcul, mais que prouveraient les inférences qui en résultent ? C'est seulement une fois établie la correction des règles par rapport à la sémantique que l'on peut faire confiance à un calcul formel. Seulement alors, on peut s'appuyer sur une compréhension intuitive de la sémantique pour guider les preuves formelles.

Chapitre 2 : Définitions générales

1 Les logiques générales

Une *logique générale* est justement la définition des ces trois aspects (syntaxe, sémantique, preuves formelles) et on comprend donc pourquoi cette notion est importante en informatique.

En pratique, le volet syntaxique dans une logique repose sur la construction inductive de formules à partir de symboles de fonctions, prédicats, connecteurs logiques, quantificateurs... (cf. cours de mise à niveau en calcul symbolique). On remarque également qu'une logique donnée contient des « parties fixes » (on connaît l'ensemble des quantificateurs qu'elle accepte par exemple, la définition inductive des formules bien formées est bien déterminée, etc.), et qu'elle contient également des « parties utilisateur » que l'on introduit dans un cadre donné, pour résoudre un problème donné (par exemple, si l'on veut raisonner sur les entiers naturels dans une logique donnée, on va se placer dans un langage où l'on dispose des fonctions 0, successeur, $+$, $\times$, etc. alors que si l'on veut appliquer la même logique pour traiter des propriétés sur des files d'attente, on va se munir de prédicats de priorité, d'opérations d'ajout, de retrait, etc.)

La « partie utilisateur » mentionnée ci-dessus est appelée classiquement une *signature*. Elle caractérise les éléments de base spécifiques au problème traité ponctuellement. Ainsi on aura une signature pour les entiers naturels, une autre signature pour les files d'attente, une troisième pour la description d'un problème de gestion de centrale électrique, etc, alors que par contre les quantificateurs ou les connecteurs par exemple seront communs à ces trois sujets dans la logique considérée. En pratique une signature est généralement un ensemble de symboles, mais rien ne l'impose en théorie.

Un autre aspect utile en logique est d'être capable de comparer les signatures. Par exemple, une signature dédiée à écrire des formules sur les entiers naturels sera intuitivement incluse dans une signature dédiée aux entiers relatifs (cette dernière contiendra en plus la notion d'opposé entre autres). Par exemple encore, il peut être utile de renommer certaines opérations en fonction du sujet traité ($\times$ peut devenir $*$ pour des besoins particuliers) ; cela donne lieu à des liens entre les signatures possibles qui méritent d'être considérés (ultérieurement on peut vouloir bénéficier d'une formule prouvée avec $\times$ pour obtenir sa traduction avec $*$ sans refaire la preuve...). Ces liens se traduisent par une notion de *morphismes entre signatures*. En pratique ce sont souvent des applications entre ensembles de symboles, mais rien ne l'impose en théorie.

On peut ainsi considérer l'ensemble de toutes les signatures acceptables pour une logique donnée. Sur cet ensemble de signatures travaillent les morphismes de signatures. Un ensemble d'objets sur lequel travaillent des morphismes est appelé une catégorie. Ici on parle de la catégorie des signatures, puisque les objets sont les signatures.

Lorsque la signature est connue, disons Σ, on peut en déduire précisément quelles sont exactement les formules que l'on peut écrire (les formules bien formées pour la logique considérée au-dessus de la signature Σ). Mieux : étant donné un morphisme de signatures $\sigma : \Sigma_1 \to \Sigma_2$ (intuitivement : une application entre ces deux ensembles de symboles) et étant donnée une formule φ_1 au-dessus de la signature Σ_1, il n'est pas difficile de la « repeindre » en changeant les symboles selon σ. On dit alors que le passage des signatures aux formules transporte les morphismes de signatures.

1.1 Aspect sémantique : les institutions

Une « institution » est en quelque sorte une logique à laquelle manquent tous les mécanismes d'inférence, pour se focaliser sur les rapports entre la syntaxe et la sémantique.

Définition 1 : Une *institution* est un quadruplet $(Sign, For, Mod, \models)$ où

- $Sign$ est un ensemble, dont les éléments sont appelés des signatures, et qui est muni de morphismes entre signatures[1] de la forme $\sigma : \Sigma_1 \to \Sigma_2$.
- For est une application de $Sign$ dans Set (où Set dénote l'ensemble des ensembles[2]) qui transporte les morphismes de signatures. Cela signifie en fait que For dénote deux applications : l'une qui associe des ensembles de formules $F_1 = For(\Sigma_1),\ F_2 = For(\Sigma_2), \cdots$ à chacune des signatures $\Sigma_1,\ \Sigma_2, \cdots$ appartenant à $Sign$, et l'autre qui associe une application $\overline{\sigma} : F_1 \to F_2$ à chaque morphisme de signatures $\sigma : \Sigma_1 \to \Sigma_2$.
- Mod est une application de $Sign$ dans Set qui transporte les morphismes de signatures en les inversant. Cela signifie que Mod dénote deux applications : l'une qui associe à une signature Σ l'ensemble de ses modèles $Mod(\Sigma)$, et l'autre qui associe à chaque morphisme de signatures $\sigma : \Sigma_1 \to \Sigma_2$ une application $U\_\sigma : Mod(\Sigma_2) \to Mod(\Sigma_1)$.
- $\models$ est une famille de prédicats indexée par $Sign$ telle que, pour chaque $\Sigma \in Sign$, le prédicat $\models \_\Sigma$ porte sur $Mod(\Sigma) \times For(\Sigma)$.

$Sign$ contient en fait toutes les signatures imaginables pour traiter tous les problèmes abordables avec la logique considérée. Les morphismes de $Sign$ dénotent, intuitivement, tous les rapports de renommage ou d'inclusion de signatures mentionnés plus haut.

Pour chaque signature, For détermine l'ensemble des formules bien formées au-dessus de cette signature. Pour chaque morphisme de signature σ, l'application $\overline{\sigma}$ est intuitivement le remplacement de symboles qui permet de traduire directement les formules au-dessus de Σ_1 en des formules au-dessus de Σ_2.

On a vu qu'un problème possède une signature et qu'il faut, pour chaque problème, modéliser mathématiquement l'ensemble de tous les comportements possibles du « monde réel » afin de donner une sémantique. Naturellement, il est inenvisageable de modéliser « tout » le monde réel, on ne modélise que ce que les symboles manipulables dans le cadre du problème peuvent en appréhender. C'est pourquoi l'ensemble des modèles possibles dépend de la signature.
Par ailleurs, considérons un morphisme de signatures $\sigma : \Sigma_1 \to \Sigma_2$, par exemple l'inclusion de la signature des entiers naturels (0, *succ*, $+$, $\times$) dans celle des entiers relatifs (0, *succ*, $+$, $\times$, *opposé*, $-$). On peut considérer un modèle au-dessus de Σ_1, c'est-à-dire un élément M de $Mod(\Sigma_1)$, par exemple l'ensemble des entiers naturels $\mathbb{N}$ muni des opérations 0, *succ*, $+$ et $\times$ habituelles. Il n'est pas aisé de faire correspondre à ce modèle un autre modèle au-dessus de (0, *succ*, $+$, $\times$, *opposé*, $-$) car il faudrait inventer ce que peuvent bien faire des opérations comme *opposé* dans $\mathbb{N}$. Par contre, si l'on considère un modèle au-dessus de Σ_2, par exemple l'ensemble des entiers relatifs $\mathbb{Z}$ (muni des opérations habituelles 0, *succ*, $+$, $\times$, *opposé* et $-$), il n'est pas difficile d'oublier que les opérations *opposé* et $-$ ont un sens dans $\mathbb{Z}$. Il est donc facile, par *oubli*, de fabriquer un modèle au-dessus de Σ_1 à partir d'un modèle de Σ_2 (et pas le contraire). C'est pourquoi U_σ va de $Mod(\Sigma_2)$ dans $Mod(\Sigma_1)$, donc « en sens inverse » de σ.

Enfin, étant donnés un modèle (modélisant un « monde imaginable ») et une formule (décrivant une propriété donnée), il est légitime de se demander si le modèle satisfait la formule. Il faut remarquer que pour que ceci ait un sens, il est bien sûr nécessaire que tous les symboles apparaissant dans la formule aient un sens pour le modèle. C'est pourquoi on requiert qu'ils partagent la même signature. Dès lors il faut un prédicat $\models \_\Sigma$ pour chaque signature $\Sigma \in Sign$, et $M \models \_\Sigma\varphi$ signifie que le

[1] Pour les puristes, $Sign$ est en fait une catégorie au sens mathématique du terme.

[2] Pour les mêmes puristes, Set est en fait une *classe* au sens mathématique du terme (rien à voir avec l'OO), tout comme $Sign$ précédemment.

modèle $M \in Mod(\Sigma)$ satisfait la formule $\varphi \in For(\Sigma)$. On appelle $\models \_\Sigma$ le *prédicat de satisfaction* (parfois de *validation*).

1.2 Aspect preuve : les systèmes d'inférence

Définition 2 : Un *système d'inférence* est un triplet $(Sign, For, \vdash)$ où
- $Sign$ ... (comme avant)
- For ... (comme avant)
- $\vdash$ est une famille de prédicats indexée par $Sign$ telle que, pour chaque $\Sigma \in Sign$, le prédicat $\vdash \_\Sigma$ porte sur $\mathcal{P}(For(\Sigma)) \times For(\Sigma)$.

Les prédicats $\vdash \_\Sigma$ représentent les inférences possibles, c'est-à-dire le calcul, dans la logique considérée. De même que pour le prédicat de satisfaction, les inférences se font ici à signature fixe[3]. $\mathcal{P}(For(\Sigma))$ dénote l'ensemble des parties de $For(\Sigma)$, et la notation $\Gamma \vdash \_\Sigma\varphi$ signifie que l'ensemble de formules Γ permet de déduire la formule φ.

1.3 Propriétés classiques

Les notions d'institution ou de système d'inférence recouvrent de nombreuses instances possibles, qui sont autant de logiques particulières pouvant être à leur tour utilisées pour développer des méthodes formelles. La question de choisir une logique plutôt qu'une autre se pose alors, et ce choix est largement guidé par les propriétés qu'ont ou n'ont pas ces logiques. On détaille ci-dessous les propriétés les plus courantes.

Propriétés portant sur les institutions :
- **Condition de satisfaction** :

$$\forall \sigma : \Sigma_1 \to \Sigma_2, \quad \forall \varphi_1 \in For(\Sigma_1), \ \ \forall M_2 \in Mod(\Sigma_2), \quad U\_\sigma(M_2) \models_{\Sigma_1} \varphi_1 \iff M_2 \models_{\Sigma_2} \overline{\sigma}(\varphi_1)$$

  En se fondant sur l'intuition déjà utilisée d'un morphisme σ traduisant une inclusion de Σ_1 dans Σ_2, c'est-à-dire que Σ_2 est « plus riche » que Σ_1, la condition de satisfaction traduit l'idée suivante. Si l'on considère un modèle M_2 sur la signature riche et une formule φ_1 sur la signature pauvre, on peut se demander si d'une façon ou d'une autre M_2 satisfait φ_1. C'est assez légitime puisque M_2 propose une sémantique pour tous les symboles apparaissant dans φ_1. Deux manières d'opérer s'offrent à nous :

  1. oublier les opérations supplémentaires que propose M_2 puis voir si le modèle apauvri $U\_\sigma(M_2)$ satisfait la formule pauvre φ_1,
  2. ou transporter la formule pauvre φ_1 sur la signature riche via $\overline{\sigma}$ et voir si le modèle riche M_2 satisfait la formule $\overline{\sigma}(\varphi_1)$.

  La condition de satisfaction requiert que ces deux façons d'opérer donnent toujours le même résultat, qu'il soit vrai ou faux.
  Aussi utile que puisse paraître cette propriété, elle n'est pas souvent adéquate. Ainsi par exemple dans des logiques que l'on qualifie de finiment engendrées (définies plus loin dans le cours) et si l'on reprend notre exemple d'inclusion des opérations des entiers naturels dans celles des entiers relatifs, on a $U\_\sigma(\mathbb{Z}) = \mathbb{N}$. La propriété φ_1 suivante

$$\forall x, \ x \geq 0$$

[3]On verra dans le chapitre sur l'induction structurelle que certaines preuves peuvent néanmoins être menées « à cheval » sur plusieurs signatures.

est fausse dans $M_2 = \mathbb{Z}$ mais vraie dans $U\_\sigma(M_2) = \mathbb{N}$ (et réciproquement pour sa négation). Les logiques finiment engendrées ne vérifient donc pas la condition de satisfaction.

[Fin du premier cours ici]

Propriétés portant sur les systèmes d'inférence :

- **Reflexivité** :

$$\{\varphi\} \vdash \varphi$$

À partir d'une formule, on peut toujours en déduire elle-même. On imagine mal une logique n'ayant pas cette propriété.

- **Transitivité** :

$$\text{si } \Gamma \vdash \Gamma' \text{ et si } \Gamma \cup \Gamma' \vdash \varphi \text{ alors } \Gamma \vdash \varphi$$

Cette propriété traduit l'idée de pouvoir mener les preuves en passant par des lemmes intermédiaires : pour prouver φ à partir de Γ, on commence par prouver un ensemble de lemmes Γ', et on l'utilise pour prouver φ. On voit mal comment il serait possible de mener une preuve avec une logique qui n'aurait pas cette possibilité. Nota : la notation $(\Gamma \vdash \Gamma')$ est un abus de notation pour $(\forall \psi \in \Gamma',\ \Gamma \vdash \psi)$.

- **Monotonie** :

$$\text{si } \Gamma \vdash \varphi \text{ et si } \Gamma \subseteq \Gamma' \text{ alors } \Gamma' \vdash \varphi$$

Cette propriété signifie que si l'on peut mener une preuve de φ à partir d'un ensemble Γ d'hypothèses, alors on peut également (*a fortiori*) mener une preuve de φ à partir de plus d'hypothèses. Certaines logiques ne possèdent pas cette propriété. C'est par exemple le cas des logiques des défauts où une formule donnée n'est vraie que « par défaut ». Exemple : si Γ affirme que `les oiseaux volent`, que `les pingouins sont des oiseaux` et que `titi est un pingouin`, on pourra en déduire que `titi vole`. Toutes ces formules sont à comprendre avec la réserve *par défaut*. Ainsi si Γ' contient la formule supplémentaire `les pingouins ne volent pas`, alors on ne peut plus en déduire `titi vole`. Les logiques des défauts sont par exemple utilisées en informatique pour manipuler des spécifications avec traitement d'exceptions et « handler » d'exceptions. Elles sont également utilisées pour l'apprentissage dans le domaine de l'intelligence artificielle.

- **$\vdash$-translation** :

$$\text{si } \Gamma \vdash \_\Sigma\varphi \text{ et si } \sigma : \Sigma \to \Sigma' \text{ alors } \overline{\sigma}(\Gamma) \vdash_{\Sigma'} \overline{\sigma}(\varphi)$$

Cette propriété est en grande partie la version « calcul » de la propriété sémantique que nous avons appelée la condition de satisfaction. Elle n'est donc pas plus adéquate en pratique. Elle signifie que si Γ et φ sont respectivement des hypothèses et formule « pauvres » alors toute preuve menée directement au-dessus de la signature pauvre peut être menée au-dessus de la signature riche, en traduisant Γ et φ dans la signature riche puis en menant la preuve au-dessus de cette signature. Un contre-exemple sur la même formule :

$$\forall x,\ x \geq 0$$

Cette formule est démontrable par récurrence sur la signature des entiers naturels, alors qu'elle n'est pas démontrable sur la signature des entiers relatifs (on n'arrive pas à prouver que si $x \geq 0$ alors $\mathit{opposé}(x) \geq 0$).

Propriétés portant sur les deux (institution et système d'inférence réunis) :

- **Correction** :

$$\forall \Gamma \subset For(\Sigma),\ \forall \varphi \in For(\Sigma),\ (\Gamma \vdash \varphi) \Longrightarrow (\forall M \in Mod(\Sigma),\ M \models \Gamma \Longrightarrow M \models \varphi)$$

- **Complétude** :

$$\forall \Gamma \subset For(\Sigma),\ \forall \varphi \in For(\Sigma),\ (\forall M \in Mod(\Sigma),\ M \models \Gamma \Longrightarrow M \models \varphi) \Longrightarrow (\Gamma \vdash \varphi)$$

Bref :

Définition 3 : Une *logique générale* est un quintuplet $(Sign, For, Mod, \models, \vdash)$ tel que :

- $(Sign, For, Mod, \models)$ est une institution
- $(Sign, For, \vdash)$ est un système d'inférence
- Pour chaque signature de $Sign$, le système d'inférence est correct par rapport au prédicat de satisfaction (i.e. propriété de correction définie précédemment).

La définition habituelle d'une logique générale (celle que l'on trouve dans tous les articles sur le sujet) impose également la condition de satisfaction, la réflexivité, la transitivité, la monotonie et la $\vdash$-translation. On a vu que la condition de satisfaction, la monotonie et la $\vdash$-translation ne sont pas toujours souhaitables, et on a donc décidé d'adopter ici une définition « minimaliste » : après tout, tout ce que l'on demande, c'est simplement d'avoir un mécanisme d'inférence ($\vdash$) qui reflète [partiellement à défaut de mieux] la sémantique ($\models$) c'est-à-dire qui soit correct. Le reste n'est que détails d'implémentation. En fait, la plupart des développements en logique répondent à une motivation plus ou moins implicite de modéliser le raisonnement mathématique humain ; cela explique sans doute quelques inadéquations de certaines définitions générales lorsqu'on veut les appliquer à la formalisation en informatiques.

Il semble que la réflexivité et la transitivité soit souhaitables en informatique. Voici de plus une version « forte » de la réflexivité qui est utile si l'on ne veut pas imposer la monotonie :

$$\text{si } \varphi \in \Gamma \text{ alors } \Gamma \vdash \varphi$$

2 Les systèmes formels

La notion de logique générale et les propriétés générales qu'on peut définir dessus sont seulement des guides qui permettent de s'abstraire des contingences calculatoires ou de modélisation. Comme on l'a vu, rien n'impose dans ce qui précède que les signatures soient des ensembles de symboles par exemple, ni que les formules soient des suites de symboles, ou que les inférences soient obtenues par des techniques quelconques de raisonnement effectif. Il faut cependant les incarner (de diverses façons) par de « véritables » manipulations de symboles. Ceci peut être fait d'une façon qui reste encore assez générale avec la notion de système formel.

2.1 Définitions et notations

Définition 4 : Un *système formel*, également appelé *calcul*, est un triplet $C = (A, \mathcal{F}, \mathcal{R})$ où :

- A est un ensemble, appelé *alphabet* et dont les éléments sont appelés *symboles*
- $\mathcal{F}$ est un sous-ensemble de A^*, dont les éléments sont appelés les *formules* bien formées
- $\mathcal{R}$ est un ensemble fini de prédicats d'arité ≥ 1 sur les formules ; chaque prédicat r de $\mathcal{R}$ est appelé une *règle de déduction*, ou encore une *règle d'inférence*, et porte donc sur $\mathcal{F}^n$ où n est l'arité de r

Par rapport aux sections précédentes, l'alphabet A d'un calcul correspond intuitivement à l'union de trois ensembles de symboles :

1. les connecteurs, quantificateurs ou prédicats qui constituent ce qu'on avait appelé la « partie fixe » de la logique considérée
2. les symboles apparaissant dans ce qu'on avait appelé la « partie utilisateur », c'est-à-dire la signature
3. des symboles annexes, comme les parenthèses et la virgule, qui permettent de structurer l'expression des formules.

En pratique, l'ensemble $\mathcal{F}$ n'est pas quelconque, il est défini inductivement, par une grammaire ou tout autre moyen autorisant un algorithme simple de reconnaissance des formules bien formées.

Une règle d'inférence r sert en fait à définir une étape élémentaire de preuve. Si $\varphi_1 \cdots \varphi_n$ vérifient que $r(\varphi_1, \cdots, \varphi_n)$ est vrai, cela signifie intuitivement que φ_n est déduit de φ_1 et ... et φ_{n-1}. Là aussi, en pratique, une règle d'inférence n'est pas quelconque, elle est définie par un moyen algorithmique très simple pour savoir si le prédicat est vrai ou faux.

Terminologie :

- On dit alors que $(\varphi_1, \cdots, \varphi_n)$ est une *instance* de la règle r, que $\varphi_1 \ \ldots \ \varphi_{n-1}$ sont les *prémisses* de cette instance, et que φ_n en est la conclusion.
- Pour des raisons beaucoup plus historiques que logiques, on distingue le cas particulier des règles d'arité 1 et on les appelle des *schémas d'axiomes*. Une instance d'un schéma d'axiomes est donc réduit à une formule qui en est sa conclusion ; on l'appelle alors *un axiome*. Un axiome n'est déduit de rien (puisque l'arité n de r est égale à 1) et est donc d'une certaine façon « admis ».

Pour bien distinguer la notion de règle (ou de schéma d'axiomes si l'arité est 1) de la notion d'instance (ou d'axiome si l'arité est 1), prenons un exemple. Supposons que $A = \{ X , Y , \Rightarrow \}$ et que $\mathcal{F}$ soit l'ensemble des mots qui commencent et finissent par une lettre (X ou Y), et ne contiennent jamais deux lettres consécutives. Une règle d'arité 3 de $\mathcal{R}$ peut être par exemple appelée *mp* et définie par le fait que que pour toutes lettres l_1 et l_2, $mp(l_1 , l_1 \Rightarrow l_2 , l_2)$ est vrai (et faux pour toutes autres formules). Une instance de *mp* est par exemple celle ayant pour prémisses X et $X \Rightarrow Y$, et pour conclusion Y. Une autre instance de *mp* est par exemple celle ayant pour prémisses X et $X \Rightarrow X$, et pour conclusion X. Il existe en fait 4 instances de cette règle ici [exercice : donner les deux manquantes]. Un schéma d'axiomes peut être par exemple la règle vérifiant que $r(l \Rightarrow l)$ est vrai pour toute lettre l (et faux pour toutes autres formules). Ce schéma d'axiomes recouvre donc deux axiomes qui sont $X \Rightarrow X$ et $Y \Rightarrow Y$.

