Logique mathématique : introduction.

Paul Rozière
Paris 7 – MT3062

29 septembre 2004
(version provisoire — 17: 36)

Introduction.

La logique est souvent associée à « l'art de raisonner ». Elle étudie un certain type de discours argumenté, étude qui a commencé très tôt. Ainsi Aristote (300 av JC), dégage certaines figures de raisonnement (les syllogismes) qui sont valides indépendamment des assertions qu'elles mettent en oeuvre. C'est exactement le terrain d'étude de la logique : ce qui dans le raisonnement est indépendant du sujet étudié.

Très tôt également la logique est associée aux mathématiques, comme terrain d'étude privilégié. Déjà l'ambition des mathématiciens grecs de l'antiquité est effet de présenter leur science comme purement déductive : les théorèmes se déduisent d'autres théorèmes et ultimement de certains axiomes bien identifiés considérés comme évidents. Les chaînes de déductions sont purement formelles : elles peuvent être établies indépendamment du sujet étudié. Les éléments d'Euclide (300 av JC) en sont l'exemple le plus achevé, puisqu'il va constituer le cadre formel des mathématiques européennes jusqu'au XVIIième siècle.

Cependant, sans négliger les apports antérieurs, on peut dire que la logique moderne – celle que nous allons étudier – date essentiellement de la deuxième moitié du XIXième siècle, avec les travaux fondateurs de George Boole, Augustus De Morgan, Charles S. Peirce et surtout Gottlob Frege.

Le développement du calcul intégral et du calcul infinitésimal introduits par Isaac Newton et Gottfried Leibniz au cours du XVIIième siècle, a conduit à sortir du cadre de la géométrie d'Euclide. C'est au cours du XIXème siècle que l'on définit formellement les notions qui fondent l'analyse moderne comme celles de limite, et de continuité. L'ambition de certains mathématiciens comme Richard Dedekind et Georg Cantor est alors de redonner des fondements axiomatiques sûrs aux mathématiques, en partant non plus de la géométrie mais de l'arithmétique, puis de la notion d'ensemble.

Parallèlement l'ambition de certains logiciens de l'époque[1] est de mathématiser la logique, de l'axiomatiser de la même façon qu'une théorie mathématique, et ils utilisent pour cela des notions et des notations, comme la notation fonctionnelle, les variables, apparues en mathématique.

Le premier système logique à la fois entièrement formalisé et suffisamment riche pour formaliser les mathématiques (mais ce n'était pas sa seule ambition) est dû à Frege en 1879.

Frege souhaite donner des fondements purement logiques aux mathématiques. Il rejoint en cela Cantor qui fonde les mathématiques sur la théorie des ensembles (mais ne cherche pas à formaliser la logique elle même). La notion d'ensemble est en effet très proche de de la notion logique de "prédicat" (une propriété définit l'ensemble des objets ayant cette propriété). La théorie des ensembles est d'ailleurs considérée actuellement comme une branche de la logique mathématique.

Le développement de la logique a permis ensuite de clarifier puis de reformuler ces axiomatisations, après la découverte de *paradoxes* dans les théories de Cantor et Frege. L'élaboration de la logique comme discipline mathématique a permis au début du XXième siècle de poser de façon précise un certain nombre de problèmes relatifs aux fondements des mathématiques (c'est le cas d'un certain nombre des "problèmes futurs des mathématiques" listés en 1900 par David Hilbert). Ainsi Kurt Gödel a pu démontrer en 1931 le premier théorème d'incomplétude, qui fixe les limites des axiomatisations, à savoir que dans toute théorie axiomatique "raisonnable", c'est à dire pour laquelle il est possible de reconnaître mécaniquement les axiomes parmi les énoncés de la théorie et "suffisamment expressive", c'est à dire permettant de développer l'arithmétique[2], il restera toujours des énoncés qui ne sont pas conséquences de la théorie en question et dont la négation n'est pas non plus conséquence de la théorie. Dit d'une façon plus platonicienne, il existe des énoncés "vrais" de l'arithmétique qui ne sont démontrables dans aucune théorie axiomatique "raisonnable".

Ce cours est un cours d'introduction. On s'efforcera d'abord de faire saisir les notions fondamentales comme celles de démontrabilité et de vérité. Le théorème le plus élaboré que nous démontrerons, le théorème de complétude de Gödel, fera justement le rapport entre ces deux notions. On verra également dans quelle mesure on peut axiomatiser les mathématiques et de quelle

[1] Le premier à poser un tel programme est le mathématicien et philosophe du XVIIième siècle Leibniz

[2] il est tout à fait possible de donner un sens précis à ces deux hypothèses

façon. On ne démontrera pas le théorème d'incomplétude de Gödel cité au paragraphe précédent, mais au moins son énoncé devrait devenir plus compréhensible.