Notation : étant donnée une instance de règle $r(\varphi_1, \varphi_2, \cdots, \varphi_n)$, on la note de la façon suivante :

$$\frac{\varphi_1 \quad \varphi_2 \quad \cdots \quad \varphi_{n-1}}{\varphi_n} r$$

On peut même supprimer le nom de la règle r lorsque c'est évident par le contexte.

2.2 Système formel vu comme système d'inférence

À partir d'un calcul, on peut construire des preuves en utilisant des instances de règles les unes après les autres de telle sorte que les prémisses utilisées à chaque étape soient toutes déjà prouvées dans les étapes précédentes.

Définition 5 : Etant donnés un calcul $C = (A, \mathcal{F}, \mathcal{R})$ et un ensemble de formules $\Gamma \subseteq \mathcal{F}$, une *déduction* dans C à partir des hypothèses Γ est une séquence $\psi_1 \ \cdots \ \psi_k$ de formules de $\mathcal{F}$ telle que pour tout i de 1 à k :

- ou ψ_i appartient à Γ
- (ou ψ_i est un axiome de C)
- ou encore il existe une instance $\frac{\varphi_1 \ \cdots \ \varphi_{n-1}}{\varphi_n} r$ d'une règle de $\mathcal{R}$ dont la conclusion φ_n soit égale à la formule ψ_i et chacune des prémisses φ_m $(1 \leq m \leq n-1)$ soit égale à l'une des formules de la preuve qui précèdent ψ_i (i.e. ψ_j avec $j < i$).

On remarque que la définition précédente ne contient en fait que deux cas. En effet, le deuxième cas n'est qu'un cas particulier du troisième, où l'arité de la règle r est 1 ; c'est pourquoi on l'a écrit entre parenthèses.

On peut maintenant facilement retrouver la notion d'inférence ($\vdash$) des systèmes d'inférence :

Définition 6 : Etant donnés un calcul $C = (A, \mathcal{F}, \mathcal{R})$, un ensemble de formules $\Gamma \subseteq \mathcal{F}$ et une formule $\varphi \in \mathcal{F}$, on dit que Γ infère φ s'il existe une déduction dans C à partir de Γ dont la dernière formule soit égale à φ (i.e. $\psi_1 \ \cdots \ \psi_k$ où ψ_k est égale à φ).
Dans ce cas, on note $\Gamma \vdash_C \varphi$, et on remarque que le triplet $(Sign, For, \vdash)$ où
- $Sign = \{A\}$
- For est l'application qui associe à l'unique signature A de $Sign$ l'ensemble $\mathcal{F}$, c'est-à-dire que $For(A) = \mathcal{F}$
- $\vdash_A = \vdash_C$

forme un système d'inférence (cf. définition 2).

Ainsi, un système formel C définit un système d'inférence dans lequel il n'y a aucun choix possible de signature. C'est une conséquence du fait qu'un alphabet A ne fait aucune distinction entre les trois ensembles de symboles mentionnés juste après la définition 4. Un système formel se focalise sur *une* démarche systématique de preuve, sans posséder de sémantique (Mod manque) et en ayant une notion de syntaxe réduite à sa plus simple expression (des mots sur un alphabet).

Il est alors naturel d'inventorier les propriétés que possède le système d'inférence obtenu, parmi celles définies en section 1.3 :

- Réflexivité : oui.
  Preuve : on veut prouver que $\{\varphi\} \vdash_C \varphi$, c'est-à-dire qu'il existe une séquence de formules qui forme une déduction pour C à partir de $\Gamma = \{\varphi\}$ et qui se termine par φ. Ici il suffit d'une preuve de longueur 1, puisque φ appartenant à Γ, il peut être posé directement.
- Transitivité : oui.
  Preuve : On veut prouver que si $\Gamma \vdash_C \Gamma'$ et si $\Gamma \cup \Gamma' \vdash \varphi$, alors $\Gamma \vdash \varphi$. De $\Gamma \cup \Gamma' \vdash \varphi$ on déduit qu'il existe une déduction $\psi_1 \ \cdots \ \psi_{k-1}\varphi$ dans C à partir de $\Gamma \cup \Gamma'$ qui termine sur φ. On veut construire une déduction à partir de Γ seul. Pour chacune des formules ψ_i, elle peut être obtenue par une règle de $\mathcal{R}$ (schémas d'axiomes compris), ou parce qu'elle appartient à $\Gamma \cup \Gamma'$. Le seul cas gênant est naturellement lorsque $\psi_i \in \Gamma'$ (et $\psi_i \notin \Gamma$) puisqu'alors elle n'est plus légitimement à sa place dans une déduction à partir de Γ seul. Cependant, puisque $\Gamma \vdash_C \Gamma'$, pour chacune des formule ψ_i qui se trouvent dans ce cas il existe une preuve $\eta_1 \cdots \eta_{m-1} \ \psi_i$ de ψ_i. Il suffit donc de remplacer ces ψ_i dans la séquence $\psi_1 \ \cdots \ \psi_{k-1}\varphi$ par leur preuve entière $\eta_1 \cdots \eta_{m-1} \ \psi_i$. On obtient alors une preuve φ à partir de Γ.
- Monotonie : oui. (Preuve laissée en exercice)
- $\vdash$-translation : sans objet puisqu'il n'y a qu'une seule signature.

Bref, un système formel est une instance (avec manipulations explicites de symboles) de système d'inférence ayant quasiment toutes les « bonnes »propriétés qui en rendent l'utilisation facile. On a vu que la monotonie pouvait être restrictive lorsqu'on aborde des logiques liées à des notions de traitement d'exception. Ainsi est-on amené parfois à proposer des systèmes d'inférence qui ne sont pas des systèmes formels au sens défini ici. Toutefois, en pratique, tous les systèmes de preuve manipulés peuvent être définis par des systèmes formels[4].

2.3 Propriétés classiques des systèmes formels

Toutes les propriétés mentionnées sur les systèmes d'inférence étant satisfaites par les systèmes formels, elles ne peuvent pas être utilisées comme critère pour choisir un système formel plutôt qu'un autre. D'autres propriétés plus fines doivent être définies, qui puissent être vérifiées par certains systèmes formels et pas par d'autres. Pour ce faire, une notion préalable centrale est celle de théorèmes d'un calcul.

Définition 7 : Etant donné un calcul $C = (A, \mathcal{F}, \mathcal{R})$, l'*ensemble des théorèmes* de C est le sous-ensemble $Th(C)$ de $\mathcal{F}$ défini inductivement par :

[4]Souvent en ajoutant le symbole $\vdash$ lui-même dans l'alphabet A considéré, comme on le verra plus loin

– (tout axiome de C appartient à $Th(C)$)
– si toutes les prémisses d'une instance d'une règle d'inférence r de $\mathcal{R}$ appartiennent à $Th(C)$, alors sa conclusion appartient aussi à $Th(C)$.

Ici également, le premier cas n'est qu'un cas particulier du second ; c'est pourquoi nous l'avons mis entre parenthèses.

Proposition 8 : $\forall\, \varphi \in \mathcal{F}\,, \quad \varphi \in Th(C) \Longleftrightarrow \emptyset \vdash_C \varphi$

La preuve de cette proposition est donnée en devoir à rendre. On pourra utiliser des techniques décrites dans le cours de mise à niveau en calcul symbolique.

Cette proposition signifie en fait que la notion d'inférence de la définition 5 n'est qu'une « mise à plat » des preuves de théorèmes. En effet, un théorème selon la définition 7 est le résultat d'une construction récursive que l'on appelle un *arbre de preuve* :

– aux feuilles de cet arbre on trouve les axiomes (qui ne sont déduits de rien)
– un nœud interne de l'arbre est une conclusion d'une instance d'une règle d'arité > 1, et ses fils sont les prémisses de l'instance
– les branches de l'arbre sont donc des liens signifiant, du fils vers le père, « est une prémisse de » (étiquetées par le nom de la règle le cas échéant).

Cela conduit à une représentation graphique des preuves :

$$\dfrac{\dfrac{\psi_1 \;\cdots\; \psi_u}{\varphi_1} r_1 \quad \dfrac{\dfrac{\nu_1 \cdots \nu_i}{\gamma_1} r_2 \;\cdots\; \gamma_v}{\varphi_2} r_3 \quad \cdots \quad \dfrac{\dfrac{\eta_1 \cdots \eta_j}{\beta_1} r_4 \;\cdots\; \dfrac{\alpha_1 \cdots \alpha_k}{\beta_w} r_5}{\varphi_{n-1}} r_6}{\varphi_n} r_7$$

et cet arbre de preuve donne lieu à autant d'inférences permettant de prouver $\emptyset \vdash_C \varphi_n$ qu'il y a de parcours *en profondeur d'abord* de l'arbre.

La notion de théorème étant définie rigoureusement, on peut maintenant définir quelques propriétés parmi les plus courantes sur les divers systèmes formels. Un système formel $C = (A, \mathcal{F}, \mathcal{R})$ sera dit :

– **cohérent** s'il existe des formules de $\mathcal{F}$ qui ne sont pas des théorèmes, i.e. $Th(C) \neq \mathcal{F}$
– **décidable** s'il existe un algorithme permettant de décider, pour toute formule $\varphi \in \mathcal{F}$, si c'est un théorème ou non
– **saturé** si pour toute formule φ de $\mathcal{F}$ qui n'est pas un théorème de C, le calcul C' obtenu à partir de C en ajoutant φ comme axiome n'est plus cohérent
– avec **axiomes indépendants** si pour tout axiome α de C, le calcul C' obtenu à partir de C en supprimant cet axiome de $\mathcal{R}$ a un ensemble de théorèmes strictement plus petit (i.e. $Th(C') \neq Th(C)$).

On doit remarquer que *les systèmes formels n'ont aucune sémantique*, ils ont seulement pour objet de proposer un mécanisme de preuve par transformations de symboles. De ce fait, les notions de correction et de complétude n'ont aucun sens pour un système formel seul. Cependant, on peut « biaiser » de la façon suivante :

– En pratique, lorsque l'on propose un calcul C, les règles d'inférence (de $\mathcal{R}$) que l'on propose sont issues d'une intuition provenant d'un phénomène réel sur lequel on veut raisonner de manière mécaniquement vérifiable.
– On a donc un ensemble de modèles (ou un seul modèle) en tête et l'on a également une idée précise de ce que signifie la satisfaction d'une formule de $\mathcal{F}$ par (ce ou) ces modèles.
– Si on a la chance de pouvoir caractériser exhaustivement l'ensemble de toutes les formules qui doivent être vérifiées d'après cette « sémantique implicite », alors on dispose d'une définition rigoureuse d'un ensemble $\mathcal{S}em \subset \mathcal{F}$ de formules qui doivent être vraies d'après la sémantique.
– On peut alors légitimement se demander si le calcul C permet d'inférer effectivement les formules de $\mathcal{S}em$ ou non, et cela donne lieu à des notions un peu particulières de correction et de complétude, qu'on pourrait presque qualifier de « syntaxique ».

Définition 9 : Un système formel $C = (A, \mathcal{F}, \mathcal{R})$ est dit *correct* par rapport à un ensemble de formules $\mathcal{S}em \subset \mathcal{F}$ si et seulement si $Th(C) \subseteq \mathcal{S}em$. C'est-à-dire qu'on ne peut pas inférer une formule qui n'est pas dans $\mathcal{S}em$.

Définition 10 : Un système formel $C = (A, \mathcal{F}, \mathcal{R})$ est dit *complet* par rapport à un ensemble de formules $\mathcal{S}em \subset \mathcal{F}$ si et seulement si $\mathcal{S}em \subseteq Th(C)$. C'est-à-dire qu'on peut inférer toutes les formules de $\mathcal{S}em$.

En pratique, on ne considère la complétude que lorsque la correction est préalablement prouvée. On voit mal, en effet, l'intérêt d'un calcul qui ne serait pas correct. [Fin du deuxième cours ici].

3 Un exemple simple

Supposons que l'on veuille développer une logique simple pour capturer des équations algébriques (identités remarquables, etc.) ; nous allons voir comment en faire une logique générale et des systèmes formels. L'idée est qu'une formule est une équation entre deux termes, par exemple $(a+b)^2 = (a^2 + b^2) + 2 \times a \times b$. Cela conduit à la logique dite *équationnelle*.

3.1 En tant qu'institution

On doit définir l'ensemble des signatures ($Sign$), l'application qui définit l'ensemble des formules au-dessus de chaque signature (For), celle qui définit l'ensemble des modèles au-dessus de chaque signature (Mod) et le prédicat de satisfaction ($\models$).

3.1.1 Signatures

Une signature dans ce cadre, c'est l'ensemble des opérations qu'on peut utiliser pour construire les deux termes de part et d'autre du signe $=$. Pour l'identité remarquable mentionnée plus haut, il faudrait par exemple choisir une signature contenant au moins l'addition, la multiplication et le carré. Naturellement, pour savoir quelles sont les formules bien formées il faut savoir quels sont les termes bien formés et pour cela il faut donner l'arité de chaque opération d'une signature. Il faut également connaître l'ensemble des variables autorisées (a, b, etc).

On peut définir $Sign$ comme l'ensemble des signatures définies ci-dessous.
- Une signature $\Sigma = (V, F)$ est définie par un ensemble de variables V et un ensemble de symboles d'opération F où chaque symbole est muni d'une arité appartenant à $\mathbb{N}$.
- Un morphisme entre deux signatures Σ_1 et Σ_2 est défini par une application injective entre variables $\sigma_V : V_1 \to V_2$ et une application entre opérations $\sigma_F : F_1 \to F_2$ qui respecte les arités.

Comme exemples de signatures, on peut imaginer :
- celle des listes, Σ_1, où l'ensemble des variables V_1 est l'ensemble des lettres majuscules et l'ensemble d'opérations F_1 contient *nil* d'arité 0, *cons* d'arité 2 et *concat* d'arité 2
- celle des entiers, Σ_2, où l'ensemble des variables V_2 est l'ensemble des lettres minuscules et l'ensemble d'opérations F_2 contient 0 d'arité 0, + d'arité 2, $\times$ d'arité 2 et *carré* d'arité 1
- un morphisme possible $\sigma : \Sigma_1 \to \Sigma_2$ peut être défini par σ_V qui associe à chaque lettre majuscule la même en minuscule, et $\sigma_F(nil) = 0$, $\sigma_F(cons) = +$, $\sigma_F(concat) = \times$; il y a naturellement d'autres morphismes possibles.

3.1.2 Formules

Pour chaque signature $\Sigma = (V, F)$ l'ensemble des formules bien formées est tout naturellement l'ensemble $For(\Sigma)$ des équations de la forme $t = t'$ où t et t' sont des termes construits avec les opérations de F au-dessus des variables de V, comme définis dans le cours de mise à niveau en calcul symbolique (i.e. t et t' appartiennent à $T_F(V)$).

On rappelle qu'il ne suffit pas de dire quel ensemble de formules l'application For associe à chaque signature, il faut aussi définir comment For transporte les morphismes. Etant donné un morphisme de signatures $\sigma : \Sigma_1 \to \Sigma_2$, il nous faut définir $\overline{\sigma} : For(\Sigma_1) \to For(\Sigma_2)$. C'est assez simple : étant donnée une formule $\varphi_1 \in For(\Sigma_1)$, $\overline{\sigma}(\varphi_1)$ est la formule φ_2 obtenue à partir de φ_1 en remplaçant chaque occurrence d'une variable $x \in V_1$ par la variable $\sigma_V(x) \in V_2$ et chaque occurrence d'une opération $f \in F_1$ par l'opération $\sigma_F(f) \in F_2$.

Ainsi pour l'exemple de morphisme de signatures donné précédemment, la Σ_1-formule

$$concat(cons(A, B), C) = cons(A, concat(B, C))$$

devient par $\overline{\sigma}$ la Σ_2-formule $(a + b) \times c = a + (b \times c)$.

3.1.3 Modèles

Dans le cas qui nous préoccupe, on veut modéliser des opérations qui effectuent des calculs sur des données. Un modèle au-dessus d'une signature fixée sera donc simplement un ensemble M d'éléments (i.e. de données) sur lequel travaillent les opérations de la signature (en accord avec leur arité). Cela signifie que pour chaque symbole $f \in F$ d'arité n, on doit disposer d'une application $f_M : M^n \to M$ (qu'on appelle la sémantique de f dans le modèle M).

Ainsi on peut définir Mod comme l'application qui associe à chaque signature Σ l'ensemble $Mod(\Sigma)$ de tous les modèles M possibles munis de toutes les familles possibles de f_M pour $f \in F$.

Un exemple de modèle sur la signature Σ_2 précédente peut être l'ensemble $\mathbb{R}$ des nombres réels, les opérations de F_2 ayant la sémantique habituelle.

Il faut également définir comment Mod transporte un morphisme de signatures $\sigma : \Sigma_1 \to \Sigma_2$, c'est-à-dire définir $U\_\sigma : Mod(\Sigma_2) \to Mod(\Sigma_1)$. C'est relativement simple : étant donné un modèle M_2, $U(M_2)$ est le modèle M_1 tel que l'ensemble sous-jacent est le même que M_2, et pour chaque opération $f \in F_1$, l'application f_{M_1} est égale à l'application $\sigma(f)_{M_2}$.
Si par exemple M_2 est le modèle $\mathbb{R}$ précédent, $U\_\sigma(\mathbb{R})$ est tout simplement encore l'ensemble des nombres réels $\mathbb{R}$, mais 0 s'écrit *nil*, l'addition s'écrit *cons*, la multiplication s'écrit *concat* (renommages) et l'opération *carré* a été oubliée. Attention, c'est seulement un « coup de peinture » syntaxique, l'oubli ne transforme en aucun cas les réels en listes !)

3.1.4 Satisfaction

Le prédicat de satisfaction est défini via les notions de *substitution* des variables par des valeurs d'un modèle M.

Définition 11 : Étant donnés une signature $\Sigma = (V, F)$ et un modèle $M \in Mod(\Sigma)$, une *interpré-tation* est une application ν de V dans M (qui associe donc une valeur dans M à chaque variable).

On prolonge toute interprétation ν en une *substitution* $\nu^\natural$ de $T_F(V)$ dans M de la manière inductive suivante :

– pour tout terme t réduit à une variable x, on pose $\nu^\natural(t) = \nu(x)$
– pour toute opération $f \in F$ d'arité n et tous termes $t_1 \ \cdots \ t_n$, on pose

$$\nu^\natural(f(t_1, \cdots, t_n) = f_M(\nu^\natural(t_1), \cdots, \nu^\natural(t_n))$$

Par abus de notation, $\nu^\natural$ est encore noté ν.

On prolonge de même ν en un prédicat sur $For(\Sigma)$ qui est vrai pour une formule $(t = t')$ si et seulement si $\nu(t)$ (qui est donc un élément de M) est égal à $\nu(t')$. On note alors ce prédicat de la manière suivante : $M \models^\nu_\Sigma (t = t')$, qui se lit « M satisfait $(t = t')$ pour la substitution ν ».

Définition 12 : Etant donnés une signature $\Sigma = (V, F)$, un modèle $M \in Mod(\Sigma)$ et une formule $\varphi \in For(\Sigma)$, M satisfait φ si et seulement si pour toute substitution $\nu : V \to M$, $M \models^\nu_\Sigma (t = t')$. Dans ce cas on écrit $M \models \_\Sigma(t = t')$, et cela définit bien la famille de prédicats $\models$ des institutions.

3.2 En tant que système d'inférence

Sign et *For* ayant déjà été définis, il reste à définir $\vdash$. Une seconde notion de substitution (sur les formules) est utile :

Notation 13 : Etant donnés une signature $\Sigma = (V, F)$ et une substitution $\tau : V \to T_F(V)$, on étend τ en une substitution $\tau^\flat : For(\Sigma) \to For(\Sigma)$ telle que, pour toute formule $t = t'$ de $For(\Sigma)$, $\tau^\flat(t = t')$ est la formule $\tau^\natural(t) = \tau^\natural(t')$)

Par exemple, si φ est la formule $(a+b)^2 = (a^2+b^2)+2\times a\times b$ et τ associe le terme $b+c$ à la variable a et le terme b^2 à la variable b, alors $\tau^\flat(\varphi)$ est la formule $((b+c)+b^2)^2 = ((b+c)^2+(b^2)^2)+2\times(b+c)\times b^2$

Définition 14 : Etant donnés une signature $\Sigma = (V, F)$ et un ensemble de formules $\Gamma \subset For(\Sigma)$, on note $Th\_\Sigma(\Gamma)$ le sous-ensemble de $For(\Sigma)$ définit inductivement par :
– $\Gamma \subseteq Th\_\Sigma(\Gamma)$
– pour tout terme $t \in T_F(V)$, $(t = t) \in Th\_\Sigma(\Gamma)$
– si $(t = t') \in Th\_\Sigma(\Gamma)$ alors $(t' = t) \in Th\_\Sigma(\Gamma)$
– si $(t = t') \in Th\_\Sigma(\Gamma)$ et $(t' = t <'' >) \in Th\_\Sigma(\Gamma)$ alors $(t = t <'' >) \in Th\_\Sigma(\Gamma)$
– si φ appartient à $Th\_\Sigma(\Gamma)$ alors pour toute substitution $\tau : V \to T_F(V)$, $\tau^\flat(\varphi)$ appartient encore à $Th\_\Sigma(\Gamma)$
– pour toute opération $f \in F$ d'arité n, si $t_1 = t'_1, \ \ldots \ , t_n = t'_n$ sont n formules appartenant à $Th\_\Sigma(\Gamma)$, alors $f(t_1, \cdots, t_n) = f(t'_1, \cdots, t'_n)$ appartient encore à $Th\_\Sigma(\Gamma)$.

On note $\Gamma \vdash \_\Sigma\varphi$ pour $\varphi \in Th\_\Sigma(\Gamma)$.

La définition précédente dit simplement que l'on peut mener les inférences de la logique équationnelle en utilisant le fait que l'égalité est réflexive, symétrique, transitive, compatible avec les substitutions et compatible avec les opérations de la signature.

La logique générale $(Sign, For, Mod, \models, \vdash)$ que l'on vient de définir est la *logique équationnelle*. Elle possède de plus toutes les propriétés des logiques générales définies en section 1.3. Seule la preuve de complétude présente de réelles difficultés, et est faite en cours d'option sur les spécifications par propriétés ; la preuve des autres propriétés est laissée en exercice (attention : la preuve de correction doit être faite en se référant *très soigneusement* aux définitions de $\models$ et $\vdash$ et est de ce fait très fastidieuse[5]).

[5]Si elle vous semble courte, c'est probablement que quelque chose vous a échapé.

3.3 En tant que systèmes formels

La logique équationnelle que nous venons de définir n'impose qu'un seul symbole : l'égalité. Tous les autres symboles servent à construire les termes et dépendent de la signature choisie. Par conséquent, il y a autant d'alphabets et de systèmes formels issus de la logique équationnelle qu'il y a de signatures possibles. On va donc définir non pas « le » système formel de la logique équationnelle, mais plutôt associer un système formel $C = (A, \mathcal{F}, \mathcal{R})$ à chaque signature $\Sigma = (V, F)$ de $Sign$.

- Pour définir l'alphabet A, il faut non seulement les symboles de variables et d'opérations mais, sachant que par définition les formules d'un système formel sont « à plat » ($\mathcal{F} \subset A^*$, il faut également ajouter les symboles de parenthèses et la virgule pour écrire les termes. Ainsi, $A = V \cup F \cup \{ \, (\, , \, , \,) \, \}$.
- Naturellement, $\mathcal{F} = For(\Sigma)$.
- Enfin $\mathcal{R}$ contient 5 règles :
  1. le schéma d'axiomes appelé **réflexivité** de l'égalité : $\frac{}{t = t}$, qui donne lieu à un axiomes pour chaque terme $t \in T_F(V)$
  2. la règle appelée **symétrie** de l'égalité : $\frac{t = t'}{t' = t}$, qui possède autant d'instances qu'il y a de termes avec variables t et t'
  3. la règle appelée **transitivité** de l'égalité : $\frac{t = t' \quad t' = t <''>}{t = t <''>}$, qui possède autant d'instances qu'il y a de termes avec variables t, t' et $t <''>$
  4. la règle dite **de substitution** : $\frac{\varphi}{\tau^\flat(\varphi)}$, qui possède autant d'instances qu'il y a d'équations φ et de substitution $\tau : V \to T_F(V)$
  5. la règle de **compatibilité** de l'égalité avec la signature : $\frac{t_1 = t'_1 \quad \cdots \quad t_n = t'_n}{f(t_1, \cdots, t_n) = f(t'_1, \cdots, t'_n)}$, qui possède autant d'instances qu'il y a d'opérations dans F et de termes $t_1 \cdots t_n, t'_1 \cdots t'_n$ où n est l'arité de f.

  Attention : ne pas confondre les règles d'inférences appelées réflexivité et transitivité *de l'égalite* avec les propriétés du même nom qui portent sur les systèmes d'inférence. Ca n'a rien à voir.

Nous venons de définir ici ce qu'on appelle le calcul de Birkhoff, du nom du logicien qui en a prouvé la complétude. Bien sûr, l'inférence associée au calcul de Birkhoff pour chaque signature est celle du système d'inférence équationnel défini en section 3.2.

Devoir à rendre : Supposons par exemple que Γ soit l'ensemble de formules suivant sur la signature des entiers relatifs :

$\varphi_1 : \; x + y = y + x$
$\varphi_2 : \; x + (y + z) = (x + y) + z$
$\varphi_3 : \; x \times y = y \times x$
$\varphi_4 : \; x \times (y \times z) = (x \times y) \times z$
$\varphi_5 : \; x \times (y + z) = x \times y + x \times z$
$\varphi_6 : \; x + 0 = x$
$\varphi_7 : \; x + oppose(x) = 0$
$\varphi_8 : \; x - y = x + oppose(y)$
$\varphi_9 : \; x^2 = x \times x$

et que l'on veuille prouver la formule $(a + b) \times (a - b) = a^2 - b^2$.

Complétez l'arbre de preuve ci-dessous (sans dupliquer les sous-arbres correspondant à des lemmes utilisés plusieurs fois dans la preuve).

$$\dfrac{\dfrac{\mathcal{A}_1 \qquad \mathcal{A}_2}{(a+b)\times(a-b) = (a^2 - a\times b) + (b\times a - b^2)} r3 \quad \dfrac{\ldots}{(a^2 - a\times b) + (b\times a - b^2) = a^2 - b^2} r3}{(a+b)\times(a-b) = a^2 - b^2}$$

$\mathcal{A}_2$ étant le sous-arbre de preuve ci-dessous que vous devez également compléter :

$$\frac{\dots}{(a \times (a-b)) + (b \times (a-b)) = (a^2 - a \times b) + (b \times a - b^2)}$$

et $\mathcal{A}_1$ étant le sous-arbre de preuve (complet et à suivre en exemple) ci-dessous :

$$\dfrac{\dfrac{\dfrac{\dfrac{}{x \times y = y \times x}\Gamma}{(a+b) \times c = c \times (a+b)} r4[x/(a+b), y/c] \qquad \dfrac{\mathcal{A}_3 \qquad \mathcal{A}_4}{c \times (a+b) = a \times c + b \times c} r3}{(a+b) \times c = a \times c + b \times c} r3}{(a+b) \times (a-b) = (a \times (a-b)) + (b \times (a-b))} r4[c/(a-b)]$$

où $\mathcal{A}_3$ est le sous-arbre :

$$\frac{\dfrac{}{x \times (y+z) = x \times y + x \times z}\Gamma}{c \times (a+b) = c \times a + c \times b} r4[x/c, y/a, z/b]$$

et $\mathcal{A}_4$ est le sous-arbre :

$$\frac{\dfrac{\dfrac{}{x \times y = y \times x}\Gamma}{c \times a = a \times c} r4[x/c, y/a] \qquad \dfrac{\dfrac{}{x \times y = y \times x}\Gamma}{c \times b = b \times c} r4[x/c, y/b]}{c \times a + c \times b = a \times c + b \times c} r5+$$

Comme on l'a constaté sur l'exemple de la logique équationnelle, il est clairement équivalent de définir inductivement l'inférence du système formel (comme en définition 14 avec les 5 cas) ou de donner les règles du calcul correspondant (les 5 règles du calcul de Birkhoff). De manière similaire dans tous les chapitres suivants, on définira le prédicat d'inférence des logiques exposées ($\vdash$) en donnant directement des règles d'inférence définissant un calcul adéquat[6]. [Fin du troisième cours]

[6]En somme, on ne fera plus systématiquement la distinction entre *système formel* et *système d'inférence*.

Chapitre 3 : La logique propositionnelle

L'exemple de la logique equationnelle est relativement simple parce qu'il ne propose aucun connecteur logique, aucun quantificateur et un seul prédicat. Nous allons maintenant nous attacher à définir deux logiques élémentaires très classiques qui introduisent successivement ces pouvoirs d'expression : la *logique propositionnelle* (également appelée *logique des propositions*) offre les connecteurs et la *logique des prédicats* (chapitre suivant) étend le pouvoir d'expression aux prédicats et quantificateurs.

1 Syntaxe, sémantique et calcul de Hilbert

L'idée est que la « partie utilisateur » est une collection d'affirmations (propositions) élémentaires, à partir desquelles on construit les autres formules au moyen des connecteurs habituels (implication, négation, conjonction, disjonction).

Définition 1 : La logique propositionnelle est définie par le quintuplet $(Sign, For, Mod, \models, \vdash)$ suivant :

- Un élément de $Sign$ est un *ensemble* P dont les éléments sont appelés des *variables propositionnelles*. Etant donnés deux ensembles de variables propositionnelles P et P', un morphisme entre ces deux signatures est simplement une application $\sigma : P \to P'$.
- Etant donné un ensemble de variables propositionnelles P, $For(P)$ est défini inductivement par :
  - $P \subset For(P)$ (toute variable propositionnelle est une formule, qu'on appelle alors un *atome*)
  - si $\varphi \in For(P)$ alors $\neg\varphi \in For(P)$
  - si $\varphi_1 \in For(P)$ et $\varphi_2 \in For(P)$ alors $(\varphi_1 \to \varphi_2) \in For(P)$ et $(\varphi_1 \wedge \varphi_2) \in For(P)$ et $(\varphi_1 \vee \varphi_2) \in For(P)$.

  De plus, étant donné un morphisme de signatures $\sigma : P \to P'$, on définit $\overline{\sigma} : For(P) \to For(P')$ de manière évidente :
  - pour toute variable propositionnelle $p \in P$, $\overline{\sigma}(p) = \sigma(p)$
  - $\overline{\sigma}(\neg\varphi) = \neg\,\overline{\sigma}(\varphi)$
  - $\overline{\sigma}(\varphi_1 \to \varphi_2) = \overline{\sigma}(\varphi_1) \to \overline{\sigma}(\varphi_2)$
  - $\overline{\sigma}(\varphi_1 \wedge \varphi_2) = \overline{\sigma}(\varphi_1) \wedge \overline{\sigma}(\varphi_2)$
  - $\overline{\sigma}(\varphi_1 \vee \varphi_2) = \overline{\sigma}(\varphi_1) \vee \overline{\sigma}(\varphi_2)$
- Etant donné un ensemble de variables propositionnelles P, $Mod(P)$ est l'ensemble des *valuations* de P, c'est-à-dire l'ensemble des applications v de P dans $\{0,1\}$. Une proposition $p \in P$ est dite *vraie pour* v si et seulement si $v(p) = 1$.

  De plus, étant donné un morphisme de signatures $\sigma : P \to P'$, on définit $U\_\sigma : Mod(P') \to Mod(P)$ par $U\_\sigma(v') = v$ avec $v(p) = v'(\sigma(p))$.
- Etant données une formule φ et une valuation $v : P \to \{0,1\}$, on définit inductivement $v \models \varphi$ par :
  - si φ est réduit à une variable propositionnelle, $v \models \varphi$ si et seulement si $v(\varphi) = 1$;
  - si φ est de la forme $\neg\psi$, $v \models \varphi$ si et seulement si $v(\psi) = 0$;
  - si φ est de la forme $(\varphi_1 \to \varphi_2)$ (resp. $(\varphi_1 \wedge \varphi_2)$, resp. $(\varphi_1 \vee \varphi_2)$), alors $v \models \varphi$ si et seulement si $v(\varphi_1) \leq v(\varphi_2)$ (resp. $v(\varphi_1) \times v(\varphi_2) = 1$, resp. $v(\varphi_1) + v(\varphi_2) \geq 1$) ;
- Enfin l'inférence $\vdash$ est définie comme l'inférence résultant de l'ensemble des règles du *calcul propositionnel de Hilbert* suivant :

1. $\dfrac{}{\varphi \to (\psi \to \varphi)}$ [implication-a]
2. $\dfrac{}{(\varphi \to (\psi \to \eta)) \to ((\varphi \to \psi) \to (\varphi \to \eta))}$ [implication-b]
3. $\dfrac{}{(\varphi \wedge \psi) \to \varphi}$ [conjonction-a]
4. $\dfrac{}{(\varphi \wedge \psi) \to \psi}$ [conjonction-b]
5. $\dfrac{}{\varphi \to (\psi \to (\varphi \wedge \psi))}$ [conjonction-c]
6. $\dfrac{}{\varphi \to (\varphi \vee \psi)}$ [disjonction-a]
7. $\dfrac{}{\psi \to (\varphi \vee \psi)}$ [disjonction-b]
8. $\dfrac{}{(\varphi \to \eta) \to ((\psi \to \eta) \to ((\varphi \vee \psi) \to \eta))}$ [raisonnement par cas]
9. $\dfrac{}{(\varphi \to \psi) \to ((\varphi \to \neg\psi) \to \neg\varphi)}$ [raisonnement par l'absurde]
10. $\dfrac{}{\neg\neg\varphi \to \varphi}$ [tiers exclus]
11. $\dfrac{\varphi \quad (\varphi \to \psi)}{\psi}$ [modus ponens]

(On prouvera un peu plus loin la correction du calcul par rapport à la sémantique).

Remarque 2 : Il existe une version simplifiée de la logique propositionnelle dans laquelle les connecteurs $\wedge$ et $\vee$ sont supprimés. On peut alors simuler $\varphi \vee \psi$ (resp. $\varphi \wedge \psi$) par $\neg\varphi \to \psi$ (resp. $\neg(\varphi \to \neg\psi)$). Cette version n'utilise plus que 4 règles :

1. $\dfrac{}{\varphi \to (\psi \to \varphi)}$ [implication-a]
2. $\dfrac{}{(\varphi \to (\psi \to \eta)) \to ((\varphi \to \psi) \to (\varphi \to \eta))}$ [implication-b]
3. $\dfrac{}{(\neg\psi \to \neg\varphi) \to (\varphi \to \psi)}$ [contraposée]
4. $\dfrac{\varphi \quad (\varphi \to \psi)}{\psi}$ [modus ponens]

Pour donner un exemple de manipulation de ce genre de calcul, on peut montrer formellement qu'il est possible de prouver $\varphi \to \varphi$ pour chaque formule $\varphi \in For(P)$

$$\dfrac{\dfrac{}{\varphi \to (\varphi \to \varphi)}[\text{impl-a}] \qquad \mathcal{A}}{\varphi \to \varphi}[\text{mod.pon.}]$$

où $\mathcal{A}$ est le sous-arbre :

$$\dfrac{\dfrac{}{\varphi \to ((\varphi \to \varphi) \to \varphi)}[\text{impl-a}] \quad \dfrac{}{\varphi \to ((\varphi \to \varphi) \to \varphi) \to ((\varphi \to (\varphi \to \varphi)) \to (\varphi \to \varphi))}[\text{impl-b}]}{(\varphi \to (\varphi \to \varphi)) \to (\varphi \to \varphi)}[\text{mod.pon.}]$$

Remarquons au passage la difficulté de la preuve pour une propriété aussi simple !

2 Théorème de la déduction

L'une des raisons principales de la difficulté des preuves est qu'aucune règle d'inférence ne traduit la pratique mathématique courante qui consiste à prouver que $(\varphi \to \psi)$ en posant φ comme hypothèse et en en déduisant ψ. Pourtant, de fait, cette démarche est fondée :

Théorème 3 : (Théorème de déduction) Avec le calcul de Hilbert (simplifié ou non) on a, pour tout ensemble de formules Γ et toutes formules φ et ψ sur une même signature :

$$\Gamma \vdash (\varphi \to \psi) \qquad \text{si et seulement si} \qquad \Gamma \cup \{\varphi\} \vdash \psi$$

(Cette propriété permet de simplifier considérablement les preuves.)

Preuve : *De gauche à droite :* On suppose que $\Gamma \vdash (\varphi \to \psi)$ et on veut prouver que $\Gamma \cup \{\varphi\} \vdash \psi$. Par monotonie, on a : $\Gamma \cup \{\varphi\} \vdash (\varphi \to \psi)$. Par introduction d'hypothèse on a : $\Gamma \cup \{\varphi\} \vdash \varphi$. Il suffit alors d'appliquer la règle de modus ponens pour en déduire $\Gamma \cup \{\varphi\} \vdash \psi$.

De droite à gauche : On suppose que $\Gamma \cup \{\varphi\} \vdash \psi$ et on veut prouver que $\Gamma \vdash (\varphi \to \psi)$. Cela revient à prouver que pour tout $\psi \in Th(\Gamma \cup \{\varphi\})$ on a $\Gamma \vdash (\varphi \to \psi)$; on peut donc raisonner par induction sur la définition de l'ensemble $Th(\Gamma') = Th(\Gamma \cup \{\varphi\})$.

- Si $\psi \in \Gamma'$, alors
  - soit $\psi = \varphi$ et on a justement prouvé précédemment que $\Gamma \vdash (\varphi \to \varphi)$.
  - soit $\psi \in \Gamma$, donc $\Gamma \vdash \psi$ et en appliquant [implication-a]$\Gamma \vdash (\psi \to (\varphi \to \psi))$ puis par modus ponens $\Gamma \vdash (\varphi \to \psi)$
- Sinon, ψ est obtenu par application d'une règle du calcul de Hilbert (simplifié ou non), et c'est donc soit un axiome ($\frac{}{\psi}$), soit le modus ponens ($\frac{\psi' \quad \psi' \to \psi}{\psi}$).
  - Si c'est un axiome alors $\Gamma \vdash \psi$ et on raisonne comme dans le cas où $\psi \in \Gamma$ précédent.
  - Si c'est le modus ponens, on prend donc pour hypothèses d'induction $\Gamma \vdash (\varphi \to \psi')$ d'une part et $\Gamma \vdash (\varphi \to (\psi' \to \psi))$ d'autre part, et l'on doit prouver $\Gamma \vdash (\varphi \to \psi)$. L'axiome [implication-a] donne $\Gamma \vdash (\varphi \to (\psi' \to \psi)) \to ((\varphi \to \psi') \to (\varphi \to \psi))$ et en appliquant deux fois le modus ponens on obtient $\Gamma \vdash (\varphi \to \psi)$

Ce qui termine la preuve du théorème de déduction. □

3 Correction du calcul propositionnel

Avant de poursuivre, un peu de terminologie est utile :

Terminologie : Une formule φ est une *tautologie* si et seulement si :

$$\forall v : P \to \{0,1\} \ , \quad v \models \varphi$$

Une valuation v est *un modèle* d'un ensemble de formules Γ (noté $v \models \Gamma$) si et seulement si :

$$\forall \varphi \in \Gamma \ , \quad v \models \varphi$$

Γ est *satisfiable* si et seulement si il possède au moins un modèle, i.e. :

$$\exists v : P \to \{0,1\} \ , \quad v \models \Gamma$$

Enfin Γ est *consistant* si et seulement si :

$$\exists \psi \ , \quad \Gamma \not\vdash \psi$$

Pour que le quintuplet $(Sign, For, Mod, \models, \vdash)$ de la logique propositionnelle forme réellement une logique générale, il nous reste à prouver la correction de l'inférence par rapport à la relation de satisfaction. Pour cela, on utilise deux lemmes :

Lemme 4 : Si $\Gamma \models \varphi$ et $\Gamma \models (\varphi \to \psi)$, alors $\Gamma \models \psi$.

Preuve : On doit prouver que si pour toute valuation v on a $v(\varphi) = v(\varphi \to \psi) = 1$, alors pour toute valuation v on a $v(\psi) = 1$. La preuve de ce premier lemme résulte directement de la table de vérité de l'implication $(\varphi \to \psi)$

$\varphi \backslash \psi$	0	1
0	1	1
1	0	1

où $v(\varphi) = 1$ indique que pour toute valuation v on se situe nécessairement sur la ligne du bas, et $v(\varphi \to \psi) = 1$ sur la case en bas à droite. Or cette case est nécessairement dans la colonne où $v(\psi) = 1$, donc pour toute valuation v on a bien $v(\psi) = 1$. □

Lemme 5 : Tous les axiomes du calcul des propositions (simplifié ou non) sont des tautologies.

(Preuve avec les tables de vérité laissée en exercice, facile mais fastidieuse).

Théorème 6 : Le calcul des propositions (simplifié ou non) est correct par rapport à la sémantique des propositions, c'est-à-dire : si $\Gamma \vdash \varphi$ (i.e. $\varphi \in Th(\Gamma)$) alors $\Gamma \models \varphi$

Preuve : Il suffit de raisonner par induction sur l'ensemble $Th(\Gamma)$:

- Si $\varphi \in \Gamma$ alors $\Gamma \models \varphi$ par définition même de cette notation.
- Sinon, φ est obtenu par application d'une règle du calcul de Hilbert, et c'est donc soit un axiome ($\frac{}{\varphi}$), soit le modus ponens ($\frac{\psi \quad \psi \to \varphi}{\varphi}$).
  - Si c'est un axiome alors le lemme 5 s'applique immédiatement.
  - Si c'est le modus ponens, on prend donc pour hypothèses d'induction $\Gamma \models \psi$ d'une part et $\Gamma \models (\psi \to \varphi))$ d'autre part, et l'on doit prouver $\Gamma \models \varphi$. C'est justement le lemme 4.

Ceci termine la preuve. □

[Fin du quatrième cours]

4 Complétude du calcul propositionnel

L'objectif de cette section est de faire comprendre comment on établit la complétude du calcul de Hilbert (tout ce qui est sémantiquement vrai est prouvable), qui s'énonce comme suit :

Théorème 7 : Pour toute signature P de la logique propositionnelle, pour tout ensemble de formules $\Gamma \subset For(P)$ et pour toute formule $\varphi \in For(P)$, si $\Gamma \models \varphi$ alors $\Gamma \vdash \varphi$.