Afin d'éviter les malentendus, précisons que ce cours ne traite que de logique mathématique. Bien-sûr la logique ne se réduit pas à la logique mathématique. Cette dernière a quelques caractéristiques très particulières. Elle est bien plus pauvre que la logique naturelle : la logique mathématique classique n'a que deux valeurs de vérités, un énoncé est vrai ou faux, il n'y a pas de notion d'incertain, de possible, de nécessaire, le temps n'intervient pas ... Mais en un autre sens la logique mathématique est bien plus riche que la logique naturelle : les énoncés peuvent être beaucoup plus complexes, certains raisonnements comme le raisonnement par l'absurde semblent surtout utilisés en mathématique, les chaînes de déductions sont beaucoup plus longues...

Ce cours ne sera pas non plus un apprentissage de «l'art de raisonner» en mathématique. La logique des mathématiques repose sur le présupposé d'une aptitude commune à raisonner qui nous permet de communiquer et de convaincre qu'un raisonnement est correct. S'il existe bien un raisonnement mathématique, il s'élabore sur une spécialisation du raisonnement commun dans le contexte des mathématiques. Ses spécificités s'acquièrent d'abord ... par la pratique des mathématiques (y compris bien-sûr la pratique de la logique mathématique), même si nous espérons que la formalisation de la logique que nous allons donner permettra de clarifier et de préciser cette pratique.

1 Une présentation informelle du langage de la logique mathématique.

Il n'est bien-sûr pas question d'étudier la langue mathématique en général.

Tout d'abord un texte mathématique contient quasi forcément des éléments de nature non mathématique, annotations utiles à la compréhension d'une preuve, mais aussi rappels historiques et bien d'autres choses.

Ensuite le langage formel que nous allons décrire est artificiel et ne couvre pas tous les énoncés mathématiques tels quels. Par exemple nous n'accepterons pas directement la formulation «Tout entier naturel est pair ou impair», que l'on considère comme une abréviation de «$\forall x \in \mathbb{N}(pair(x) \vee impair(x))$», à supposer que *pair* et *impair* aient été introduits dans le langage formel. L'avantage est que ce langage artificiel peut être présenté de façon précise et étudié mathématiquement. L'étude du langage mathématique en général serait un travail (difficile) de linguistique.

Nous allons commencer de décrire, assez informellement pour le moment, un langage formel pour les mathématiques, en isolant et précisant un certain nombre de notations du langage mathématique usuel. Il y a un peu d'arbitraire dans certains des choix de formalisation : nous essayons d'être compatibles avec l'assistant de preuves Phox qui sera utilisé en travaux pratiques.

Les notations que nous allons utiliser ($\forall$, $\exists$, $\Rightarrow$...), ne sont pas celles introduites par Gottlob Frege qui ont eu peu de succès, entre autre à cause de leur complexité. Elles sont essentiellement dues à Giuseppe Peano (1894) (à quelques questions de graphie près), et ont été popularisées par les "Principia mathematica" de Bertrand Russell et Alfred Whitehead (1910). Elles sont pour la plupart assez communes dans les mathématiques actuelles.

1.1 Les objets, les énoncés, les preuves.

En mathématiques on traite d'*objets* : les nombres, les points, les droites, les ensembles, etc. On énonce des propriétés de ces objets — les *énoncés* mathématiques — de façon organisée. Certains énoncés initiaux, les *axiomes* sont admis, considérés évidents sur des domaines connus (arithmétique, géométrie...), ou définissant implictement une certaine théorie (algèbre). On en déduit d'autres énoncés, les *théorèmes* en utilisant certaines règles de raisonnement la plupart du temps implicites mais que toute personne pratiquant les mathématiques admet. Les théorèmes décrivent des propriétés de moins en moins évidentes des objets considérés. Une *preuve* d'un énoncé est une construction qui permet de convaincre que l'on a bien utilisé pour déduire l'énoncé les règles de raisonnement communément admises.

Donnons en exemple un énoncé d'exercice. Nous reviendrons plus tard sur les preuves.

$$\text{Résoudre dans } \mathbb{R} \text{ l'inégalité } 2x - 5 < \sqrt{10 - x}. \tag{1}$$

Les mots $\mathbb{R}$, x, 5, $2x$, $2x - 5$, $5 - x$, 10 et $\sqrt{10 - x}$ désignent des objets (un ensemble, des nombres réels), et $2x - 5 < \sqrt{10 - x}$ un énoncé. La solution de l'exercice pourrait se conclure par :

$$\{x \in \mathbb{R} \ / \ 2x - 5 < \sqrt{10 - x}\} =]1, \frac{15}{4}[\tag{2}$$

qui désigne un énoncé. Les constituants $\{x \in \mathbb{R} \ / \ 2x - 5 < \sqrt{10 - x}\}$, 1, $\frac{15}{4}$ et $]1, \frac{15}{4}[$ désignent des objets. Cet énoncé est un théorème, même si en mathématique il n'a pas suffisamment de portée pour que l'on prenne la peine de le désigner comme tel.

Les énoncés se distinguent des objets en ce qu'ils sont susceptibles d'être vrai ou faux. Dans l'exemple précédent cela n'a aucun sens de dire que $]1, \frac{15}{4}[$ est vrai, mais on peut dire que l'égalité (2) est vraie. Le mot inégalité est une façon redondante de nommer l'énoncé.