Nous ne chercherons pas ici a donner une preuve de ce théorème dans toute sa généralité. En effet, le but de ce cours n'est pas la logique en elle même ou pour un quelconque intérêt mathématique ; il est de donner une formation de base sur les techniques employées dans le domaine de la logique, afin de savoir les utiliser pour les méthodes formelles en informatique. Il s'avère que la preuve du théorème de complétude précédent présente des « longueurs techniques » qui font immanquablement perdre le fil directeur de la preuve. On peut considérablement écourter ces longueurs si l'on se restreint au cas où P ne contient qu'un *nombre fini* de variables propositionnelles. C'est donc ce que nous ferons dans cette section.

4.1 Fil directeur de la preuve

TBFL (To Be Filled Later). (Voir vos notes de cours. Ne sera pas pénalisant pour l'examen cette année.)

4.2 La preuve

TBFL. (Notes manuscrites jointes. Ne sera pas pénalisant pour l'examen cette année.)

5 Déduction naturelle pour la logique propositionnelle

Comme on peut le constater à plusieurs reprises dans ce qui précède, il n'est pas facile de mener une preuve avec le calcul de Hilbert, aussi simple que puisse paraître la formule à prouver. La raison de cette difficulté est principalement que la seule règle de ce calcul qui permette de réduire la taille des formules manipulées est le *modus ponens* ; toutes les autres règles sont des schémas d'axiomes que l'on doit introduire judicieusement dans les preuves. Ceci conduit à une méthode

de preuve où l'on est amené à « construire par des détours » chaque formule intermédaire dont on peut avoir besoin. Il faut à chaque étape « parachuter » la bonne instance d'axiome de telle sorte qu'après application de quelques *modus ponens* on obtienne la formule intermédaire visée. Par exemple, si à une certaine étape de ma preuve j'ai prouvé (Flemmard $\vee$ Bosseur) et (Flemmard $\rightarrow$ FautBosser) et (Bosseur $\rightarrow$ FautBosser), et si je veux prouver FautBosser (ce qui déductible de manière parfaitement évidente), il faut que je parachute l'axiome (Flemmard $\rightarrow$ FautBosser) $\rightarrow$ ((Bosseur $\rightarrow$ FautBosser) $\rightarrow$ ((Flemmard $\vee$ Bosseur) $\rightarrow$ FautBosser)) et que j'applique rien moins que 3 *modus ponens*.

En fait, si l'on analyse ce qui conduit à parachuter l'axiome précédent, c'est que l'on « voit » ses deuxième, quatrième et dernière implications comme des « $\vdash$ » plutôt que des « $\rightarrow$ ». Bref, l'œil du prouveur voit de manière beaucoup plus naturelle la règle suivante : $\dfrac{(\varphi \vee \psi) \quad (\varphi \rightarrow \eta) \quad (\psi \rightarrow \eta)}{\eta}$.

Par ailleurs, comme on l'a déjà souligné, $(\varphi \rightarrow \eta)$ sera lui même établi de manière beaucoup plus naturelle en utilisant le théorème de la déduction, c'est-à-dire que l'on prouvera en fait $(\Gamma \cup \{\varphi\} \vdash \eta)$ plutôt que $(\Gamma \vdash (\varphi \rightarrow \eta))$. C'est la même chose pour $(\psi \rightarrow \eta)$. Bref, la règle de déduction la plus naturelle est donc la suivante : $\dfrac{\Gamma \vdash (\varphi \vee \psi) \quad \Gamma \cup \{\varphi\} \vdash \eta \quad \Gamma \cup \{\psi\} \vdash \eta}{\Gamma \vdash \eta}$.

Il est important de remarquer que la règle de déduction précédente ne porte pas sur le même ensemble de formules que les règles de Hilbert. En effet, elle ne porte pas sur des φ ou ψ de la logique des propositions, mais sur des $(\Gamma \vdash \varphi)$ où Γ dénote un ensemble de formules et φ une formule. Par conséquent, $\vdash$ est ici un symbole comme un autre plutôt qu'un prédicat d'inférence. Pour cette raison, on va le noter $\mid\!\sim$ au lieu de $\vdash$ afin de ne pas confondre le symbole $\mid\!\sim$ qui participe à la formation des *formules* $(\Gamma \mid\!\sim \varphi)$ avec le *prédicat* d'inférence de la logique propositionnelle qui porte sur $\mathcal{P}(For(P)) \times For(P)$. L'ensemble de formules sur lequel travaille cette déduction naturelle est donc l'ensemble $Infer(P)$ des $(\Gamma \mid\!\sim \varphi)$ où Γ est un sous-ensemble fini de $For(P)$ et $\varphi \in For(P)$.

La constitution de règles de déductions naturelles que nous avons faite sur la disjonction s'applique en fait à tous les connecteurs, ce qui donne en fin de compte les règles de *la déduction naturelle* ci-dessous.

1. Pour toute formule $\varphi \in \Gamma$: $\dfrac{}{\Gamma \mid\!\sim \varphi}$ [introduction d'hypothèse]
2. $\dfrac{\Gamma \cup \{\varphi\} \mid\!\sim \psi}{\Gamma \mid\!\sim (\varphi \rightarrow \psi)}$ [$\rightarrow$-introduction]
3. $\dfrac{\Gamma \mid\!\sim (\varphi \rightarrow \psi) \quad \Gamma \mid\!\sim \varphi}{\Gamma \mid\!\sim \psi}$ [$\rightarrow$-élimination]
4. $\dfrac{\Gamma \mid\!\sim \varphi \quad \Gamma \mid\!\sim \psi}{\Gamma \mid\!\sim (\varphi \wedge \psi)}$ [$\wedge$-introduction]
5. $\dfrac{\Gamma \mid\!\sim (\varphi \wedge \psi)}{\Gamma \mid\!\sim \varphi}$ [$\wedge$-élimination-1]
6. $\dfrac{\Gamma \mid\!\sim (\varphi \wedge \psi)}{\Gamma \mid\!\sim \psi}$ [$\wedge$-élimination-2]
7. $\dfrac{\Gamma \mid\!\sim \varphi}{\Gamma \mid\!\sim (\varphi \vee \psi)}$ [$\vee$-introduction-1]
8. $\dfrac{\Gamma \mid\!\sim \psi}{\Gamma \mid\!\sim (\varphi \vee \psi)}$ [$\vee$-introduction-2]
9. $\dfrac{\Gamma \mid\!\sim (\varphi \vee \psi) \quad \Gamma \cup \{\varphi\} \mid\!\sim \eta \quad \Gamma \cup \{\psi\} \mid\!\sim \eta}{\Gamma \mid\!\sim \eta}$ [$\vee$-élimination]
10. $\dfrac{\Gamma \cup \{\varphi\} \mid\!\sim (\psi \wedge \neg\psi)}{\Gamma \mid\!\sim \neg\varphi}$ [$\neg$-introduction] (Si φ infère une absurdité, on en déduit $\neg\varphi$; en fait, le choix de la formule ψ importe peu.)
11. $\dfrac{\Gamma \mid\!\sim (\psi \wedge \neg\psi)}{\Gamma \mid\!\sim \varphi}$ [$\neg$-élimination] (Une absurdité suffit pour déduire n'importe quelle formule.)

12. $\dfrac{}{\Gamma \mid\!\sim (\varphi \vee \neg\varphi)}$ [tiers exclus]

Notons $\vdash^{Hilbert}$ le prédicat d'inférence issu du calcul de Hilbert et notons $\vdash^{deduction}$ le prédicat d'inférence défini par $\Gamma \vdash^{deduction} \varphi$ si et seulement si $(\Gamma \mid\!\sim \varphi)$ est un théorème de la déduction naturelle. Il est alors légitime de se demander si ces deux inférences définissent le même prédicat sur $\mathcal{P}(For(P)) \times For(P)$, et donc la même logique générale. La réponse est oui :

Théorème 8 : Le symbole d'inférence défini par la déduction naturelle coïncide avec le prédicat d'inférence du calcul propositionnel.

La preuve est laissée en exercice : pour prouver que les deux prédicats $\vdash^{Hilbert}$ et $\vdash^{deduction}$ sont égaux, il suffit, par un argument simple d'induction sur la longueur des preuves, de prouver :

- que pour toute règle $\dfrac{\varphi_1 \quad \cdots \quad \varphi_n}{\psi}$ du calcul de Hilbert, la formule $(\{\varphi_1 \ , \cdots , \ \varphi_n\} \ \mid\!\sim \ \psi)$ est un théorème du calcul de la déduction naturelle.
- et que pour toute règle $\dfrac{\Gamma_1 \mid\!\sim \varphi_1 \ , \cdots , \ \Gamma_n \mid\!\sim \varphi_n}{\Gamma \mid\!\sim \varphi}$ de la déduction naturelle, si l'on a $\Gamma_i \vdash^{Hilbert} \varphi_i$ pour $i = 1..n$ alors on a $\Gamma \vdash^{Hilbert} \varphi$

Il résulte donc du théorème précédent que la déduction naturelle définit un prédicat d'inférence correct et complet pour la logique des propositions.

En fait, hormis son intérêt historique, la version de Hilbert ne reste d'actualité que parce qu'elle écourte un peu les preuves mathématiques de correction et complétude ; les inférences formelles du calcul sont par contre considérablement plus faciles à mener avec la déduction naturelle. [Exercices : refaites quelques preuves déjà données en exercice en utilisant la déduction naturelle et comparez.]

Il existe ainsi plusieurs calculs formels qui peuvent être utilisés pour mécaniser le même prédicat d'inférence de la logique des propositions. Un calcul très connu est le *calcul des séquents*, qui sera présenté sous une forme complète plus loin pour la logique des prédicats. Ce calcul des séquents présente la particularité d'être bien adapté pour automatiser (au moins partiellement) les preuves. [Fin du cinquième cours]

Chapitre 4 : La logique des prédicats du premier ordre

1 Motivation

Le calcul propositionnel est mal adapté pour pratiquer la pensée déductive étudiée depuis l'antiquité. Elle donne simplement la structure générale (c-à-d., son squelette) de tout raisonnement déductif mais sans aucune indication sur les objets manipulés par cette déduction. Par exemple, si l'on veut modéliser le raisonnement suivant :

Tous les hommes sont mortels
Socrate est un homme
Donc, Socrate est mortel

Le calcul propositionnel nous permet simplement au moyen des trois variables propositionnelles p q et r représentant les trois phrases ci-dessus, de modéliser le raisonnement par la simple structure suivante :

$$\{p, q\} \vdash r$$

Cependant, on ne voit pas les relations sous-jacentes qui lient ces variables propositionnelles. Le calcul des prédicats répond à ce manque de description en augmentant son lexique linguistique. Maintenant, ce dernier va contenir en plus des variables propositionelles, des variables d'individu, x, y, z ... mises pour des objets de nature indéterminée, des fonctionnalités et des relations sur ces objets, ainsi que des quantificateurs existentiel ($\exists$) et universel ($\forall$) pour effectuer des "jugements" sur les objets manipulés. Un tel langage va alors nous permettre de faire une division des propositions en sujets et prédicats. Pour revenir à notre exemple de départ, on pourra alors considérer un symbole de prédicat $H(x)$ qui intuitivement veut dire x *est un homme*, un symbole fonctionnel *socrate* pour désigner l'individu *Socrate* et un symbole de prédicat $M(x)$ pour signifier que x *est mortel*. Le raisonnement précédent se modélisera alors par :

$$\{\forall x,\ (H(x) \Rightarrow M(x)), H(socrate)\} \vdash M(socrate)$$

où "$\forall$" se lit *quel que soit*.

2 Institution des prédicats du 1er ordre

Comme à notre habitude depuis le début du cours, nous allons définir dans cette section, les notions de *signature* et de *morphisme entre signatures* pour obtenir l'ensemble $Sign$, la classe des modèles au-dessus d'une signature pour caractériser l'application Mod, l'ensemble des formules construites à partir d'une signature pour l'application For, et enfin la relation de satisfaction $\models$ dans le cadre du premier ordre.

2.1 Signatures et morphismes de signatures

Une signature du premier ordre sera donc définie par un ensemble de constantes, de fonctions et de prédicats, et aussi un ensemble de variables pour dénoter des individus génériques sur lesquels on exprimera des propriétés d'existence ou d'universalité. De plus, ces derniers seront utiles pour définir les autres éléments syntaxiques que sont les *termes* et les *formules*.

Définition 1 : Une *signature* Σ est un triplet $(\mathcal{F}, \mathcal{R}, \mathcal{V})$ où :
- $\mathcal{F}$ est un ensemble de noms de fonction, chacun muni d'un entier naturel appelé son *arité*. Dans la suite, pour désigner un nom de fonction f munie d'une arité n, nous noterons : f^n. On appelle toute fonction muni d'arité nulle, une *constante*.
- $\mathcal{R}$ est un ensemble de noms de prédicat, chacun muni d'un entier naturel non nul appelé son *arité*. Dans la suite, pour désigner un nom de prédicat r muni d'une arité n, nous noterons : R^n.
- $\mathcal{V}$ est un ensemble dont les éléments sont appelés des *variables* tel que aucun des noms de constantes, de fonctions, et de prédicats n'appartiennent à cet ensemble.

Comme exemples de signature, on peut considérer :

1. la signature utilisée pour spécifier l'arithmétique usuelle (appelée aussi l'arithmétique de Peano), et définie par une constante 0, un symbole de fonction unaire *succ*, un symbole de fonction binaire + pour désigner la somme entre entiers, et un symbole de prédicat binaire "=" pour dénoter l'égalité usuelle entre entiers. Si nous notons Σ_{Peano} cette signature, nous avons alors : $\Sigma_{Peano} = (\mathcal{F}_{Peano}, \mathcal{R}_{Peano}, \mathcal{V}_{peano})$ où $\mathcal{F}_{Peano} = \{0^0, succ^1, +^2\}$, $\mathcal{R}_{Peano} = \{=^2\}$, et $\mathcal{V}_{Peano}$ est l'ensemble des lettres minuscules.
2. la signature utilisée pour spécifier l'arithmétique de Presburger [7]. À la différence de la signature Σ_{Peano}, nous avons deux constantes 0 et 1, un unique symbole de fonction + d'arité 2 , et deux symboles de prédicat d'arité 2 : "$<$" pour désigner la relation d'ordre totale sur $\mathbb{N}$, et "=" pour l'égalité entre entiers. On obtient la signature Σ_{Presb} suivante : $\Sigma_{Presb} = (\mathcal{F}_{Presb}, \mathcal{R}_{Presb}, \mathcal{V}_{Preb})$ où $\mathcal{F}_{Presb} = \{0^0, 1^0, +^2\}$, $\mathcal{R}_{Presb} = \{<^2, =^2\}$, et $\mathcal{V}_{Presb}$ est l'ensemble des lettres minuscules.
3. enfin, la signature associée à la théorie des ensembles de Zermelo-Fraenkel (ZF), composée simplement des deux symboles de prédicats binaires (c-à-d., munies d'une arité 2) : "=" pour dénoter le symbole d'égalité habituel entre ensembles, et "$\in$" pour dénoter l'appartenance. On a alors la signature Σ_{ZF} suivante : $\Sigma_{ZF} = (\mathcal{F}_{ZF}, \mathcal{R}_{ZF}, \mathcal{V}_{ZF})$ où $\mathcal{F}_{ZF} = \emptyset$, $\mathcal{R}_{ZF} = \{=^2, \in^2\}$, et $\mathcal{V}_{ZF}$ est l'ensemble des lettres minuscules.

Remarque 2 : Dans la suite, tout symbole de prédicat binaire (c-à-d., d'arité 2) sera notée sous sa forme infixe.

Définition 3 : Soient $\Sigma_1 = (\mathcal{F}_1, \mathcal{R}_1, \mathcal{V}_1)$ et $\Sigma_2 = (\mathcal{F}_2, \mathcal{R}_2, \mathcal{V}_2)$ deux signatures. Un *morphisme de signatures* $\sigma : \Sigma_1 \to \Sigma_2$ est défini par les trois applications suivantes, aussi dénotées par σ pour simplifier :
1. $\sigma : \mathcal{F}_1 \to \mathcal{F}_2$ telle que pour toute fonction $f \in \mathcal{F}_1$ d'arité $n \in \mathbb{N}$, $\sigma(f) \in \mathcal{F}_2$ est d'arité n,
2. $\sigma : \mathcal{R}_1 \to \mathcal{R}_2$ telle que pour tout prédicat $R \in \mathcal{R}_1$ d'arité $n \in \mathbb{N}$, $\sigma(R) \in \mathcal{R}_2$ est d'arité n.
3. $\sigma : \mathcal{V}_1 \to \mathcal{V}_2$ telle qu'elle est injective [8].

[7] L'arithmétique de Presburger est une théorie décidable, c'est-à-dire qu'il existe un algorithme pour décider si une formule est valide ou non pour cette théorie. À l'inverse, l'arithmétique usuelle de Peano dont la signature est donnée ci-dessus, a été démontrée indécidable (fameuse conséquence du non moins fameux théorème d'incomplétude de Gödel).

[8] Nous imposons cette condition car sans elle, la logique des prédicats du 1er ordre ne vérifierait pas la condition de satisfaction (cf. ci-après). Enfin, elle n'ajoute ou n'enlève aucun pouvoir d'expression à ce formalisme de description.

À titre d'exemple, nous pouvons enrichir la signature Σ_{Peano} en ajoutant le symbole de fonction $pred^1$ pour dénoter la fonction prédecesseur, ainsi que les deux symboles de prédicats unaires $Positif$ et $Negatif$. Nous obtenons alors la signature Σ'_{Peano} suivante : $\Sigma'_{Peano} = (\mathcal{F}'_{Peano}, \mathcal{R}'_{Peano}, \mathcal{V}'_{Peano})$ où $\mathcal{F}'_{Peano} = \{0^0, succ^1, pred^1, +^2\}$, $\mathcal{R}'_{Peano} = \{=^2, Positif^1, Negatif^1\}$, et $\mathcal{V}'_{Peano} = \mathcal{V}_{Peano}$. De là, nous avons alors le morphisme de signature ν défini comme l'identité sur les ensembles des constantes et des variables, et comme l'inclusion sur les ensembles des fonctions et des prédicats.

2.2 Modèles

Un modèle (ou structure) donne un *sens mathématique* aux composantes de notre signature. Il est décrit simplement au moyen d'un ensemble muni d'opérations internes (ou applications) et de relations avec éventuellement ce que l'on a coutume d'appeler des "éléments distingués" [9]. L'ensemble caractérisera le *domaine des individus*, les lois internes donneront une signification mathématique (ou sémantique) aux fonctions et les relations aux symboles de prédicats de la signature.

Définition 4 : Étant donnée une signature $\Sigma = (\mathcal{F}, \mathcal{R}, \mathcal{V})$, un *modèle au dessus de* Σ ou encore un Σ-modèle $\mathcal{M}$, est la donnée d'un ensemble M muni :
- pour chaque symbole de fonction f d'arité n, d'une application $f^{\mathcal{M}} : M^n \to M$ (si $n = 0$, $f^{\mathcal{M}}$ est une constante de M)
- pour chaque symbole de prédicat R d'arité n, d'un sous-ensemble $R^{\mathcal{M}}$ de M^n.

On note $Mod(\Sigma)$ la classe des Σ-modèles.

À partir des exemples donnés dans la section 2.1, on peut considérer les structures suivantes :
- pour la signature associée à l'arithmétique de Peano, le modèle $\mathcal{M}_{Peano}$:

$$\mathcal{M}_{Peano} =< \mathbb{N}, 0_{\mathbb{N}}, succ_{\mathbb{N}}, +_{\mathbb{N}}, =_{\mathbb{N}} >$$

Le modèle $\mathcal{M}_{Peano}$ est celui auquel on pense comme modèle canonique de l'arithmétique usuelle (et dont les axiomes vont être donnés dans la section 2.3). Cependant, il n'est pas unique [10]. En effet, tous les modèles $\mathcal{M}^n_{Peano}$ où $n \geq 2$ est un entier et définis par :

$$M = \mathbb{Z}/n\mathbb{Z} \qquad \begin{array}{c} succ^{\mathcal{M}} : M \to M \\ m \mapsto (m +_{\mathbb{N}} 1)\ mod\ n \end{array} \qquad \begin{array}{c} +^{\mathcal{M}} : M \times M \to M \\ (m,p) \mapsto (m +_{\mathbb{N}} p)\ mod\ n \end{array} \qquad \equiv \mod n$$

sont aussi des modèles qui valident l'arithmétique de Peano.
- pour la signature associée à l'arithmétique de Presburger, le modèle $\mathcal{M}_{Presb}$:

$$\mathcal{M}_{Presb} =< \mathbb{N}, 0_{\mathbb{N}}, 1_{\mathbb{N}}, +_{\mathbb{N}}, =_{\mathbb{N}}, <_{\mathbb{N}} >$$

- pour la signature de la théorie des ensembles selon ZF, le modèle $\mathcal{M}_{ZF}$ dont le domaine M_{ZF} est défini inductivement par :

$$M_0 = E,\ M_n = \mathcal{P}(M_{n-1}),\ M_{ZF} = \bigcup_n \mathcal{P}(M^n)$$

$\mathcal{P}(M)$ dénote l'ensemble des parties de M, E est un ensemble, et muni de l'égalité usuelle entre ensembles pour le symbole de prédicat "=", et de la relation $\in^{\mathcal{M}}$ suivante :

$x \in^{\mathcal{M}} y$ si et seulement s'il existe $n \in \mathbb{N}$ tel que $x \in M_n$, $y \in M_{n+1}$, et $x \in y$ [11]

[9] En mathématique, on parle alors de structures.

[10] Ceci est aussi vrai pour la théorie des ensembles de ZF. Par contre, il n'existe qu'un unique modèle pour la théorie de l'arithmétique de Presburger (toujours une conséquence de sa décidabilité).

[11] Attention, ici, $\in$ désigne l'appartenance usuelle dans les ensembles. $\in^{\mathcal{M}}$ est le sens mathématique donné au symbole de prédicat "$\in^2$" de la signature Σ_{ZF}.

on a alors : $\forall y \in E,\ \forall x \in M,\ x \not\in^{\mathcal{M}} y$

Pour définir complètement Mod, nous devons aussi définir comment l'application Mod transporte un morphisme de signature $\sigma : \Sigma_1 \to \Sigma_2$, c'est-à-dire, à un Σ_2-modèle $\mathcal{M}$, comment on doit définir le Σ_1-modèle $\mathcal{U}_\sigma(\mathcal{M})$.

Définition 5 : Soient $\sigma : \Sigma_1 \to \Sigma_2$ un morphisme de signature. L'application $\mathcal{U}_\sigma : Mod(\Sigma_2) \to Mod(\Sigma_1)$ est défini par : pour tout Σ_2-modèle $\mathcal{M}$, $\mathcal{U}_\sigma(\mathcal{M})$ est le Σ_1-modèle $\mathcal{B}$ où :
- l'ensemble sous-jacent B est égal à M,
- pour chaque fonction $f \in \mathcal{F}$ on a $f^{\mathcal{B}} = \sigma(f)^{\mathcal{M}}$,
- et pour chaque prédicat $R \in \mathcal{R}$ on a $R^{\mathcal{B}} = \sigma(R)^{\mathcal{M}}$.

2.3 Termes et formules

Nous avons tous les ingrédients pour définir la notion de formules pour la logique du premier ordre. Comme on l'a vu dans la section 1, la logique des prédicats a pour but de décrire formellement des propriétés sur des individus. Pour désigner ces individus, nous avons alors besoin d'un premier élément syntaxique, les *termes*.

Définition 6 : Étant donnée une signature $\Sigma = (\mathcal{F}, \mathcal{R}, \mathcal{V})$, l'ensemble $T_\Sigma(\mathcal{V})$ des *termes avec variables dans V* de la signature Σ est le plus petit ensemble (au sens de l'inclusion) tel que :
- il contient toutes les variables et les symboles de constantes (c-à-d., l'ensemble $\mathcal{V}$ et les symboles de fonctions d'arité 0),
- pour chaque symbole de fonction $f \in \mathcal{F}$ d'arité $n \neq 0$, et chaque n-uplet $(t_1, \ldots, t_n)$ de $T_\Sigma(\mathcal{V})^n$, $f(t_1, \ldots, t_n)$ est un terme de $T_\Sigma(\mathcal{V})$.

Nous pouvons maintenant aborder la notion de formules sur une signature Σ. Ces dernières sont définies de façon inductive à partir de l'ensemble de base des formules dites *atomiques*, des connecteurs propositionnels, et des quantificateurs existentiel ($\exists$) et universel($\forall$).

Définition 7 : Étant donnée une signature $\Sigma = (\mathcal{F}, \mathcal{R}, \mathcal{V})$, une Σ-*formule atomique* est une expression de la forme $R(t_1, \ldots, t_n)$ où R est un symbole de prédicats d'arité n et $(t_1, \ldots, t_n)$ un n-uplet de $T_\Sigma(\mathcal{V})^n$.

À partir des formules atomiques, nous pouvons définir les formules plus générales.

Définition 8 : Étant donnée une signature Σ dont l'ensemble des variables est $\mathcal{V}$, l'ensemble $For(\Sigma)$ des Σ-*formules bien-formées au-dessus de* Σ est le plus petit ensemble (au sens de l'inclusion) tel que :
- il contient toutes les formules atomiques,
- à chaque fois qu'il contient deux mots φ et ψ, il contient également :

$$\neg\varphi,\ \varphi \wedge \psi,\ \varphi \vee \psi,\ \varphi \Rightarrow \psi,\ \varphi \Leftrightarrow \psi$$

- à chaque fois qu'il contient un mot φ, contient pour toute variable $x \in \mathcal{V}$:

$$\forall x.\varphi,\ \exists x.\varphi$$

Toujours à partir des trois signatures développées dans la section 2.1, nous pouvons considérer les exemples de formules suivantes :

- pour la signature associée à l'arithmétique de Peano, on a les axiomes suivants :
  - 0 n'est successeur de personne : $\forall x.\neg(succ(x) = 0)$

- tout individu autre que 0 est successeur de quelqu'un :

$$\forall x.\exists y.(\neg(x = 0) \Rightarrow succ(y) = x)$$

- *succ* est une application injective :$\forall x.\forall y.(succ(x) = succ(y) \Rightarrow x = y)$
- définition de + via les constructeurs *succ* et 0 :
  - $\forall x.(x + 0 = x)$
  - $\forall x.\forall y.(x + succ(y) = succ(x + y))$

  Enfin, nous devons ajouter, la formule suivante :
  $(\varphi(x/0) \wedge \forall x.(\varphi(x) \Rightarrow \varphi(x/succ(x))) \Rightarrow \forall x.\varphi(x)$

  Ce dernier schéma de formules décrit une induction pour la formule φ. [12]
- pour la signature associée à l'arithmétique de Presburger, on a les axiomes suivants :
  - 0 est neutre pour l'addition : $\forall n.(n + 0 = n)$
  - "+" est associatif : $n + (m + p) = (n + m) + p$
  - "<" est un ordre strict, total et discret :
    - "<" est un ordre strict et total :
      - Anti-réflexif : $\forall n.\neg(n < n)$
      - transitif : $\forall x.\forall y.\forall z.(x < y \wedge y < z \Rightarrow x < z)$
      - anti-symétrique : $\forall x.\forall y.\neg(x < y \wedge y < x)$
      - total : $\forall x.\forall y.(x < y \vee y < x \vee x = y)$
    - "<" est discret :
      - < n'est pas dense : $\forall x.\exists y.(x < y \wedge \forall z.(x < z \Rightarrow (y < z \vee y = z)))$
      - tout élément excepté 0 a un unique prédécesseur :

$$\forall x.\forall y.(y < x \Rightarrow \exists z.\forall w.(z < x \wedge (z < w \Rightarrow x < w \vee x = w)))$$

  - tout élément est plus petit que son successeur direct : $\forall n.n < n + 1$
- pour la signature associée à la théorie des ensembles de ZF, on a parmi tous les axiomes de cette théorie, les formules suivantes :
  - **L'axiome d'extensionnalité**

$$\forall x.\forall y.(\forall z.(z \in x \Leftrightarrow z \in y) \Rightarrow x = y)$$

    Cette formule dit simplement que deux ensembles x et y composés des mêmes éléments sont égaux entre eux.
  - **L'axiome de la réunion**

$$\forall x.\exists y.\forall z.(z \in y \Leftrightarrow \exists w(w \in x \wedge z \in w))$$

    Cette formule assure l'existence d'un ensemble y (qui par l'axiome d'extensionnalité est unique) comme étant la réunion de tous les ensembles de l'ensemble x (c-à-d., $y = \bigcup_{w \in x} w$).
  - **L'axiome des parties**

$$\forall x.\exists y.\forall z.(z \in y \Leftrightarrow \forall w(w \in z \Rightarrow w \in x))$$

    Cette formule affirme qu'étant donné un ensemble x, il existe un ensemble y dont les éléments sont exactement les parties de x.

Comme précédemment, nous devons aussi dire la façon dont sont transportés les morphismes de signatures via l'application For.

[12] Ce principe n'est pas exprimable dans le 1er ordre. Ceci vient du fait que ne pouvons pas quantifier sur les formules au premier ordre (ceci est une particularité de la logique dite du *second ordre*), ce qui nous oblige à considérer une infinité d'axiomes, un pour chaque formule φ.

Définition 9 : Soit $\sigma : \Sigma_1 \to \Sigma_2$ un morphisme de signatures. $\overline{\sigma}$ désigne le prolongement de σ aux termes et aux formules et est défini par :
- pour tout terme $t \in T_{\Sigma_1}(\mathcal{V}_1)$ de la forme $f(t_1, \ldots, t_n)$, $\overline{\sigma}(t)$ est le terme de $T_{\Sigma_2}(\mathcal{V}_2)$ de la forme $\sigma(f)(\overline{\sigma}(t_1), \ldots, \overline{\sigma}(t_n))$,
- pour toute Σ_1-formule atomique φ de la forme $R(t_1, \ldots, t_n)$, $\overline{\sigma}(\varphi)$ est la Σ_2-formule atomique de la forme $\sigma(R)(\overline{\sigma}(t_1), \ldots, \overline{\sigma}(t_n))$,
- pour toute Σ_1-formule φ de la forme $\neg\psi$, $\psi \wedge \pi$, $\psi \vee \pi$, ou $\psi \Rightarrow \pi$, $\overline{\sigma}(\varphi)$ est la Σ_2-formule de la forme $\neg\overline{\sigma}(\psi)$, $\overline{\sigma}(\psi) \wedge \overline{\sigma}(\pi)$, $\overline{\sigma}(\psi) \vee \overline{\sigma}(\pi)$, ou $\overline{\sigma}(\psi) \Rightarrow \overline{\sigma}(\pi)$,
- pour toute Σ_1-formule φ de la forme $\exists x.\psi$ (resp. $\forall x.\psi$), $\overline{\sigma}(\varphi)$ est la Σ_2-formule de la forme $\exists\overline{\sigma}(x).\overline{\sigma}(\psi)$ (resp. $\forall\overline{\sigma}(x).\overline{\sigma}(\psi)$).

2.4 Relation de satisfaction

Avant de donner une valeur de vérité à nos formules, nous devons tout d'abord donner un sens mathématique aux premiers éléments syntaxiques de ces dernières que sont les termes. Ceci va se faire via la notion de *substitution* (ou *interprétation*) des variables dans le domaine d'un modèle $\mathcal{M}$ donné.

Définition 10 : Étant donnés une signature Σ dont l'ensemble des variables est $\mathcal{V}$ et un Σ-modèle $\mathcal{M}$, une *interprétation* ν est une application de $\mathcal{V}$ dans M.

À partir d'une interprétation $\nu : \mathcal{V} \to M$, nous pouvons étendre de manière canonique cette dernière à une application $\nu^\sharp : T_\Sigma(\mathcal{V}) \to M$.

Théorème 11 : Soient une signature Σ dont l'ensemble des variables est $\mathcal{V}$ et $\mathcal{M}$ un Σ-modèle. Étant donnée une interprétation $\nu : \mathcal{V} \to M$, il existe une unique application $\nu^\sharp : T_\Sigma(\mathcal{V}) \to M$, extension de ν (c-à-d., pour tout $x \in \mathcal{V}$, $\nu^\sharp(x) = \nu(x)$).

Preuve : $\nu^\sharp$ est la même application que celle appelée *substitution* dans la section 3.1.4. Cette dernière est définie de façon inductive sur tous les termes de $T_\Sigma(\mathcal{V})$. Elle définie totalement une application de $T_\Sigma(\mathcal{V})$ dans M. Nous concluons alors directement sur l'existence et l'unicité de $\nu^\sharp$. □

Nous pouvons de même prolonger l'application ν en un prédicat unique, noté Mod^ν_Σ, sur les formules de $For(\Sigma)$ de la façon suivante :

Définition 12 : Soient Σ une signature dont l'ensemble des variables est $\mathcal{V}$, $\mathcal{M}$ un Σ-modèle, et φ une Σ-formule. $\mathcal{M}$ *satisfait* φ pour une interprétation $\nu : \mathcal{V} \to M$ donnée, notée $\mathcal{M} \models^\nu_\Sigma \varphi$, si et seulement si :
- si φ est une formule atomique de la forme $R(t_1, \ldots, t_n)$
  alors $\mathcal{M} \models^\nu_\Sigma \varphi$ ssi $(\nu^\sharp(t_1), \ldots, \nu^\sharp(t_n)) \in R^\mathcal{M}$,
- si φ est de la forme $\neg\psi$ alors $\mathcal{M} \models^\nu_\Sigma \varphi$ ssi $\mathcal{M} \not\models^\nu_\Sigma \psi$,
- si φ est de la forme $\psi \wedge \pi$ alors $\mathcal{M} \models^\nu_\Sigma \varphi$ ssi $\mathcal{M} \models^\nu_\Sigma \psi$ *et* $\mathcal{M} \models^\nu_\Sigma \pi$,
- si φ est de la forme $\psi \vee \pi$ alors $\mathcal{M} \models^\nu_\Sigma \varphi$ ssi $\mathcal{M} \models^\nu_\Sigma \psi$ *ou* $\mathcal{M} \models^\nu_\Sigma \pi$,
- si φ est de la forme $\psi \Rightarrow \pi$ alors $\mathcal{M} \models^\nu_\Sigma \varphi$ ssi si $\mathcal{M} \models^\nu_\Sigma \psi$ *alors* $\mathcal{M} \models^\nu_\Sigma \pi$,
- si φ est de la forme $\psi \Leftrightarrow \pi$ alors $\mathcal{M} \models^\nu_\Sigma \varphi$ ssi $\mathcal{M} \models^\nu_\Sigma \psi \Rightarrow \pi$ *et* $\mathcal{M} \models^\nu_\Sigma \pi \Rightarrow \psi$,
- si φ est de la forme $\exists x.\psi$ alors $\mathcal{M} \models^\nu_\Sigma \varphi$ ssi il existe une interprétation ν' qui affecte la même valeur à toutes les variables que ν excepté peut-être pour la variable x (dans ce cas-là, on dit que ν' est *x-équivalente* à ν), telle que $\mathcal{M} \models^{\nu'}_\Sigma \psi$,
- si φ est de la forme $\forall x.\psi$ alors $\mathcal{M} \models^\nu_\Sigma \varphi$ ssi $\mathcal{M} \not\models^\nu_\Sigma \exists x.\neg\psi$.

$\mathcal{M}$ *valide* φ, noté $\mathcal{M} \models_\Sigma \varphi$ si et seulement si pour toute interprétation $\nu : V \to A$, $\mathcal{M} \models^\nu_\Sigma \varphi$.

À titre d'exemple, montrons que le modèle défini sur la signature associée à la théorie des ensembles ZF donné en section 2.2 valide l'axiome de la réunion (AR) (cf. section 2.3). Pour cela, supposons une interprétation $\nu : \mathcal{V}_{ZF} \to A$ quelconque, et montrons que :

$$\mathcal{M} \models^{\nu}_{\Sigma} AR$$

Soit ν' une interprétation x-équivalente à ν. on a alors trois cas à considérer :

1. cas où $\nu'(x) \in E$. Il nous suffit alors de choisir n'importe quel interprétation ν'' y-équivalente à ν' telle que $\nu''(y) \in E$. En effet, pour toute interprétation ν''' z-équivalente à ν'', nous avons :

   $$\mathcal{M} \models^{\nu'''}_{\Sigma} (z \in y \Leftrightarrow \exists w.(w \in x \wedge z \in w))$$

   (la relation $\in^{\mathcal{M}}$ n'est pas vérifiée pour les éléments de E)
2. cas où $\nu'(x) \subseteq \mathcal{P}(E)$. Ici, le seul ensemble pour lequel la propriété ci-dessus est vérifiée est l'ensemble vide. Donc, il suffit de choisir une interprétation ν'' y-équivalente à ν' telle que $\nu''(y) = \emptyset$.
3. cas où $\nu'(x) \in M_n$ avec $n \geq 2$. Par définition, $\nu'(x)$ est un sous-ensemble de M_{n-1}. Donc, nous pouvons choisir l'interprétation ν'' y-équivalente à ν' telle que $\nu''(y) = \bigcup_{S \in \nu'(x)} S$. Cet ensemble existe car $\nu'(x)$ est un ensemble dont les éléments sont aussi des ensembles (au sens mathématique du terme). On peut donc faire une union entre eux.

Exercices :

1. Montrer que l'axiome des parties est aussi validé par le modèle $\mathcal{M}_{ZF}$ donné en section 2.2.
2. Montrer que tous les modèles $\mathcal{M}^n_{Peano}$ valident les axiomes définissant l'arithmétique de Peano.
3. Faire la même chose avec le modèle $\mathcal{M}_{Presb}$ et les axiomes associés à l'arithmétique de Presburger.

La logique des prédicats a été définie pour donner un fondement aux objets manipulés par les mathématiciens.

Définition 13 : Dans ce cadre, un ensemble d'hypothèses Γ sur une signature Σ (c-à-d., $\Gamma \subseteq For(\Sigma)$) est souvent appelée une Σ-*théorie*.
Un Σ-modèle $\mathcal{M}$ valide une Σ-théorie Γ, noté $\mathcal{M} \models \Gamma$ si et seulement si $\mathcal{M} \models \varphi$ pour tout $\varphi \in \Gamma$.
Enfin, une Σ_1-formule ψ est une *conséquence sémantique* d'une Σ-théorie Γ, notée $\Gamma \models \psi$, si et seulement si, pour tout Σ_1-modèle $\mathcal{M}$ si $\mathcal{M} \models \Gamma$ alors $\mathcal{M} \models \psi$.

Exercice : À partir des axiomes de l'arithmétique de Presburger donnés en section 2.3, montrez dans la section suivante que la formule qui suit est une conséquence sémantique des ces derniers :

$$\exists x.\forall y.((y < x \vee y = x) \Rightarrow x = y)$$

Cette formule signifie simplement que "$<$" possède un élément minimal qui dans notre cas est 0 (et donc fait bien de "$<$" un ordre discret).

2.5 La condition de satisfaction

Nous sommes en mesure de définir l'institution $(Sig, For, Mod, \models)$ dans le cadre de la logique du premier ordre :

– Sig est l'ensemble des signatures du premier ordre données à la définition 1.
– For est l'application qui à chaque signature Σ associe l'ensemble $For(\Sigma)$ des formules bien-formées de la définition 8.
– Mod est l'application qui à chaque signature Σ associe la classe $Mod(\Sigma)$ des Σ-modèles de la définition 4.
– $\models$ est la famille de relations indexées par Sig où pour chaque signature Σ de Sig, $\models_\Sigma$ est la relation de satisfaction définie à la définition 12.

La logique des prédicats du premier ordre vérifie la *condition de satisfaction*, c'est-à-dire, que nous avons la propriété suivante :

Théorème 14 : Soit $\sigma : \Sigma_1 \to \Sigma_2$ un morphisme de signatures. Nous avons alors pour tout Σ_2-modèle $\mathcal{M}$ et toute Σ_1-formule φ, la propriété suivante :

$$\mathcal{M} \models_{\Sigma_2} \overline{\sigma}(\varphi) \Longleftrightarrow Mod(\sigma)(\mathcal{M}) \models_{\Sigma_1} \varphi$$

Preuve : Notons $\mathcal{V}'$ le sous-ensemble de $\mathcal{V}_2$ tel que $\mathcal{V}' = \sigma(\mathcal{V}_1)$.

($\Longrightarrow$) Supposons $\mathcal{M} \models \overline{\sigma}(\varphi)$. Nous avons alors pour toute interprétation $\nu : \mathcal{V}' \to A$, $\mathcal{M} \models^{\nu}_{\Sigma_2} \overline{\sigma}(\varphi)$.
Soit $\nu' : \mathcal{V}_1 \to Mod(\sigma)(A)$ une interprétation quelconque. Nous définissons alors à partir de ν', l'interprétation $\nu^+ : \mathcal{V}' \to A$ par : $\nu^+(\sigma(x)) = \nu'(x)$.
Montrons alors que pour tout terme t de $T_{\Sigma_1}(\mathcal{V}_1)$, nous avons : $(\nu')^*(t) = (\nu^+)^*(\overline{\sigma}(t))$.

`laiss'ee en exercice.`

Nous avons alors pour tout Σ_1-formule atomique $R\ t_1 \dots t_n$:

$$((\nu^+)^*(\overline{\sigma}(t_1)), \dots, (\nu^+)^*(\overline{\sigma}(t_n))) \in \overline{\sigma}(R)^{\mathcal{M}} \Longleftrightarrow ((\nu')^*(t_1), \dots, (\nu')^*(t_1)) \in R^{Mod(\sigma)(\mathcal{M})}$$

Il facile d'étendre cette équivalence à toute Σ_1-formule par récurrence sur la taille des formules, et donc d'obtenir :

$$\mathcal{M} \models^{\nu^+}_{\Sigma_2} \overline{\sigma}(\varphi) \Longleftrightarrow Mod(\sigma)(\mathcal{M}) \models^{\nu'}_{\Sigma_1} \varphi$$

ce que nous voulons démontrer.

($\Longleftarrow$)

`laiss'ee en exercice`

□

[Fin cours 6]

3 Logique générale des prédicats du 1er ordre

Pour obtenir une logique générale, nous devons ajouter à notre institution $(Sig, For, Mod, \models)$, une famille $\vdash$ de relations indexées par Sig où pour chaque signature $\Sigma \in Sig$, $\vdash_\Sigma$ est un sous-ensemble

de $\mathcal{P}(For(\Sigma)) \times For(\Sigma)$. En logique du 1er ordre, nous avons plusieurs possibiltés pour définir cette relation $\vdash_\Sigma$ pour une signature $\sigma \in Sig$ donnée. Ici, nous allons en présenter principalement trois : l'axiomatisation de Frege-Hilbert, la déduction naturelle de Gentzen-Jaskowski, et le calcul des séquents de Gentzen. Chacune de ces possibilités est en fait un système formel (c-à-d., défni au moyen de règles de déduction) et tous ont le même pouvoir démonstratif. La différence réside surtout dans leur facilité d'utilisation.

3.1 La méthode de Frege-Hilbert

Cette méthode de preuve est en fait la plus mal-aisée à utiliser. Elles possèdent en effet certains inconvénients, entre autres de refléter assez mal la façon dont les preuves sont pensées par un cerveau humain [13]. De plus, à la différence des deux autres types de méthodes de preuves présentées ci-dessous, elles se prètent peu à l'analyse des preuves, appelée plus communément *théorie de la démonstration*, (preuves constructives et non-constructives, leurs structures (forme normale), les critères d'équivalence, ...) particulièrement importante pour l'informatique théorique. En revanche, c'est une méthode un peu plus proche de la façon dont sont écrites les preuves. Enfin, elle n'a besoin d'aucune notion supplémentaire pour être présentée (à la différence des deux autres systèmes de preuves).