Nous ne chercherons pas à analyser le mot résoudre, qui ne peut se comprendre que dans le cadre d'une certaine pratique scolaire.

1.2 Syntaxe et sémantique.

On a besoin en logique de distinguer entre le mot et ce qu'il désigne. Ainsi pour prendre un exemple dans la langue courante, quand on dit que le chien a 4 pattes, le mot chien fait référence à un quelque chose d'extérieur, ici le monde réel, au sens du mot chien, c'est la *sémantique*. C'est tout à fait différent quand l'on dit que "chien" a 5 lettres : on parle du mot "chien", c'est de la *syntaxe*.

De la même façon on peut dire que $2x - 5$ est construit comme la différence du produit de 2 par la variable x et de 5, c'est plutôt une remarque d'ordre syntaxique. On peut dire que $2x - 5$ désigne un réel dont la valeur dépend de celle du réel x, c'est une remarque d'ordre sémantique.

Par exemple les expressions "2", "1 + 1" et "2 + 0" sont syntaxiquement différentes mais désignent le même objet, l'entier 2.

La syntaxe d'un langage s'occupe de son lexique, de ses règles de formation. La sémantique d'un langage s'occupe de lui donner un sens, d'interpréter les expressions du langage dans un monde a priori extérieur au langage.

Quand des mots, des assemblages de signes, ne sont pas syntaxiquement corrects, ont dit qu'ils n'ont pas de sens. Par exemple "1+", $(3-2$, "$x =$" ne sont pas corrects syntaxiquement. C'est tout à fait différent de dire que "$1+1 = 1$" est faux (pour les entiers). En effet l'expression "$1+1 = 1$" est syntaxiquement correcte et a un sens : elle est fausse, c'est de la sémantique.

De même "Pour tout entier x, $x + 1 = 2 \Rightarrow 1$" n'est pas syntaxiquement correct, la phrase "Pour tout entier x, $x + 1 = 2 \Rightarrow x = 2$" est syntaxiquement correcte et fausse.

La formalisation de la syntaxe du langage doit permettre de décrire les expressions qui ont un sens sans faire référence à ce sens. C'est très difficile pour un langage usuel, mais tout à fait possible pour le langage artificiel que nous allons étudier.

Il n'est bien-sûr pas question de chercher une signification a un assemblage de signes qui n'est pas syntaxiquement correct, mais toute expression syntaxiquement correcte doit avoir un sens. Une expression syntaxiquement correcte est construite à partir de composants élémentaires des symboles comme "1", "x", "+", "=". L'interprétation d'une expression sera obtenue à partir de l'interprétation de ses composants élémentaires.

Les choses ne sont parfois pas si simples. Ainsi l'expression $\sqrt{10 - x}$ semble avoir un sens, elle ne désigne pourtant aucun nombre (aucun réel en tout cas) si x est remplacé par un réel strictement supérieur à 10. Des expressions comme "le plus grand nombre premier", "l'entier naturel strictement compris entre 0 et 1" ne désignent aucun entier. Ces expressions demandent qu'une certaine condition soit vraie pour pouvoir désigner vraiment un objet. On sort donc a priori du cadre de la syntaxe.

1.3 Syntaxe.

Entamons maintenant une description informelle de ce ce que pourrait être la syntaxe d'un langage mathématique formel. Tout d'abord on appellera *termes* les expressions du langage qui désignent des objets, on appellera *formules* les expressions du langage qui désignent des énoncés.

1.3.1 Les termes.

Les constituants élémentaires des termes sont les constantes, comme 0, 1 pour les entiers, et les variables, comme x, y, z. On a souvent besoin d'indiquer à quel domaine appartient une variable, voire une constante. Par exemple x désigne-t-il un réel ou un entier ? Telles quelles ces indications seraient de nature sémantique. On va parler de *sorte* : x est de sorte réel signifie que x varie a priori dans l'ensemble des réels. Quand un langage utilise plusieurs sortes d'objets on parle de langage multisorte, quand il utilise une seule sorte d'objet de langage monosorte.

On construit de nouveaux termes à l'aide de symboles de fonctions (ou opérations). Un symbole de fonction doit avoir une sorte qui indique son usage. Par exemple si nat désigne la sorte des entiers, l'addition sur les entiers est de sorte $nat \to nat \to nat$, ce qui indique qu'elle prend deux arguments a et b qui doivent être de sorte nat, des entiers, et que le terme obtenu $a + b$ est de sorte nat, un entier[3]. Si k est la sorte des objets d'un corps $\mathbb{K}$ est v celle des objets d'un espace vectoriel E, la multiplication externe est de sorte $k \to v \to v$, ce qui indique qu'elle prend en

[3]La notation $nat \to nat \to nat$ se parenthèse $nat \to (nat \to nat)$. Une fonction à deux arguments, comme l'addition, est vue comme une fonction qui associe à son premier x argument une nouvelle fonction, celle qui associe au deuxième argument y le résultat $x + y$, x étant fixé.

premier argument un élément du corps, en deuxième argument un élément de l'espace vectoriel et que le résultat est un élément de l'espace vectoriel.