3.1.1 Axiomes et règles

Ce système veut traduire la notion simple de démonstration suivante :

un théorème est déduit d'un ensemble de propostions données au départ au moyen de règles bien précises de déduction.

Il est défini au moyen de 15 schémas d'axiomes (c-à-d., de règles d'inférence d'arité 1) et de seulement 2 règles de déductions.

Les axiomes logiques

Les axiomes logiques sont toutes les formules que l'on peut obtenir à partir des 10 schémas d'axiomes donnés pour le calcul propostionnel en substituant les variables propostionnelles par n'importe quelle formule du 1er ordre. On ajoute en plus 5 schémas d'axiomes du aux nouveaux symboles du langage. Pour cela, nous devons d'abord définir la notion de *variables libre dans une formule*, et la notion de *substitution d'un terme dans une formule*.

Définition 15 : Une occurence de variable x dans une formule φ est *libre* si on ne trouve aucun quantificateur de la forme $\exists x$ ou $\forall x$ dans le chemin de cette occurence à ce quantificateur lorsque l'on représente la formule φ sous-forme d'arbre. Si une occurence de la variable x dans φ n'est pas libre alors elle est dite *liée*.
On dit qu'une variable x est *libre* dans une formule φ, et on le note $\varphi(x)$ si elle y a au moins une occurence libre.

Définition 16 : Étant donnés une formule φ et un terme t, on note $\varphi(x/t)$ la formule obtenue en remplaçant toute occurence libre de la variable x par t avec la condition qu'aucune des variables de t n'apparaissent comme variable liée dans t.

[13] Certains diraient de mathématiciens mais ne prétons pas la rigueur qu'à ces gens-là.

On peut maintenant donner les nouveaux schémas d'axiomes.

1. $(\forall x.\varphi) \Rightarrow \varphi(x/t)$ [instantiation]
2. $(\exists x.\varphi) \Rightarrow (\neg\forall x.\neg\varphi)$ [existence-a]
3. $(\neg\forall x.\neg\varphi) \Rightarrow (\exists x.\varphi)$ [existence-b]
4. $(\forall x.\varphi) \Rightarrow (\forall y.\varphi(x/y))$ [renomage]
5. $\forall x(\psi \Rightarrow \varphi) \Rightarrow (\psi \Rightarrow (\forall x.\varphi))$ [distribution]
   où x n'a pas d'occurence libre dans ψ.

Les règles de déduction

Nous avons bien entendu le modus ponens qui est à la base du moindre raisonnement formel, ainsi que la nouvelle règle, appelée *règle de généralisation* qui se définit par :

$$\frac{\varphi}{\forall x.\varphi}$$

Plus succintement, elle traduit le fait que toute variable libre dans une formule est en fait quantifiée universellement. Le pendant sémantique de cette règle se trouve dans la notion de validité d'une formule φ pour un modèle $\mathcal{A}$ donné, dans le sens où l'on regarde la satisfaction de φ dans $\mathcal{A}$ pour toutes les interprétations ν (et donc celles de la variable x).

3.1.2 En tant que systèmes formels

Il est bien entendu très simple de définir pour chaque signature signature $\Sigma \in Sig$, le tripet $C_\Sigma = (A_\Sigma, \mathcal{F}_\Sigma, \mathcal{R}_\Sigma)$ à partir du système d'inférence ci-dessus :

- A_Σ est l'alphabet $\mathcal{F} \cup \mathcal{F} \cup \mathcal{R} \cup \mathcal{V} \cup \{\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow, \forall, \exists.(,)\}$
- $\mathcal{F}_\Sigma = For(\Sigma)$
- enfin, $\mathcal{R}_\Sigma$ est l'ensemble des règles obtenues à partir des schémas de règles donnés précédemment.

Comme nous l'avons déjà fait dans le cours sur les systèmes formels, nous posons : $\vdash_\Sigma = \vdash_{C_\Sigma}$. De plus, nous avons déjà démontré que cette relation vérifiait les propriétés de : réflexivité, transitivité, monotonie et compacité. Il nous reste simplement à vérifier la propriété de $\vdash$-translation.

Théorème 17 : Pour tout morphisme de signature $\sigma : \Sigma \to \Sigma'$, toute Σ-théorie $\Gamma \subseteq For(\Sigma)$ et toute formule $\varphi \in For(\Sigma)$, nous avons :

$$\Gamma \vdash_\Sigma \varphi \Longrightarrow \overline{\sigma}(\Gamma) \vdash_{\Sigma'} \overline{\sigma}(\varphi)$$

`la preuve est laiss'ee en exercice`

Pour obtenir une logique générale selon la définition donnée en cours, nous devons montrer que toute relation $\vdash_\Sigma$ est correcte :

Théorème 18 : Étant donnée une Σ-théorie Γ et une Σ-formule φ, on a :

$$\Gamma \vdash_\Sigma \varphi \Longrightarrow \Gamma \models_\Sigma \varphi$$

`la preuve est laiss'ee en exercie`

Maintenant, le uplet $(Sig, For, Mod, \models, \vdash)$ où $\vdash = \{\vdash_\Sigma\}_{\Sigma \in Sig}$ est une *logique générale*.

3.1.3 Exemple de preuve

À titre d'exemple d'utilisation, montrons que la formule $\neg\forall x.\varphi \Rightarrow \exists x.\neg\varphi$ peut se déduire à partir de n'importe quelle relation $\vdash_\Sigma$ (c-à-d., $\vdash_\Sigma \neg\forall x.\varphi \Rightarrow \exists x.\neg\varphi$). Par le théorème de la déduction qui m,arche aussi pour la logique des prédicats du 1er ordre, ceci revient à montrer : $\{\neg\forall x.\varphi\} \vdash_\Sigma \exists x.\neg\varphi$. Nous avons alors les étapes de preuve suivante :

1. $\{\neg\forall x.\varphi\} \vdash_\Sigma (\neg\forall x.\neg\varphi) \Rightarrow (\exists x.\varphi)$ [existence-b]
2. $\{\neg\forall x.\varphi\} \vdash_\Sigma (\neg\forall x.\neg\neg\varphi) \Rightarrow (\exists x.\neg\varphi)$ [existence-b] $[\varphi/\neg\varphi]$
3. $\{\neg\forall x.\varphi\} \vdash_\Sigma \neg\neg\varphi \Rightarrow \varphi$ [tiers exclus]
4. $\{\neg\forall x.\varphi\} \vdash_\Sigma \forall x.\neg\neg\varphi \Rightarrow \neg\neg\varphi$ [instantiation] $[\varphi/\neg\neg\varphi]$ $[t/x]$
5. $\{\neg\forall x.\varphi\} \vdash_\Sigma (\forall x\neg\neg\varphi \Rightarrow \neg\neg\varphi) \Rightarrow ((\neg\neg\varphi \Rightarrow \varphi) \Rightarrow (\forall x\neg\neg\varphi \Rightarrow \varphi))$ lemme à démontrer
6. $\{\neg\forall x.\varphi\} \vdash_\Sigma \forall x\neg\neg\varphi \Rightarrow \varphi$ modus ponens 2 fois
7. $\{\neg\forall x.\varphi\} \vdash_\Sigma \forall x.(\forall x\neg\neg\varphi \Rightarrow \varphi)$ généralisation
8. $\{\neg\forall x.\varphi\} \vdash_\Sigma \forall x.(\forall x\neg\neg\varphi \Rightarrow \varphi) \Rightarrow (\forall x\neg\neg\varphi \Rightarrow \forall x.\varphi)$ [distribution] $[\psi/\forall\neg\neg\varphi]$
9. $\{\neg\forall x.\varphi\} \vdash_\Sigma \forall x\neg\neg\varphi \Rightarrow \forall x.\varphi$ modus ponens
10. $\{\neg\forall x.\varphi\} \vdash_\Sigma \neg\forall x.\varphi$ hypothèse
11. $\{\neg\forall x.\varphi\} \vdash_\Sigma (\forall x\neg\neg\varphi \Rightarrow \forall x.\varphi) \Rightarrow (\neg\forall x.\varphi \Rightarrow \neg\forall x\neg\neg\varphi)$ lemme à démontrer
12. $\{\neg\forall x.\varphi\} \vdash_\Sigma \neg\forall x\neg\neg\varphi$ modus ponens 2 fois
13. $\{\neg\forall x.\varphi\} \vdash_\Sigma \exists x.\neg\varphi$ modus ponens

3.1.4 Théorème de complétude

Il existe deux sens du mot complétude en logique mathématique : la *complétude syntaxique* et la *complétude sémantique*. La complétude syntaxique s'intéresse au pouvoir de démonstration (via la relation $\vdash$) au sein d'une théorie Γ.

Définition 19 : Une théorie Γ sera alors *syntaxiquement complète* si elle est consistante et si de plus, l'on peut démontrer toute formule φ ou sa négation (c-à-d., $\Gamma \vdash \varphi$ ou $\Gamma \vdash \neg\varphi$).

L'intérêt d'une telle notion est qu'elle entraine la décidabilité de la théorie Γ. [14]
La complétude sémantique établit une équivalence entre la notion de dérivabilité et de validité. C'est

[14] En effet, l'ensemble des théorèmes ainsi que des non-théorèmes sont recursivement énumérables.

la notion de complétude qui vous a été donnée dans le cours sur les logiques générales, et celle qui nous interesse dans cette section.

Le résultat de complétude a d'abord été démontré au départ par K. Godël en 1930, puis généralisé par L. Henkin en 1949. Dans la littérature, nous trouvons principalement la preuve de complétude donnée Par L. Henkin. L'intéret de cette dernière sur celle établit par K. Godël porte principalement sur deux points :

1. elle considère des langages qui peuvent contenir une infinité de symboles primitifs.
2. elle peut-être étendue à des logiques d'ordre supérieur (logique du second ordre, théorie des types,...).

Nous allons donc prouver le résultat suivant :

si Γ *est une théorie consistante alors* Γ *a un modèle.*

La preuve est plus complexe que celle donnée pour la logique propositionnelle parce qu'elle passe nécessairement par la construction d'un modèle. Pour cela, nous avons besoin de quelques définitions et lemmes intermédiaires. Tout d'abord, donnons la définition qui va nous permettre de construire notre modèle.

Définition 20 : Étant donnée une signature Σ, on appelle un Σ-*axiome de Henkin* toute Σ-formule de la forme :

$$\exists x.\ \varphi \Rightarrow \varphi(x/c)$$

où c est une constante de la signature Σ.

On appelle Σ-*théorie de Henkin* toute Σ-théorie Γ qui contient tous les Σ-axiomes de Henkin.

La preuve de complétude selon L. Henkin se scinde alors en deux parties. Tout d'abord, nous montrons que toute Σ-théorie de Henkin Γ complète syntaxiquement, admet un modèle. Enfin, nous montrons que pour toute Σ-théorie consistante il existe une Σ'-théorie de Henkin complète syntaxiquement.

Lemme 21 : Si Γ est une Σ-théorie de Henkin complète syntaxiqement alors Γ admet un modèle.

Preuve : Notons $\mathcal{M}$ le Σ-modèle suivant :

- $M = T_\Sigma(\emptyset)$, [15]
- pour chaque symbole de la signature Σ, nous donnons la sémantique suivante :
  - pour chaque symbole de constante c de $\mathcal{F}$, nous avons : $c^{\mathcal{M}} = c$,
  - pour chaque symbole de fonction f^n de $\mathcal{F}$, nous posons $f^{\mathcal{M}} : M^n \to M$ par : $(t_1, \ldots, t_n) \mapsto f(t_1, \ldots, t_n)$,
  - pour chaque symbole de prédicat R^n de $\mathcal{R}$, nous posons :

$$R^{\mathcal{R}} = \{(t_1, \ldots, t_n) \mid \Gamma \vdash R(t_1, \ldots, t_n)\}$$

De part la définition de $\mathcal{M}$, pour toute formule atomique φ, on a : $\Gamma \vdash_\Sigma \varphi \Longleftrightarrow \mathcal{M} \models_\Sigma \varphi$. Montrons alors que cette équivalence peut-être étendue à toutes les Σ-formules.

La preuve se fait par induction sur la structure des Σ-formules. Les cas dont le connecteur principal est un connecteur propositionnel ne pose pas de problème particulier. Par conséquent, les seuls cas que nous allons mentionner ici sont ceux faisant intervenir les deux quantificateurs "$\forall$" et "$\exists$" comme symbole de tête.

[15] $T_\Sigma(\emptyset)$ dénote l'ensemble des termes avec variables dans l'ensemble vide. On appelle ce dernier, l'*ensemble des termes clos.*

1. Supposons que $\Gamma \vdash_\Sigma \forall x.\varphi$. Nous avons aussi pour tout terme t de $T_\Sigma(\emptyset)$: $\Gamma \vdash_\Sigma \forall x.\varphi \Rightarrow \varphi(x/t)$ (axiome 2 de la définition 16). Par modus ponens, nous avons alors : $\Gamma \vdash_\Sigma \varphi(x/t)$. Par hypothèse de récurrence, nous pouvons alors écrire : $\mathcal{M} \models_\Sigma \varphi(x/t)$ et ce pour tout terme t de $T_\Sigma(\emptyset)$. Il en découle alors directement : $\mathcal{M} \models_\Sigma \forall x.\varphi$.
   Montrons la réciproque par contraposée. Supposons alors que $\Gamma \not\vdash_\Sigma \forall x.\varphi$. Comme Γ est syntaxiquement complète, nous avons alors : $\Gamma \vdash_\Sigma \neg(\forall x.\varphi)$. Nous avons montré dans la section 3.1.3 que $\vdash_\Sigma \neg(\forall x.\varphi) \Leftrightarrow \exists x.\neg\varphi$, donc par monotonie, nous avons : $\Gamma \vdash_\Sigma \neg(\forall x.\varphi) \Rightarrow \exists x.\neg\varphi$. Par modus ponens, nous obtenons alors : $\Gamma \vdash_\Sigma \exists x.\neg\varphi$. Maintenant, nous savons que Γ est une théorie de Henkin. Nous avons alors : $\Gamma \vdash_\Sigma \exists x.\neg\varphi \Rightarrow \neg\varphi(x/c)$. D'où par modus ponens, nous pouvons écrire : $\Gamma \vdash_\Sigma \neg\varphi(x/c)$. Par hypothèse de récurrence, nous obtenons alors : $\mathcal{M} \models_\Sigma \neg\varphi(x/c)$ d'où nous pouvons directement conclure par : $\mathcal{M} \not\models_\Sigma \forall x.\varphi$.
2. `le connecteur d'existence est laiss'e en exercice.`

□

La seconde partie de la preuve repose sur une technique bien connue en théorie des modèles : la *méthode des diagrammes*. Succintement, elle consiste à enrichir une signature de départ en ajoutant une ensemble de constantes nouvelles en un nombre suffisant afin de pouvoir construire toutes les nouvelles formules utiles (dans notre cas, les axiomes de Henkin) pour résoudre un problème donné [16].

Lemme 22 : Soit Γ une Σ-théorie consistante. Il existe une signature Σ' contenant Σ (c-à-d., il existe un morphisme de signature d'inclusion entre Σ et Σ') et une Σ'-théorie de Henkin Γ' complète syntaxiquement qui contient Γ.

Preuve : Tout d'abord, on construit inductivement sur $\mathbb{N}$ une nouvelle théorie consistante Γ' contenant Γ sur une signature Σ'. Pour cela, on pose comme point départ, $\Gamma_0 = \Gamma$ et $\Sigma_0 = \Sigma$, et on définit Γ_{n+1} (resp. Σ_{n+1}) à partir de Γ_n (resp. Σ_n) de la façon suivante :

$$\Gamma_{n+1} = \Gamma_n \bigcup \{\exists x.\varphi \Rightarrow \varphi(x/c_\varphi) \mid \varphi \text{ est une } \Sigma_n\text{-formule}\}$$

$$\Sigma_{n+1} = \Sigma_n \bigcup \{c_\varphi \mid \varphi \text{ est une } \Sigma_n\text{-formule}\}$$

Maintenant, montrons que Γ_{n+1} est une Σ_{n+1}-théorie consistante. Ceci se fait par récurrence sur les entiers n. Par hypothèse de départ, nous avons bien entendu que Γ_0 est consistante. Pour Γ_{n+1}, raisonnons par l'absurde. Supposons alors que Γ_{n+1} est inconsitante. Cela veut dire qu'il existe un nouvel axiome de Henkin $\exists x.\varphi \Rightarrow \varphi(x/c_\varphi)$ n'apparaissant pas dans Γ_n tel que : $\Gamma_n \vdash_{\Sigma_n} \exists x.\varphi \wedge \neg\varphi(x/c_\varphi)$. Montrons alors que si $\Gamma_n \vdash_{\Sigma_n} \neg\varphi(x/c_\varphi)$ alors $\Gamma_n \vdash_{\Sigma_n} \forall x.\neg\varphi$. Si $\Gamma_n \vdash_{\Sigma_n} \neg\varphi(x/c_\varphi)$, nous avons une suite $(\varphi_1, \ldots, \varphi_m)$ dénotant une preuve de $\neg\varphi(x/c_\varphi)$. Substituons alors dans toutes les formules φ_i pour i compris entre 1 et m, la constante c_φ par une nouvelle variable fraiche z (c-à-d., n'apparaissant dans aucune des formules φ_i). Nous obtenons alors une nouvelle suite $(\psi_1, \ldots, \psi_m)$ dénotant une preuve de $\neg\varphi(x/z)$ dans Γ_n. Nous avons par la règle de généralisation : $\Gamma_n \vdash_{\Sigma_n} \forall z.\neg\varphi(x/z)$. Or, il est facile de montrer que $\vdash_{\Sigma_n} \forall z.\neg\varphi(x/z) \Rightarrow \forall x.\neg\varphi$ (le faire à titre d'exercice). De ceci, nous déduisons : $\Gamma_n \vdash_{\Sigma_n} \exists x.\varphi \wedge \forall x.\neg\varphi$ ce qui n'est pas possible puisque Γ_n est consistante par hypothèse de récurrence.

On pose alors : $\Gamma' = \bigcup_n \Gamma_n$ et $\Sigma' = \bigcup_n \Sigma_n$. Maintenant, il nous reste à trouver une Σ'-théorie Γ'' complète syntaxiquement. Pour cela, nous allons utiliser un résultat classique en théorie des ensembles :

[16] Cette méthode est classiquement utilisée pour démontrer entre autres les deux résultats fondamentaux de la théorie des modèles : les théorème de compacité et de Loweineim-Skolem.

le *lemme de Zorn*.
Considérons l'ensemble S de toutes les Σ'-théories consistantes contenant Γ'. Cet ensemble n'est pas vide car il contient au moins Γ' (nous avons montré ci-dessus que Γ' est elle-même consistante). Maintenant, considérons un sous-ensemble X de S totalement ordonné par inclusion (on appelle cela une chaine). Posons alors $\overline{\Gamma} = \bigcup_{\Gamma \in X} \Gamma$ et montrons que $\overline{\Gamma}$ est aussi consistante. Pour cela, nous devons considérer le lemme intermédiaire suivant :

Lemme 23 : Si Γ est une Σ-théorie dont toutes les parties finies sont consistantes, alors Γ est elle-même consistante.

Preuve : Découle directement du fait que la relation $\vdash_\Sigma$ vérifie la propriété de compacité. □

À partir du lemme ci-dessus, montrons que $\overline{\Gamma}$ est consistante. Pour cela, supposons le contraire. Il existe un sous-ensemble fini $\overline{\Gamma}'$ de $\overline{\Gamma}$ inconsistant. Par définition de l'ensemble $\overline{\Gamma}$, chaque formule φ de $\overline{\Gamma}$ appartient à une Σ'-théorie consistante Γ_φ de X. Mais, l'ensemble $\{\Gamma_\varphi \mid \varphi \in \overline{\Gamma}'\}$ admet un plus grand élément $\overline{\Gamma}_0$ appartenant à X. Maintenant, l'ensemble de formules $\overline{\Gamma}'$ est un sous-ensemble de $\overline{\Gamma}_0$. Donc, $\overline{\Gamma}_0$ est inconsistante ce qui contredit l'hypothèse de départ.
On a alors que la Σ'-théorie $\overline{\Gamma}$ est consistante. Par définition, elle est un majorant (au sens de l'inclusion) de l'ensemble X dans S. On peut appliquer le lemme de Zorn dont l'ennoncé est le suivant :

Lemme 24 : Étant donné un ensemble non vide S muni d'une relation d'ordre $\prec$, si tout sous-ensemble S' de S totalement ordonné par $\prec$ possède un élément maximal au sens de $\prec$ dans S, alors, S possède un élément maximal (c-à-d., plus grand que tous ses autres éléments toujours au sens de $\prec$) [17]

À partir de ce lemme, il existe une Σ'-théorie Γ'' maximale au sens de l'inclusion pour S.
Pour finir montrons alors que Γ'' est complète syntaxiquement, c'est-à-dire, pour toute Σ'-formule φ soit $\Gamma'' \vdash_{\Sigma'} \varphi$, ou $\Gamma'' \vdash_{\Sigma'} \neg\varphi$. Supposons alors une Σ'-formule φ telle que $\varphi \notin \Gamma''$. Par la propriété de Γ'' d'être maximale, $\Gamma'' \bigcup \{\varphi\}$ n'est pas consistante. On endéduit alors directement que : $\Gamma'' \vdash_{\Sigma'} \neg\varphi$.
□

Maintenant, nous avons tous les ingédients pour démontrer le théorème de complétude.

Théorème 25 : Tout Σ-théorie Γ consistante, admet un modèle.

Preuve : Il suffit d'appliquer le lemme 22 pour trouver une Σ'-théorie de Henkin consistante et complète syntaxiquement. Par le lemme 21, on sait alors qu'il existe un Σ'-modèle $\mathcal{M}$ pour cette théorie. Le Σ-modèle obtenu en élaguant tous les termes de M faisant intervenir des constantes n'apparaissant pas dans Σ (c-à-d., toutes les constantes utilisées pour les axiomes de Henkin) est alors un modèle de Γ. □

On sait aussi que le théorème de complétude peut s'énoncer sous cette forme (voir le cours sur la logique propositionnelle).

Théorème 26 : Si Γ est une Σ-théorie et si φ est une Σ-formule telle que $\Gamma \models \varphi$, alors $\Gamma \vdash \varphi$.

3.2 Le calcul des séquents

L'un des problèmes majeurs de la méthode de Frege-Hilbert (que l'on retrouve aussi dans la méthode de la déduction naturelle) réside dans la possibilité de considérer lors d'une preuve d'une

[17] Le lemme de Zorn est l'équivalent de l'axiome de choix vérifiée dans la théorie des ensembles de ZF.

formule donnée, des formules indépendantes (c-à-d., qui ne sont ni des tranformées, ni des parties de la formules à prouver). Ceci demande alors une connaissance approfondie de la formule à prouver pour savoir quelles formules il faut ajouter pour sa démonstration. Dans cette section, nous proposons un nouveau système d'inférences appelé *calcul des séquents*. À l'inverse du système de preuves précédent, le calcul des séquents permettra la preuve d'une formule en décomposant ou en transformant la formule à prouver sans ajouter aucun élément extérieur.

3.2.1 La notion de séquents

Le calcul des séquents prévéligie la notion de règles d'inférences et réduit le nombre d'axiomes logiques à son minimum.

Ce système de preuve manipule des phrases que l'on appelle communément des *séquents*.

Définition 27 : Un *séquent* sur une signature Σ est un couple (Γ, Δ) où Γ et Δ sont des ensembles de Σ-formules finis et non vides. On le note $\Gamma \mid\sim \Delta$.
Dans la suite, On notera simplement "Γ, φ" pour "$\Gamma \bigcup \{\varphi\}$".

Remarque 28 : La sémantique d'un séquent est analogue à une clause. La différence est que les formules de Γ et de Δ sont totalement arbitraires.
Le séquent "$\varphi_1, \ldots, \varphi_n \mid\sim \psi_1, \ldots, \psi_m$" est alors interprété comme la formule :

$$\varphi_1 \wedge \ldots \wedge \varphi_n \Rightarrow \psi_1 \wedge \ldots \wedge \psi_m$$

Le séquent $\Gamma, \varphi, \Delta, \Delta \mid\sim \Gamma', \varphi, \Delta'$ est donc toujours vrai.

3.3 Axiomes et règles

De la même façon que pour la méthode de Frege-Hilbert, nous entendons par axiome toute formule dont la valeur de vérité est vraie (plus communément appelée une *tautologie*). De part la remarque ci-dessus, nous appellerons alors *axiome* tous les séquents de la forme : $\varphi \mid\sim \varphi$ où φ est une Σ-formule quelconque.

3.4 Les règles de déduction

Ces règles se partagent en trois ensembles distincts. Le premier ensemble contient toutes les règles dites *structurelles*. Elles caractérisent la propriété pour Γ et Δ dans un séquent "$\Gamma \mid\sim \Delta$", d'être des ensembles. L'ensemble des règles structurelles contient en plus, une règle très importante pour cette méthode de preuve : la *coupure*. Intuitivement, nous pouvons la considérer comme une généralisation de la règle du modus ponens de la méthode de Frege-Hilbert.
Le second ensemble contient toutes les règles logiques qui restranscrive syntaxiquement les propriétés sémantiques liées aux connecteurs propositionnels. Enfin, le dernier ensemble contient les règles logiques attachées aux quantificateurs.
Le deux premiers ensembles définissent le calcul des séquents pour la logique propositionnelle. Avec les règles associées aux quantificateurs, on obtient le calcul des séquents pour la logique des prédicats du 1er ordre.

Définition 29 : Soit Σ une signature dont $\mathcal{V}$ est l'ensemble des variables. Soient Γ, Δ, Λ et Π quatre ensembles de Σ-formules. Soient φ et ψ deux Σ-formules. Soient x et y deux variables de $\mathcal{V}$. Soient t, $t_1$et t_2 des termes de $T_\Sigma(\mathcal{V})$. On considère l'ensemble des règles d'inférence suivant :

Règles structurelles

Affaiblissement :

gauche $\dfrac{\Gamma \vdash \Delta}{\varphi, \Gamma \vdash \Delta}$ droite $\Gamma \vdash \Delta \Gamma \vdash \Delta, \varphi$

Diminution :

gauche $\dfrac{\varphi, \varphi, \Gamma \vdash \Delta}{\varphi, \Gamma \vdash \Delta}$ droite $\dfrac{\Gamma \vdash \Delta, \varphi, \varphi}{\Gamma \vdash \Delta, \varphi}$

Échange :

gauche $\dfrac{\Gamma, \varphi, \psi, \Lambda \vdash \Delta}{\varphi, \Gamma, \psi, \varphi, \Lambda \vdash \Delta}$ droite $\dfrac{\Gamma \vdash \Delta, \varphi, \psi, \Lambda}{\Gamma \vdash \Delta, \psi, \varphi, \Lambda}$

Coupure :

$$\frac{\Gamma \vdash \Delta, \varphi \quad \varphi, \Lambda \vdash \Pi}{\Gamma, \Lambda \vdash \Delta, \Pi}$$

Règles propositionnelles

$\dfrac{\Gamma, \varphi, \psi \vdash \Delta}{\Gamma, \varphi \wedge \psi \vdash \Delta}(\wedge \vdash)$ $\dfrac{\Gamma \vdash \Delta, \varphi \quad \Gamma \vdash \Delta, \psi}{\Gamma \vdash \Delta, \varphi \wedge \psi}(\vdash \wedge)$

$\dfrac{\Gamma, \varphi \vdash \Delta \quad \Gamma, \psi \vdash \Delta}{\Gamma, \varphi \vee \psi \vdash \Delta}(\vee \vdash)$ $\dfrac{\Gamma, \vdash \Delta, \varphi, \psi}{\Gamma \vdash \Delta, \varphi \vee \psi}(\vdash \vee)$

$\dfrac{\Gamma \vdash \Delta, \varphi}{\Gamma, \neg\varphi \vdash \Delta}(\neg \vdash)$ $\dfrac{\Gamma, \varphi \vdash \Delta}{\Gamma \vdash \Delta, \neg\varphi}(\vdash \neg)$

$\dfrac{\Gamma \vdash \Delta, \varphi \quad \Gamma, \psi \vdash \Delta}{\Gamma, (\varphi \Rightarrow \psi) \vdash \Delta}(\Rightarrow \vdash)$ $\dfrac{\Gamma, \varphi \vdash \Delta, \psi}{\Gamma \vdash \Delta, (\varphi \Rightarrow \psi)}(\vdash \Rightarrow)$

Règles pour les quantificateurs

$\dfrac{\Gamma, \varphi(x/t) \vdash \Delta}{\Gamma, \forall x.\varphi \vdash \Delta}(\forall \vdash)$ $\dfrac{\Gamma \vdash \Delta, \varphi(x/y)}{\Gamma \vdash \Delta, \forall x.\varphi}(\vdash \forall)$
la variable y n'est pas libre dans $\Gamma \cup \Delta$

$\dfrac{\Gamma, \varphi(x/y) \vdash \Delta}{\Gamma, \exists x.\varphi \vdash \Delta}(\exists \vdash)$
la variable y n'est pas libre dans $\Gamma \cup \Delta$

$\dfrac{\Gamma \vdash \Delta, \varphi(x/t)}{\Gamma \vdash \Delta, \exists x.\varphi}(\vdash \exists)$

Ces règles sont appelées *introduction* quand elles sont utilisées de haut en bas, et *élimination* quand elles sont utilisées de bas en haut.

Pour prouver un théorème avec le calcul des séquents, on conserve la représentation en arbre de preuve. Ceci vient du fait même de la définition du calcul des séquents qui cherche à montrer que la formule à proubver ne contient comme sous-formule (à transformation près) que des tautologies, et donc comme sous-formules de bases des axiomes de la forme : $\varphi \vdash \varphi$.

3.4.1 En tant que systèmes formels

Pour chaque signature $\Sigma \in Sig$, le triplet $C_\Sigma = (A_\Sigma, \mathcal{F}_\Sigma, \mathcal{R}_\Sigma)$ définit à partir du calcul des séquents diffère de celui déduit à partir de la méthode de Frege-Hilbert sur l'alphabet A_Σ que l'on considère et donc aussi les formules manipulées par ce dernier. En effet, A_Σ contient un élément supplémentaire : $\vdash$. Nous obtenons alors :

– A_Σ est l'alphabet $\mathcal{F} \cup \mathcal{R} \cup \mathcal{V} \cup \{\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow, \forall, \exists\} \bigcup \{\vdash\}$

– $\mathcal{F}_\Sigma$ contient alors alors l'ensemble de tous les séquents bien-formés que l'on peut construire à partir de l'aplhabet A_Σ.
– enfin, $\mathcal{R}_\Sigma$ est l'ensemble des règles obtenues à partir des schémas de règles donnés précédemment.

On définit alors la relation $\vdash_\Sigma$ par :

$$\forall \Gamma \subseteq For(\Sigma), \forall \varphi \in For(\Sigma),\ \Gamma \vdash_\Sigma \varphi \Longleftrightarrow \vdash_{C_\Sigma} \Gamma \mid\sim \varphi$$

De la même façon que pour la méthode de Frege-Hilbert, il reste à vérifier les propriétés de $\vdash$-translation et de correction. La première est simple à démontrer et est donc laissée en exercice au lecteur.

Théorème 30 : Tout séquent $\Gamma \mid\sim \Delta$ dérivable à partir du calcul des séquents est valide.

Preuve : Comme à son habitude, cette propriété se démontre par récurrence sur la longueur de la preuve. Pour cela, Nous supposons un modèle $\mathcal{M}$ et un séquent $\psi_1, \ldots, \psi_n \mid\sim \varphi_1, \ldots, \varphi_m$ quelconques. Le cas de base est celui où le séquent $\Gamma \mid\sim \Delta$ a été obtenu en une étape. C'est donc une axiome de la forme $\varphi \mid\sim \varphi$ dont l'interpréttion est $\varphi \Rightarrow \varphi$ dont il a déjàété montré quye c'est une tautologie. Maintenant, nous devons montrer que la validité est préservée au travaers des règles d'inférence. Vu le nombre des règles du calcul des séquents, nous n'allons pas prouver cette proriété pour cahcune de ces règles mais simplement pour la règles de coupure et celles associées au quantificateur universelle.

coupure
Nous supposons que le séquent $\psi_1, \ldots, \psi_n \mid\sim \varphi_1, \ldots, \varphi_m$ a été obtenu par application de la règle de coupure. Nous avons alors deux séquents $\psi_1, \ldots, \psi_k \mid\sim \varphi_1, \ldots, \varphi_l, \varphi$ et $\varphi, \psi_{k+1}, \ldots, \psi_n \mid\sim \varphi_{l+1}, \ldots, \varphi_m$ avec $k \leq n$ et $l \leq m$ dans l'arbre de preuve du séquent $\psi_1, \ldots, \psi_n \mid\sim \varphi_1, \ldots, \varphi_m$. Par hypothèse de récurrence, ces deux séquents sont validés par le modèle $\mathcal{M}$, c'est-à-dire : $\mathcal{M} \models_\Sigma \psi_1 \wedge \ldots \wedge \psi_k \Rightarrow \varphi_1 \vee \ldots \vee \varphi_l \vee \varphi$ et $\mathcal{M} \models_\Sigma \varphi \wedge \psi_{k+1} \wedge \ldots \wedge \psi_n \Rightarrow \varphi_{l+1} \vee \ldots \vee \varphi_m$. Par définition, ceci veut dire que pour toute interprétation $\nu : \mathcal{V} \to M$, nous avons : $\mathcal{M} \models^\nu_\Sigma \psi_1 \wedge \ldots \wedge \psi_k \Rightarrow \varphi_1 \vee \ldots \vee \varphi_l \vee \varphi$ et $\mathcal{M} \models^\nu_\Sigma \varphi \wedge \psi_{k+1} \wedge \ldots \wedge \psi_n \Rightarrow \varphi_{l+1} \vee \ldots \vee \varphi_m$. Soit $\nu^+ : \mathcal{V} \to M$ une interprétation telle que $\mathcal{M} \models^{\nu^+}_\Sigma \psi_1 \wedge \ldots \wedge \psi_k \wedge \psi_{k+1} \wedge \ldots \wedge \psi_n$. Nous avons alors deux cas à considérer :

1. $\mathcal{M} \models^{\nu^+}_\Sigma \varphi$. Dans ce cas-là, nous avons aussi $\mathcal{M} \models^{\nu^+}_\Sigma \varphi \wedge \psi_{k+1} \wedge \ldots \wedge \psi_n$, et donc $\mathcal{M} \models^{\nu^+}_\Sigma \varphi_{l+1} \vee \ldots \vee \varphi_m$. Nous avons alors $\mathcal{M} \models^{\nu^+}_\Sigma \varphi_1 \vee \ldots \vee \varphi_l \vee \varphi_{l+1} \vee \ldots \vee \varphi_m$.
2. $\mathcal{M} \not\models^{\nu^+}_\Sigma \varphi$. Dans ce cas-là, nous avons $\mathcal{M} \models^{\nu^+}_\Sigma \varphi_1 \vee \ldots \vee \varphi_l$, d'où nous déduisons directement $\mathcal{M} \models^{\nu^+}_\Sigma \varphi_1 \vee \ldots \vee \varphi_l \vee \varphi_{l+1} \vee \ldots \vee \varphi_m$.

quantificateur universel
$\forall \mid\sim$ Nous supposons que le séquent $\psi_1, \ldots, \psi_n \mid\sim \varphi_1, \ldots, \varphi_m$ a été obtenu par application de la règle $\forall \mid\sim$. Nous avons alors un séquent $\psi_1, \ldots, \psi_{n-1}, \varphi(x/t) \mid\sim \varphi_1, \ldots, \varphi_m$ dans l'arbre de preuve du séquent $\psi_1, \ldots, \psi_{n-1}, \forall x.\varphi \mid\sim \varphi_1, \ldots, \varphi_m$. Par hypothèse de récurrence, nous pouvons écrire $\mathcal{M} \models_\Sigma \psi_1 \wedge \ldots \wedge \psi_{n-1} \wedge \varphi(x/t) \Rightarrow \varphi_1 \vee \ldots \vee \varphi_m$. Par définition, ceci veut dire que pour toute interprétation $\nu : \mathcal{V} \to M$, $\mathcal{M} \models^\nu_\Sigma \psi_1 \wedge \ldots \wedge \psi_{n-1} \wedge \varphi(x/t) \Rightarrow \varphi_1 \vee \ldots \vee \varphi_m$. Supposons alors que pour une interprétation $\nu^+ : \mathcal{V} \to M$ donnée, $\mathcal{M} \models^{\nu^+}_\Sigma \psi_1 \wedge \ldots \wedge \psi_{n-1} \wedge \forall x, \varphi$. Nous avons déjà montré pour la correction du calcul de Hilbert-Frege, que la formule $\forall x.\varphi \Rightarrow \varphi(x/t)$ est une tautologie. Nous avons alors $\mathcal{M} \models^{\nu^+}_\Sigma \forall x.\varphi \Rightarrow \varphi(x/t)$, ce qui par définition nous donne, $\mathcal{M} \models^{\nu^+}_\Sigma \varphi(x/t)$. De ceci nous obtenons alors directement, $\mathcal{M} \models^{\nu^+}_\Sigma \psi_1 \wedge \ldots \wedge \psi_{n-1} \wedge \varphi(x/t)$, et donc, $\mathcal{M} \models^{\nu^+}_\Sigma \varphi_1 \wedge \ldots \wedge \varphi_m$.
$\mid\sim \forall$ Nous supposons que le séquent $\psi_1, \ldots, \psi_n \mid\sim \varphi_1, \ldots, \varphi_m$ a été obtenu par application de la règle $\mid\sim \forall$. Nous avons alors un séquent $\psi_1, \ldots, \psi_n \mid\sim \varphi_1, \ldots, \varphi_{m-1}, \varphi(x/y)$ dans l'arbre de

preuve du séquent $\psi_1, \ldots, \psi_n, \forall x.\varphi \vdash\sim \varphi_1, \ldots, \varphi_m$. Par hypothèse de récurrence, nous pouvons écrire $\mathcal{M} \models_\Sigma \psi_1 \wedge \ldots \wedge \psi_n \vdash\sim \varphi_1 \vee \ldots \vee \varphi_{m-1} \vee \varphi(x/y)$. Supposons que l'ensemble des variables libres de $\psi_1 \wedge \ldots \wedge \psi_n$ et $\varphi_1 \wedge \ldots \wedge \varphi_m$ est $\{x_1, \ldots, x_k\}$. L'ensemble des variables libres du séquent $\psi_1, \ldots, \psi_n \vdash\sim \varphi_1, \ldots, \varphi_{m-1}, \varphi(x/y)$ est alors $\{x_1, \ldots, x_k, y\}$. Nous pouvons donc écrire par la règle de généralisation du calcul de Hilbert-Frege $\mathcal{M} \models_\Sigma \forall x_1. \ldots . \forall x_k. \forall y. (\neg\psi_1 \vee \ldots \vee \neg\psi_n \vee \varphi_1 \vee \ldots \vee \varphi_{m-1} \vee \varphi(x/y))$. Comme la variable y n'est pas libre pour $\psi_1, \ldots, \psi_n$ et $\varphi_1, \ldots, \varphi_{m-1}$, nous pouvons écrire $\mathcal{M} \models_\Sigma \forall x_1. \ldots . \forall x_k. (\neg\psi_1 \vee \ldots \vee \neg\psi_n \vee \varphi_1 \vee \ldots \vee \varphi_{m-1} \vee \forall y. \varphi(x/y))$. La suite de la preuve respose sur le lemme simple suivant :

Lemme 31 : Étant données une formule φ et une varibale y qui n'apparait pas libre dans φ, nous avons : $\vdash_\Sigma \forall y. \varphi(x/y) \Rightarrow \forall x. \varphi$

Preuve : Par l'axiome d'instantiation, nous avons : $\vdash_\Sigma \forall y. \varphi(x/y) \Rightarrow \forall x. \varphi(x/y)(y/x)$. Mais par définition, dans la formule $\varphi(y/x)$, la variable x n'apparait plus libre. Donc, $\varphi(x/y)(y/x)$ est φ. □

Par ce lemme, nous obtenons alors directement : $\mathcal{M} \models_\Sigma \forall x_1. \ldots . \forall x_k. (\neg\psi_1 \vee \ldots \vee \neg\psi_n \vee \varphi_1 \vee \ldots \vee \varphi_{m-1} \vee \forall x. \varphi)$

□

3.4.2 Exemple de preuve

TBFL (To Be Filled Later).

3.4.3 Complétude

La complétude du calcul des séquents se démontre en deux étapes principales. La première étape consiste à simuler le calcul de Hilbert-Frege dans le calcul des séquents. Nous montrons alors par récurrence sur la longueur de la preuve que pour toute formule φ que l'on peut obtenir à partir du calcul de Hilbert-Frege, le séquent $\vdash\sim \varphi$ peut être établie par le calcul des séquents. À partir de ce résultat, nous obtenons le corollaire suivant :

Corollaire 32 : Pour toute formule φ, nous avons : $\models_\Sigma \varphi$ implique $\vdash\sim \varphi$ admet une preuve à partir du calcul des séquents.

Preuve : En effet, par le théorème de complétude, nous avons une preuve de φ par le calcul de Hilbert-Frege qui par le résultat explicité ci-dessus nous donne une preuve du séquent $\vdash\sim \varphi$ par le calcul des séquents. □

Mais ceci ne suffit pas à montrer la complétude du calcul des séquents. En effet, ici nous avons simplement ce résultat pour des séquents dont la forme est : $\vdash\sim \varphi$. Il nous reste alors à le montrer pour des séquents de forme plus générale, c'est-à-dire : $\Gamma \vdash\sim \Delta$. Pour cela, nous devons prouver le lemme intermédiaire suivant :

Lemme 33 : Pour $n > 0$ et $m > 0$, les séquents suivants sont prouvables :

1. $\varphi_1, \ldots, \varphi_n \vdash\sim \varphi_1 \wedge \ldots \wedge \varphi_n$
2. $\psi_1 \vee \ldots \vee \psi_m \vdash\sim \psi_1, \ldots, \psi_m$

Preuve : Par récurrence sur les entiers n et m (laissée en exercice). □

Nous pouvons maintenant montrer le théorème de complétude pour le calcul des séquents dont l'énoncé est le suivant :

Théorème 34 : Tout séquent $\varphi_1, \ldots, \varphi_n \mid\sim \psi_1, \ldots, \psi_m$ valide, admet une preuve à partir du calcul des séquents.