Dès que l'on a des fonctions partielles la notion de sorte, qui ne sert qu'à restreindre la syntaxe, diverge de celle d'ensemble de définition, comme on l'a vu au paragraphe précédent pour $\sqrt{}$ sur $\mathbb{R}$. De façon analogue la division $\div$ sur $\mathbb{R}$ est une loi binaire définie de $\mathbb{R} \times \mathbb{R}^* \mapsto \mathbb{R}$, dans $x \div y$ le deuxième argument y doit être non nul. C'est une condition de nature sémantique (ce n'est pas l'écriture de y qui ne doit pas être 0, mais bien son interprétation). Pour contrôler l'usage de $\div$, r étant la sorte des réels, on se contente de dire que $\div$ est de sorte $r \to r \to r$, l'écriture $1 \div 0$ n'est donc pas interdite syntaxiquement. De même $\sqrt{}$ sera de sorte $r \to r$.

Une fonction associe à n arguments un objet : on dit que l'*arité* de la fonction est n, ou encore que la fonction est *n-aire*, *unaire* pour un argument, *binaire* pour deux etc.

Quand il est clair qu'il y a une seule sorte d'objet, on n'a pas besoin de parler de sorte, l'arité suffit. Ainsi il suffit de dire que l'addition sur les entiers est binaire (d'arité 2).

On utilise plusieurs notations pour l'application d'une fonction à ces arguments. La notation "par défaut" est la *notation préfixe*, si f est binaire et u et v sont des termes (de la bonne sorte) $f(u, v)$ est un terme. En logique on notera souvent seulement $f\ u\ v$.

Pour certains symboles de fonctions binaires comme $+$ et $\times$ on utilise la *notation infixe*, si u et v sont des termes $u + v$, $u \times v$ sont des termes.

Enfin la *notation postfixe* — le symbole fonctionnel est placé après les termes auquel il s'applique — est assez peu utilisée. Par exemple la notation usuelle $n!$ pour la factorielle (unaire) est postfixe.

Avant d'aller plus loin, ajoutons que le signe "=" sera utilisé, sauf mention explicite, pour l'égalité des objets, pas pour l'identité syntaxique des termes qui les désignent. Pour les formules, on utilisera le signe "$\equiv$" pour dire que deux formules sont équivalentes, c'est à dire ont même interprétation.

1.3.2 Les formules atomiques.

On distingue les *formules atomiques* qui ne se décomposent pas elle-mêmes en formules, des formules composées. On utilisera *prop* comme sorte des formules.

Les formules atomiques s'écrivent à l'aide d'un symbole de prédicat, ou de relation, appliqué à un ou plusieurs arguments. De la même façon que pour les fonctions ces prédicats doivent avoir une sorte et on peut parler d'arité. Les notations préfixes, infixes et postfixes peuvent être utilisées.

Ainsi l'égalité est un symbole de prédicat (ou de relation) binaire, la notations est infixe. la relation d'incidence en géométrie plane est une relation binaire entre deux sortes différentes : si p est la sorte des points et d celle des droites, la relation d'incidence a pour sorte $p \to d \to prop$. L'ordre $<$, le prédicat de divisibilité $|$ sont des symboles de prédicat binaire sur les entiers, de sorte $nat \to nat \to prop$.

Revenons sur les fonctions partielles, comme $\sqrt{}$ sur $\mathbb{R}$ dont on a vu qu'elles posaient des problèmes de syntaxe. Le logiciel Phox règle les choses de la façon suivante. Prenons pour exemple les entiers naturels. La sorte des entiers naturels est notée *nat*. On déclare par exemple 0 de sorte *nat*, la fonction successeur de sorte $nat \to nat$, l'addition et la soustraction de sorte $nat \to nat \to nat$. On déclare également un prédicat N unaire sur la sorte *nat*, c'est à dire de sorte $nat \to prop$, $N\ x$ signifiant "x est un entier". On peut écrire (l'équivalent de) $2 - 3$ qui est de sorte *nat*, mais on ne peut pas prouver $N\ 2 - 3$, qui signifierait que $2 - 3$ est un entier naturel. En fait, on ne pourra prouver aucune proposition à propos de $2 - 3$.

1.3.3 Les connecteurs.

On peut construire de nouvelles formules (de sorte toujours *prop*) en utilisant les *connecteurs*. Nous utiliserons les signes $\neg$ pour la négation, $\wedge$ pour la conjonction "et", $\vee$ pour la disjonction "ou", "$\Rightarrow$" pour l'implication, c'est à dire "Si. . ., alors . . .", $\Leftrightarrow$ pour l'équivalence. Le connecteur $\neg$ est unaire (de sorte $prop \to prop$), tous les autre sont binaires (de sorte $prop \to prop \to prop$) et en notation infixe. On peut construire par exemple la formule suivante du langage de l'arithmétique :

$$(x|y \wedge \neg y = 0) \Rightarrow (x < y \vee x = y)$$

Les parenthèses et les crochets sont utilisées comme d'habitude pour rendre les écritures non ambiguës, c'est à dire pour qu'il n'y ait qu'une façon d'analyser la construction de la formule.

On utilisera aussi les constantes logiques $\perp$ pour l'absurde ou la contradiction, l'énoncé toujours faux, et $\top$ pour l'énoncé toujours vrai.

Le signe "$\equiv$" — $F \equiv G$ indique que les formules F et G ont même interprétation — ne fait pas partie lui même du langage. Il est équivalent de dire $F \equiv G$ et de dire que la formule $F \Leftrightarrow G$ est vraie.

1.3.4 Les quantificateurs.

Les quantificateurs apparaissent dans des expressions comme « tout entier naturel est pair ou impair », ("tout" marque la quantification universelle), ou « il existe deux entiers dont la somme des carrés est le carré d'un entier » ("il existe" marque la quantification existentielle), ou encore de façon moins apparente dans l'expression « x n'a d'autre diviseur que 1 et lui même ». Pour exprimer les quantifications on va utiliser les variables (l'utilisation des variables pour la quantification apparaît en logique formelle dans les travaux de Peirce et Frege). Les trois énoncés ci-dessus se traduiront par :

$$\forall n \in \mathbb{N}(pair(n) \vee impair(n)) \ ; \ \exists a \in \mathbb{N} \exists b \in \mathbb{N} \exists c \in \mathbb{N} \ a^2 + b^2 = c^2 \ ; \ \forall p \in \mathbb{N} \, [p|x \Rightarrow (p = 1 \vee p = x)]$$

Ces quantifications, comme le plus souvent en mathématique, sont des *quantifications bornées* : on indique le domaine sur lequel varie la variable quantifiée. On peut en fait décomposer ces écritures en utilisant les connecteurs. La quantification universelle bornée utilise l'implication, ainsi

$$\forall n \in \mathbb{N}(pair(n) \vee impair(n)) \equiv \forall n \, [n \in \mathbb{N} \Rightarrow (pair(n) \vee impair(n))]$$

(pour tout n si n est un entier, alors n est pair ou impair). La quantification existentielle bornée utilise la conjonction, ainsi

$$\exists a \in \mathbb{N} \exists b \in \mathbb{N} \exists c \in \mathbb{N} \ a^2 + b^2 = c^2 \equiv \exists a \exists b \exists c (a \in \mathbb{N} \wedge b \in \mathbb{N} \wedge c \in \mathbb{N} \wedge a^2 + b^2 = c^2) \ .$$

Un quantificateur non borné prend en argument une variable (d'une certaine sorte) et une formule et construit une nouvelle formule. La formule $n \in \mathbb{N} \Rightarrow (pair(n) \vee impair(n))$ dépend de n, alors que la formule $\forall n \, [n \in \mathbb{N} \Rightarrow (pair(n) \vee impair(n))]$ n'en dépend plus puisqu'elle peut se dire sans cette variable ("tout entier est pair ou impair"). On remarque également que le nom n n'a pas d'importance, la formule considérée s'écrit tout aussi bien $\forall p \, [p \in \mathbb{N} \Rightarrow (pair(p) \vee impair(p))]$. Bien-sûr ces remarques sont d'ordre sémantique, mais on comprend bien que l'on peut définir ces notions sans faire appel au sens des formules considérées.

On dira donc que la variable n est *libre* dans la formule $n \in \mathbb{N} \Rightarrow (pair(n) \vee impair(n))$ et *liée* dans la formule $\forall n \, [n \in \mathbb{N} \Rightarrow (pair(n) \vee impair(n))]$. On dit parfois *variable muette* pour variable liée, *variable parlante* pour variable libre.

On dit qu'une formule qui ne contient aucune variable libre est *close*. Ainsi les formules $\forall n \in \mathbb{N}(pair(n) \vee impair(n))$ et $\exists a \in \mathbb{N} \exists b \in \mathbb{N} \exists c \in \mathbb{N} \ a^2 + b^2 = c^2$ sont closes.

La formule $\forall p \in \mathbb{N} \, [p|x \Rightarrow (p = 1 \vee p = x)]$ n'est pas close : la variable p est liée mais la variable x est libre, l'énoncé original ("x n'a d'autre diviseur que 1 et lui même") dépendait bien de x.

1.3.5 Signes lieurs.

Les quantificateurs $\forall$ et $\exists$ ne sont pas les seuls signes *lieurs* (on dit aussi *mutificateurs*) du langage mathématique, ni les premiers apparus historiquement. Vous connaissez par exemple la notation pour l'intégrale : dans l'expression "$\int_0^x (t^2 + t + a)dt$" la variable t est liée, les variables x et a sont libres. D'autres signes connus sont :

- la somme et le produit (fini ou infini), $\sum_{i=0}^n i^2 + i$ (i liée, n libre), $\prod_{n=1}^\infty (1 + \frac{1}{n})^{\frac{1}{n}}$ (expression close, n liée), la réunion, l'intersection etc.