Preuve : $\varphi_1, \ldots, \varphi_n \mid\sim \psi_1, \ldots, \psi_m$ est un séquent valide, donc $\varphi_1 \wedge \ldots \wedge \varphi_n \Rightarrow \psi_1 \vee \ldots \vee \psi_m$ est une formule valide. Par le corollaire ci-dessus, le séquent $\mid\sim \varphi_1 \wedge \ldots \wedge \varphi_n \Rightarrow \psi_1 \vee \ldots \vee \psi_m$ admet une preuve à partir du calcul des séquents. Par la règle ($\mid\sim\Rightarrow$), le séquent $\varphi_1 \wedge \ldots \wedge \varphi_n \mid\sim \psi_1 \vee \ldots \vee \psi_m$ admet aussi une preuve à partir du calcul des séquents. En appliquant une fois, le lemme ci-dessus et la règle de coupure, le séquent $\varphi_1, \ldots, \varphi_n \mid\sim \psi_1 \vee \ldots \vee \psi_m$ devient prouvable. Il suffit alors d'appliquer encore une fois le lemme ci-dessus pour obtenir que le séquent $\varphi_1, \ldots, \varphi_n \mid\sim \psi_1, \ldots, \psi_m$ soit aussi prouvable à partir du calcul des séquents. □

[Fin cours 7]

Chapitre 5 : Modèles finiment engendrés et preuves par induction structurelle

1 Considérations générales

TBFL

Adaptation des logiques "classiques" pour les de'dier a' des domaines particuliers (variation de la the'orie des ensembles pour B, logiques e'quationnelles pour les types abstraits alge'briques, clauses de Horn pour PROLOG*, logiques temporelles, etc.). C'est ce qui motive des logiques comme celle de ce chapitre et celles du chapitre suivant. On va se focaliser dans ce chapitre sur la logique des pre'dicats finiment engendrée.*

2 Motivation

Commencer par considerations sur l'encapsulation, comment cela se traduit dans les modèles, ce que l'on doit s'autoriser en plus au niveau des preuves, en quoi cela généralise la re'currence classique avec 0 *et succ, etc.*

En informatique, lorsque nous utilisons la logique du 1er-ordre (ou une extension de cette dernière comme les logiques typées) pour spécifier le comportement d'un système (qu'il soit logiciel ou matériel), nous avons souvent en "tête" un modèle canonique, ou plus généralement, les modèles dont chaque valeur représente le résultat d'un calcul dénoté par un terme. Cependant, l'ensemble des modèles associé à une théorie ne contient pas que des modèles ayant cette propriété. Par exemple, si nous prenons la théorie associée à l'arithmétique usuelle de Peano donnée en section 2.3 à laquelle nous avons retirée au préalable le schéma d'axiome relatif à l'induction (bien entendu), l'ensemble des modèles de cette théorie contient entre autres, les modèles suivants :

$$< I\!N, 0_{I\!N}, succ_{I\!N}, +_{I\!N}, =_{I\!N} >$$

et

$$< Z\!\!Z/nZ\!\!Z, 0_{I\!N}, succ_{I\!N} mod\ n, +_{I\!N} mod\ n, \equiv mod\ n >$$

Mais, il contient aussi le modèle :

$$M = IR \quad \begin{array}{ll} succ^{\mathcal{M}} : M \to M & +^{\mathcal{M}} : M \times M \to M \\ m \mapsto (m +_{IR} 1) & (m, p) \mapsto (m +_{IR} p) \end{array} \quad =_{IR}$$

Or, ce dernier a des valeurs qui ne sont pas le résultat d'un calcul dénoté par des termes de la signature Σ_{Peano}.
En logique du 1er ordre, quant on réduit la classe des modèles, on augmente l'ensemble des formules

valides pour cette sous-classe. On peut alors se poser la question de savoir qu'elles sont les nouvelles formules validées pour la sous-classe des modèles dont le domaine résulte du calcul des termes. Intuitivement, on peut raisonnablement penser que ces nouvelles propriétés portent sur les termes. Or, les termes sont inductivement générés à partir des symboles de la signature en respectant l'arité de ces derniers. Donc, l'ensemble de ces nouvelles propriétés contient au moins toutes celles que l'on peut prouver par induction sur le nombre d'occurence des fonctions dans un terme, appelée plus communément *induction structurelle*.

Dans ce chapitre, nous présentons la sous-logique générale des prédicats du premier ordre dont la classe des modèles est réduit à ceux dont le domaine est calculable à partir des termes pour une signature donnée, appelés *modèles finiment engendrés*. Bien entendu, dans cette logique générale, nous trouverons une nouvelle règle d'inférence caractérisant syntaxiquement ce nouveau mode de raisonnement.

Dans la suite, nous ne présenterons que ce qui change par rapport aux différentes logiques générales des prédicats du 1er ordre, c'est-à-dire, les concepts de *signature* et de *modèle*, ainsi que la nouvelle règle d'inférence d'*induction structurelle*.

3 Institution des prédicats du 1er ordre finiment engendrés

3.1 Signatures

Une signature pour cette institution, est une signature du 1er ordre, comme définie dans la définition 1, à ceci près que l'on considère parmi l'ensemble des symboles de fonctions, un sous-ensemble, appelé *ensemble des constructeurs*. C'est sur les termes générés à partir de ce dernier que l'on fera de l'induction structurelle.

Définition 1 : Une *signature avec constructeurs* est une signature $\Sigma = (\mathcal{F}, \mathcal{R}, \mathcal{V})$ munie d'un sous-ensemble $\mathcal{C}$ de $\mathcal{F}$, appelé l'*ensemble des constructeurs*.
Dans la suite, on la notera par le quadruplet $\Sigma = (\mathcal{C}, \mathcal{F}, \mathcal{R}, \mathcal{V})$.
À partir d'une signature avec constructeurs $\Sigma = (\mathcal{C}, \mathcal{F}, \mathcal{R}, \mathcal{V})$, on définit la signature du 1er ordre $\Omega = (\mathcal{F}, \emptyset, \mathcal{V})$.

Exemple 2 : À partir de la signature Σ_{Peano} définie en section 2.1, on peut donner l'ensemble des constructeurs suivants : $\mathcal{C}_{Peano} = \{0^0, succ^1\}$.

L'intérêt de considérer un ensemble de constructeurs vient du fait qu'en informatique, nous spécifions un système d'information dans le but de raisonner dessus mais aussi de définir un prototype de ce dernier par raffinements successifs (c-à-d., ajout d'informations supplémentaires au moyen de nouvelles opérations et de nouvelles formules). Un prototype d'un système est alors une théorie dont on peut "exécuter" les formules par réécriture de termes. Ce dernier est un programme exprimé dans un langage logique, et comme tout programme, ce dernier doit posséder la propriété de terminaison, c'est-à-dire, que toute succession de réécriture de termes aboutit à un terme canonique (que l'on ne peut plus réécrire). L'ensemble de ces termes canoniques sont tous ceux que l'on générer à partir de l'ensemble des constructeurs (l'ensemble $T_\Omega(\mathcal{V})$).

Remarque 3 : Dans la suite, on dira simplement signature au lieu de signature avec constructeurs sauf quand ceci amènera à ambiguïté.

Définition 4 : Soient $\Sigma_1 = (\mathcal{C}_1, \mathcal{F}_1, \mathcal{R}_1, \mathcal{V}_1)$ et $\Sigma_2 = (\mathcal{C}_2, \mathcal{F}_2, \mathcal{R}_2, \mathcal{V}_2)$ deux signatures. Un *morphisme*

de signatures avec constructeurs $\eta : \Sigma_1 \rightarrow \Sigma_2$ est un morphisme de signatures avec comme contrainte supplémentaire : $\forall f \in \mathcal{C}_1,\ \sigma(f) \in \mathcal{C}_2$.

3.2 Modèles finiment engendrés

Pour une signature Σ, un Σ-modèle pour cette institution sera donc un Σ-modèle du 1er ordre avec la particularité que chaque valeur du domaine est le résultat de l'évaluation d'au moins un Ω-terme clos (c-à-d., un terme de $T_\Omega(\emptyset)$).

Notation 5 : Étant donnée une signature du 1er ordre $\Sigma = \mathcal{F}, \mathcal{R}, \mathcal{V})$ et un Σ-modèle $\mathcal{M}$, on notera $\_^\natural$ l'application définie de $T_\Sigma(\emptyset)$ dans M.
(Par définition cette application est unique).

Définition 6 : Étant donnée une signature $\Sigma = (\mathcal{C}, \mathcal{F}, \mathcal{R}, \mathcal{V})$, un Σ-modèle $\mathcal{M}$ est dit *finiment engendré* si et seulement si pour tout élément $m \in M$, il existe un terme $t \in T_\Omega(\emptyset)$ tel que $t^\natural = m$ (en d'autres mots, $\_^\natural : T_\Sigma(\emptyset) \rightarrow M$ réduite au sous-ensemble $T_\Omega(\emptyset)$ est surjective).
On note $Gen(\Sigma)$ la sous-classe de $Mod(\Sigma)$ des Σ-modèles finiment engendrés.

Bien entendu, nous devons aussi revoir la définition de l'application $\mathcal{U}_\sigma : Gen(\Sigma_2) \rightarrow Gen(\Sigma_1)$ pour un morphisme de signature $\sigma : \Sigma_1 \rightarrow \Sigma_2$. En effet, pour tout modèle $\mathcal{M}$ de $Mod(\Sigma_2)$, nous devons élaguer toutes les valeurs résultantes de l'évaluations d'une terme de $T_{\Omega_2}(\emptyset)$ mais n'appartenant pas à l'ensemble $T_{\Omega_1}(\emptyset)$.

Définition 7 : Soit $\sigma : \Sigma_1 \rightarrow \Sigma_2$ un morphisme de signatures. L'application $\mathcal{U}_\sigma : Gen(\Sigma_2) \rightarrow Gen(\Sigma_1)$ est défini par : pour tout Σ_2-modèle finiment engendré $\mathcal{M}$, $\mathcal{U}_\sigma(\mathcal{M})$ est le Σ_1-modèle finiment engendré où :
– l'ensemble sous-jacent B est le sous-ensemble de M défini par :

$$B = \{b \mid \exists t \in T_{\Omega_1}(\emptyset),\ t^\natural = b\}$$

– pour chaque fonction $f^n \in \mathcal{F}_1$, $f^{\mathcal{B}} : B^n \rightarrow B$ est l'application définie Par :

$$\forall (b_1, \ldots, b_n) \in B^n,\ f^{\mathcal{B}}(b_1, \ldots, b_n) = \sigma(f)\mathcal{M}(b_1, \ldots, b_n)$$

– pour chaque prédicat $R^n \in \mathcal{R}_1$, $R^{\mathcal{B}}$ est la relation définie sur B^n par :

$$R^{\mathcal{B}} = \{(b_1, \ldots, b_n) \mid (b_1, \ldots, b_n) \in B^n \wedge (b_1, \ldots, b_n) \in \sigma(R)^{\mathcal{M}}\}$$

Théorème 8 : Soit $\sigma : \Sigma_1 \rightarrow \Sigma_2$ un morphisme de signatures. Nous avons alors pour tout Σ_2-modèle $\mathcal{M}$ et toute Σ_1-formule φ, la propriété suivante :

$$\mathcal{M} \models_{\Sigma_2} \overline{\sigma}(\varphi) \Longrightarrow \mathcal{U}_\sigma(\mathcal{M}) \models_{\Sigma_1} \varphi$$

Preuve : Les étapes de preuve sont identiques à celles du théorème 14 pour le même sens de l'implication. □

Remarquons que l'implication réciproque de celle donnée par le théorème 14 n'est pas vraie en général, comme l'indique le contre-exemple suivant :

Soient les signatures $\Sigma_1 = (\{c_1^0\}, \{c_1^0\}, \{p^1\}, \{x\})$ et $\Sigma_2 = (\{c_1^0, c_2^0\}, \{c_1^0, c_2^0\}, \{p^1\}, \{x\})$.

Soit $\mathcal{M}$ le Σ_2-modèle défini par :

$$M = \{c_1, c_2\},\ c_1^{\mathcal{M}} = c_1,\ c_2^{\mathcal{M}} = c_2,\ p^{\mathcal{M}} = \{c_1\}$$

Rappelons que l'ensemble $T_{\Omega_2}(\emptyset) = \{c_1, c_2\}$.

Par définition, le Σ_1-modèle $\mathcal{U}_\sigma(\mathcal{M})$ via le morphisme de signature d'inclusion $\sigma : \Sigma_1 \to \Sigma_2$ naturel est donc défini par :

$$\mathcal{U}_\sigma(M) = \{c_1\},\ c_1^{\mathcal{U}_\sigma(\mathcal{M})} = c_1,\ p^{\mathcal{U}_\sigma(\mathcal{M})} = \{c_1\}$$

Il vient : $p^{\mathcal{U}_\sigma(\mathcal{M})} = T_{\Omega_1}(\emptyset)$. Par conséquent, $\mathcal{U}_\sigma(\mathcal{M})$ valide la formule $\forall x.p(x)$ mais pas $\mathcal{M}$ (la constante c_2 n'appartient pas au domaine de définition du prédicat p dans $\mathcal{M}$).

4 Preuves par induction structurelle

Maintenant, nous devons traduire syntaxiquement le raisonnement supplémentaire d'induction. Succinctement, ce raisonnement établit une récurrence sur le nombre d'occurence de symboles de fonction apparaissant dans un terme clos. Comme tout principe de récurrence, il se concrétise en supposant qu'une certaine propriété est vérifiée sur tous les termes (et donc les valeurs dénotées par ces termes pour un modèle donné) dont la taille est inférieure à un certain entier naturel, et l'établit pour les termes de taille juste supérieur, c'est-à-dire, ceux auxquels on a ajouté un symbole de fonction en tête. Les termes sur lesquels on caractérise l'hypothèse de récurrence distingue alors des valeurs particulière pour un modèle donné. Ils agissent comme des constantes nouvelles dont la sémantique sera ces valeurs qu'ils dénotent dans les modèles. Traduire un tel raisonnement nous impose alors de ne pas travailler à signature constante (dû à l'ajout de nouvelle constante). C'est pourquoi, le raisonnement par induction n'est pas une règle selon le sens donné par la définition de ce qu'est un système formel mais plutôt comme une méta-règle mettant en jeu toutes les relations d'inférence $\vdash_\Sigma$ de $\vdash$ pour une logique générale donnée.

Nous supposons alors donnée une logique générale $(Sign, For, Mod, Mod, \models, \vdash)$ où :
- $Sign$ est l'ensemble des signatures avec constructeurs muni des moprhismes entre ces signatures comme défini dans les définitions 1 et 4.
- For est définie comme pour la logique du 1er ordre.
- Mod est l'application qui à chaque signature avec constructeurs Σ associe la classe $Gen(\Sigma)$.
- $\models$ est définie comme pour la logique du 1er ordre.
- $\vdash$ est une famille de relation défini par un système formel quelconque de la logique du 1er ordre (calcul de Hilbert-Frege, déduction naturelle, calcul des séquents, tableaux sémantiques, etc.).

La règle d'induction se traduit alors par :

$$\frac{\{\Gamma, \varphi(x/x_1), \ldots, \varphi(x/x_n) \vdash_{\Sigma \bigcup \{x_1^0, \ldots, x_n^0\}} \varphi(x/f(x_1, \ldots, x_n))\}_{f^n \in \mathcal{C}}}{\Gamma \vdash_\Sigma \forall x.\varphi}$$

Les variables x_i ne sont pas libres pour Γ et φ.

pour $\Sigma = (\mathcal{F}, \mathcal{C}, \mathcal{R}, \mathcal{V})$:

$$\Sigma \bigcup \{x_1^0, \ldots, x_n^0\} = (\mathcal{F} \bigcup \{x_1^0, \ldots, x_n^0\}, \mathcal{C} \bigcup \{x_1^0, \ldots, x_n^0\}, \mathcal{R}, \mathcal{V} \setminus \{x_1, \ldots, x_n\})$$

Exemple 9 : À partir de la théorie de Peano, on peut démontrer grâce à la règle d'induction structurelle le fait que 0 est aussi neutre à gauche (par les axiomes, il est simplement donné neutre à droite) exprimé par la formule suivante : $\forall x.0 + x = x$. Pour cela, nous devons alors démontrer les deux formules suivantes :

1. **Cas de base :** $\Gamma_{Peano} \vdash_{\Sigma_{Peano}} 0 + 0 = 0$
2. **Cas général :** $\Gamma_{Peano} \bigcup \{0 + n = n\} \vdash_{\Sigma_{Peano} \bigcup \{n^0\}} 0 + succ(n) = succ(n)$

1. – $\Gamma_{Peano} \vdash_{\Sigma_{Peano}} \forall x.(x + 0) = x$ [introduction d'axiome]
   – $\Gamma_{Peano} \vdash_{\Sigma_{Peano}} 0 + 0 = 0$ [instantiation]
2. – $\Gamma_{Peano} \bigcup \{0 + n = n\} \vdash_{\Sigma_{Peano} \bigcup \{n^0\}} 0 + n = n$ [introduction d'axiome]
   – $\Gamma_{Peano} \bigcup \{0 + n = n\} \vdash_{\Sigma_{Peano} \bigcup \{n^0\}} succ(0 + n) = succ(n)$ [compatibilité]
   – $\Gamma_{Peano} \bigcup \{0 + n = n\} \vdash_{\Sigma_{Peano} \bigcup \{n^0\}} forallx.\forall y.(x + succ(y) = succ(x + y)$ [introduction d'axiome]
   – $\Gamma_{Peano} \bigcup \{0 + n = n\} \vdash_{\Sigma_{Peano} \bigcup \{n^0\}} 0 + succ(n) = succ(0 + n)$ [instantiation]
   – $\Gamma_{Peano} \bigcup \{0 + n = n\} \vdash_{\Sigma_{Peano} \bigcup \{n^0\}} 0 + succ(n) = succ(n)$ [transitivité]

On en déduit alors : $\Gamma_{Peano} \vdash_{\Sigma_{Peano}} \forall x.(0 + x = x)$

Bien entendu, nous devons montrer que cette règle est correcte, c'est-à-dire :

Théorème 10 : Étant donnée une signature Σ, si pour toute fonction $f^n \in \mathcal{C}$ on a :

$$\Gamma, \varphi(x/x_1), \ldots, \varphi(x/x_n) \models_{\Sigma \bigcup \{x_1^0, \ldots, x_n^0\}} \varphi(x/f(x_1, \ldots, x_n))$$

alors : $\Gamma \models_\Sigma \forall x.\varphi$.

Preuve : Montrons ceci par récurrence sur la taille des termes.
– **Cas de base :** par hypothèse de départ nous avons pour toute constante c de $\mathcal{C}$, $\Gamma \models_\Sigma \varphi(x/c)$
– **Pas de récurrence :** Soit t un terme de taille $n+1$. t est alors de la forme $f(t_1, \ldots, t_n)$. Chaque terme t_i est de taille inférieure à t, donc par hypothèse de récurrence, nous avons $\Gamma \models_\Sigma \varphi(x/t_i)$. Mais, par hypothèse de départ, nous avons :

$$\Gamma, \varphi(x/x_1), \ldots, \varphi(x/x_n) \models_{\Sigma \bigcup \{x_1^0, \ldots, x_n^0\}} \varphi(x/f(x_1, \ldots, x_n))$$

Donc, pour tout $\Sigma \bigcup \{x_1^0, \ldots, x_n^0\}$-modèle $\mathcal{M}$ où chaque constante x_i^0 pour $i \in [1, n]$ est interprétée par t_i et tel que $\mathcal{M} \models_{\Sigma \bigcup \{x_1^0, \ldots, x_n^0\}} \Gamma$ et pour tout $i \in [i, n]$, $\mathcal{M} \models_{\Sigma \bigcup \{x_1^0, \ldots, x_n^0\}} \varphi(x/x_i)$, nous avons aussi : $\mathcal{M} \models_{\Sigma \bigcup \{x_1^0, \ldots, x_n^0\}} \varphi(x/f(t_1, \ldots, t_n))$, d'où nous concluons :

$$\Gamma, \varphi(x/t_1), \ldots, \varphi(x/t_n) \models_\Sigma \varphi(x/f(x_1, \ldots, x_n))$$

par le théorème de complétude du calcul des prédicats du 1er ordre, nous savons alors :
– pour chaque terme t_i, $\Gamma \vdash_\Sigma \varphi(x/t_i)$
– $\Gamma, \varphi(x/t_1), \ldots, \varphi(x/t_n) \vdash_\Sigma \varphi(x/f(x_1, \ldots, x_n))$
Par le théorème de la déduction, nous obtenons :

$$\Gamma \vdash_\Sigma \varphi(x/t_1) \Rightarrow (\ldots \Rightarrow (\varphi(x/t_n) \Rightarrow \varphi(x/f(t_1, \ldots, t_n))) \ldots)$$

En appliquant n fois le modus ponens, nous obtenons alors : $\Gamma \vdash_\Sigma \varphi(x/f(t_1, \ldots, t_n))$
qui par le théorème de complétude donne : $\Gamma \models_\Sigma \varphi(x/f(t_1, \ldots, t_n))$, ce que nous voulions démontrer.

Nous avons donc montré que pour tout terme $t \in T_\Omega(\emptyset)$, $\Gamma \models_\Sigma \varphi(x/t)$. Les domaines des Σ-modèles étant une abstraction de l'ensemble $T_\Omega(\emptyset)$ [18], nous concluons directement par : $\Gamma \models_\Sigma \forall x.\varphi$. □

[FIN COURS 8]

[18] En effet, pour tout Σ-modèle $\mathcal{M}$, l'application $\_^\natural : T_\Omega(\emptyset) \to M$ est surjective, nous pouvons alors à partir de $\_^\natural$ définir la relation déquivalence $\equiv$ par : $t \equiv t' \Leftrightarrow t^\natural = t'^\natural$ pour obtenir : $M = T_\Omega(\emptyset)_{/\equiv}$.

Chapitre 6 : Logiques modales

TBFL *Voir par exemple le vieux poly bien fait de Christine Froidevaux*