– l'ensemble des éléments d'un ensemble ayant une propriété donnée : $\{x \in \mathbb{N} \ / \ x^2 < 100\}$, $\mathcal{P}(A) = \{X \ / \ X \subset A\}$ (notation en compréhension) ;

– les fonctions : la fonction de $\mathbb{R} \to \mathbb{R}$, $x \mapsto e^x$ (on utilise aussi en logique la notation $\lambda x.e^x$).

Remarquez que les signes comment l'intégrale, la somme discrète etc. associent à un terme un terme, mais que la notation, d'ensemble en compréhension associe à une formule (la propriété caractéristique des éléments d'un ensemble), un terme (désignant cet ensemble).

La liste précédente n'est pas exhaustive. On pourrait ajouter des expressions plus proche du langage mathématique usuel comme « l'ensemble des solutions sur $\mathbb{R}$ de l'équation $ax^2 + bx + c$ » (ne dépend manifestement pas de x), « l'ensemble des vecteurs du plan de la forme $a\vec{u} + b\vec{v}$, avec $a, b \in \mathbb{Z}$ » etc.

Un même nom de variable liée peut désigner plusieurs variables différentes dans une expression. Par exemple l'expression $\int_a^b (x^2 + x)dx + \int_b^c (x^2 + x)dx$ fait apparaître deux variables liées de nom x (on peut tout aussi bien l'écrire $\int_a^b (u^2 + u)du + \int_b^c (v^2 + v)dv$).

On est donc amené à parler de la place d'un signe, nom de variable ou autre, dans une expression, on dit plutôt *occurrence* de variable. Par exemple dans la dernière expression x a 6 occurrences qui correspondent à deux variables liées. Toutes les occurrences n'ont pas le même usage. Dans $\int_a^b (x^2 + x)dx$, le x de dx est appelé *occurrence indicative*. De même pour les quantifications : la première occurrence de n dans $\forall n(pair(n) \vee impair(n))$ est l'occurrence indicative.

On parle également de *portée* ou de *champs* d'un signe lieur (ou de la variable associée) : si x est l'occurrence indicative, la sous-expression où toute occurrence libre de la lettre x dans la sous-expression fait référence à cette variable. Pour l'intégrale le champs est limité par les signes $\int$ et dx, pour un quantificateur c'est la formule immédiatement à droite de l'occurrence indicative (il faut éventuellement des parenthèses) etc. Remarquez que les occurrences liées de x à l'intérieur du champs d'une occurrence de x ne peuvent être liées à nouveau. Ainsi dans l'expression $\forall x \exists y[\exists x \ y = 2x \wedge (x = y \vee x = y + 1)]$, on a deux variables liées différentes de nom x dont les champs sont superposés. On peut écrire plus clairement $\forall x \exists y[\exists z \ y = 2z \wedge (x = y \vee x = y + 1)]$.

L'usage d'un même nom pour des variables dont les champs sont superposés, comme dans l'expression du paragraphe précédent, est à éviter car confus, même si formellement on peut l'accepter car on peut lui donner un sens. On considère qu'un variable libre a pour champs toute l'expression. Ainsi vous savez déjà qu'il vaut mieux ne pas écrire

$$\int_0^x (x^2 + x)dx \quad \text{ou même} \quad x + \int_0^y (x^2 + x)dx$$

(x apparaît comme variable libre et comme variable liée). On dira que l'expression est *polie* quand les champs de variables de même nom y sont toujours disjoints. Remarquez que l'usage de deux variables liées de même nom mais de champs disjoints, comme dans l'expression $\int_a^b (x^2 + x)dx + \int_b^c (x^2 + x)dx$ doit être admise.

Il existe aussi des formes de liaison moins évidente, comme les liaisons sans occurrence indicative : "l'ensemble des solutions de l'équation...", "les vecteurs de la forme...".

Pour terminer résumons ce qui permet de distinguer variables libres et liées.

Si une variable x est *liée* ou *muette* dans une expression E, alors on obtient une expression de même sens en remplaçant *toutes* les occurrences de cette variable x par une lettre n'apparaissant pas dans l'expression E. Sauf quand on s'autorise certaines libertés dans l'écriture des liaisons (occurrence indicative manquante), on obtient un assemblage qui n'a pas de sens en remplaçant *toutes* les occurrences d'une variable liée x par une constante ou un terme (désignant une objet du domaine dans lequel x varie) qui n'est pas une variable.

Si une variable x est *libre* ou *parlante* dans une expression E, alors on obtient une expression qui a un sens (en général différent) en remplaçant *toutes* les occurrences de cette variable x par n'importe quel terme (désignant une objet du domaine dans lequel x varie).

1.3.6 Logique du premier ordre et logique d'ordre supérieur.

Revenons sur la distinction entre objets et formules. Les ensembles sont des objets, or ils sont définis à l'aide d'une propriété. Ainsi on peut définir "x est pair $\equiv_d \exists q \in \mathbb{N} \ x = 2 \times q$".

La propriété “être pair” pourrait être notée “$x \mapsto \exists q \in \mathbb{N}\ x = 2 \times q$”, c'est un prédicat de sorte *nat* $\to$ *prop* (elle associe une formule à un entier). Ce n'est d'ailleurs pas une notation très différente de $\{x \ / \ \exists q \in \mathbb{N}\ x = 2 \times q\}$, simplement on l'interprète différemment, et on lui donne éventuellement une sorte différente. On peut considérer directement les prédicats comme des ensembles (des objets) et donc autoriser par exemple des prédicats sur les prédicats (sorte $(nat \to prop) \to prop$), mais aussi des quantifications sur les prédicats (et éventuellement sur les fonctions). Dans ce cas on parle de *logique d'ordre supérieur.* Quand on autorise uniquement les variables d'objet (de sorte atomique) et les quantifications sur ces objets, on parle de *logique du premier ordre*, multisorte s'il y a plusieurs sortes d'objets, monosorte sinon.

La logique d'ordre supérieur telle que nous l'avons esquissée est stratifiée : on ne met pas au même niveau par exemple les entiers naturels (de sorte atomique) et les ensembles d'entiers (de sorte *nat* $\to$ *prop*). Ainsi un prédicat comme “être une loi associative sur un ensemble donné”, doit être redéfini à chaque niveau. Ce prédicat devrait s'appliquer à $(\mathbb{N}, +)$, mais aussi à $(\mathbb{Z}, +)$. Supposons que les éléments de $\mathbb{Z}$ soient des classes d'équivalences d'entiers naturels, donc des ensembles d'entier, on voit que la sorte du prédicat “être associatif” est

$$(nat \to prop) \to (nat \to nat \to nat) \to prop$$

dans le premier cas et que dans le second cas il faudrait remplacer *nat* dans la sorte ci-dessus par *nat* $\to$ *prop*. De la même façon il faudrait un ensemble vide à chaque niveau. On peut également avoir besoin (penser à des chaînes de quotients) de construire des ensembles d'objets de niveaux différents. Ces problèmes sont traités dans les “Principia mathematica” de Russell et Whitehead qui proposent une fondation des mathématiques sur une logique d'ordre supérieur.

On pourrait penser que la restriction syntaxique imposée par les sortes est trop contraignante. Un prédicat pourrait s'appliquer à des expressions de sorte différentes, ou de façon équivalente on pourrait considérer que la notation $\{x \ / \ \ldots\}$ construit un objet de la sorte de base “ensembles”. Mais ceci conduit à un système incohérent, et c'est justement pour cela que Russell a introduit les sortes dans son système logique.

En effet, si un prédicat peut s'appliquer à un objet de n'importe que sorte, il peut en particulier s'appliquer à lui même. On peut alors appliquer une variable de prédicat P unaire à elle même $P(P)$, c'est une formule, on peut prendre sa négation $\neg P(P)$. On définit maintenant un nouveau prédicat R par $R(P) \equiv_d \neg P(P)$. On obtient $R(R) \equiv \neg R(R)$: contradiction. C'est le *paradoxe de Russell.* On va reformuler le paradoxe dans une notation ensembliste plus familière, mais la formulation précédente montre bien comment les sortes empêchent le paradoxe : si les sortes doivent être respectées (aucune autre égalité entre sortes que syntaxique), un prédicat de sorte $a \to prop$ ne pourra jamais s'appliquer à lui-même car a et $a \to prop$ ne seront jamais identiques.

En notation ensembliste, avec le symbole $\in$, le prédicat P correspond à l'ensemble x, $P(y)$ à $y \in x$, et donc $P(P)$ à $x \in x$. L'ensemble de Russell qui correspond au prédicat R est donc $r = \{x \ / \ x \notin x\}$. Le paradoxe découle de $r \in r \equiv r \notin r$.

On ne va pas plus loin dans ce sens, d'autant que nous n'avons pas du tout formalisé la logique d'ordre supérieur et que nous ne le ferons pas. Le système Phox utilisé en séances machines est fondé sur une logique d'ordre supérieur avec plusieurs types de base. Ce choix est fait non pour des raisons fondationnelles, comme celles qui viennent d'être exposées, mais pour des raisons pratiques. La vérification syntaxique peut se faire mécaniquement à l'aide des sortes, et cela évite de gérer au niveau des preuves certains problèmes purement syntaxiques. Des langages de programmation comme ceux de la famille ML utilisent d'ailleurs les sortes (ou types) de façon assez analogue.

La logique que nous étudierons est essentiellement la logique du premier ordre à une seule sorte d'objet. Elle n'utilise pas d'autres signes lieurs que $\forall$ et $\exists$. Il s'avère qu'elle est à la fois suffisamment simple pour avoir permis une étude mathématique poussée, et suffisamment expressive pour que l'on puisse développer une théorie des ensembles du premier ordre qui puisse fonder les mathématiques. On verra qu'en théorie des ensembles, on évite le paradoxe de Russell non pas en restreignant par les sortes l'écriture de certaines formules, donc de certains prédicats, mais en considérant que certains prédicats (définis par des formules) ne définissent pas des ensembles. Quand on parlera de logique du premier ordre sans autre précision, il s'agit de la logique du premier ordre à un seul sorte d'objet.

La logique d'ordre supérieur peut d'ailleurs aussi s'exprimer, au travers d'un codage et d'une axiomatisation, comme une théorie du premier ordre.

1.3.7 Les notations purement logique.

Parmi les diverse notations abordées, certaines sont liées à un domaine particulier, le symbole "$+$" pour l'addition, le symbole "$<$" pour relation d'ordre, "le plus petit ...", d'autres seront utilisées dans tous les langages, le connecteur "$\Rightarrow$", le quantificateur "$\forall$", les variables etc. En logique du premier ordre il n'y a que des variables d'objets.

Les notations purement logiques comprennent donc, les variables, les connecteurs, nous nous restreignons à "$\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$", les quantificateurs, nous nous restreignons à "$\forall, \exists$", auxquels on ajoute le plus souvent "$=$" l'égalité entre objets (symbole de prédicat binaire) qui a la même interprétation quelque soit le domaine étudié : l'identité de ces deux arguments.

1.3.8 Signature et théorie.

Pour le moment il ne s'agit pas de définir le langage des mathématiques en général, mais le langage d'une théorie particulière : théorie des groupes, théorie de l'ordre, arithmétique, géométrie...

Outre les notations purement logiques du paragraphe précédent, une théorie utilise des symboles de constante, de fonctions (ou opérations) et de prédicats particuliers. La suite de ces symboles est appelée *signature*. Chacun de ces symboles doit être muni d'une sorte, l'arité suffit pour la logique du premier ordre.

Par exemple le langage de la théorie des ordres stricts (une seule sorte d'objet) a pour signature $(<)$, $<$ étant un symbole de prédicat binaire. La théorie des groupes peut s'exprimer dans le langage $(e, \cdot, (\ )^{-1})$, où e est un symbole de constante, $\cdot$ un symbole de fonction binaire, $(\ )^{-1}$ un symbole de fonction unaire. L'arithmétique peut s'exprimer dans le langage de signature $(0, s, +, \times, \leqslant)$, où 0 est un symbole de constante, s (pour le successeur) un symbole de fonction unaire, $+$ et $\times$ des symboles de fonction binaire et $\leqslant$ un symbole de prédicat binaire.

La géométrie plane s'exprimer naturellement comme une théorie à deux sortes d'objet, points (p) et droites (d), dont le langage a pour signature ϵ (relation d'incidence) prédicat binaire de sorte $p \to d \to \mathit{prop}$.

Un langage du premier ordre de signature $\mathcal{S}$ n'a pour termes que des termes formés à partir des variables, des constantes de $\mathcal{S}$ quand $\mathcal{S}$ en contient, composés avec des symboles de fonctions de $\mathcal{S}$, si $\mathcal{S}$ en contient.

Un langage du premier ordre de signature $\mathcal{S}$ n'a pour formules atomiques que des prédicats appliqués à des termes de signature $\mathcal{S}$, et des égalités entre termes de $\mathcal{S}$ (sauf cas particuliers). Les formules du langage sont obtenus uniquement en composant les formules atomiques à l'aide des connecteurs et des quantificateurs.

Une *théorie* dans un langage de signature $\mathcal{S}$ est un ensemble de formules *closes* de ce langage. Par exemple la théorie des ordres stricts, dans le langage de signature $(<)$, comporte les deux formules closes :

$$\begin{array}{ll} \forall x \forall y \forall z (x < y \Rightarrow (y < z \Rightarrow x < z)) & \text{transitivité} \\ \forall x \neg x < x & \text{irreflexivité} \end{array}$$

Ceci est très restrictif. Par exemple on ne peut dire qu'un groupe est simple dans le langage du premier ordre de la théorie des groupes : pour dire "tout sous-groupe distingué est trivial", il faut quantifier sur les sous-groupes. En logique du premier ordre on ne peut quantifier que sur les éléments du groupe. Dans le langage du premier ordre des ordres, On ne peut pas définir un bon ordre (tout ensemble non vide admet un plus petit élément). Pour traiter ces notions on doit donc passer en logique d'ordre supérieur, ou changer de signature.

Rappelons que la théorie des ensembles, qui permet théoriquement d'exprimer toutes les mathématiques, est une théorie du premier ordre de signature $(\in)$ (un symbole de relation binaire).

1.4 Preuves et vérité.

Nous nous restreignons aux langages du premier ordre à une seule sorte d'objet. Après avoir décrit (informellement pour le moment) la syntaxe de ces langages, il s'agit maintenant de décrire leur sémantique, le sens de leurs éléments. Pour décrire la sémantique d'un langage naturel il est assez facile, au moins en apparence, de faire référence au monde extérieur. En mathématiques c'est moins évident. On va décrire la sémantique du langage, d'une part par l'usage des éléments du langage, d'autre part en faisant référence à un monde mathématique considéré comme existant indépendamment.

On distingue deux usages : d'une part la preuve mathématique. Par exemple on peux démontrer la formule suivante dans la théorie des ordres stricts (voir paragraphe 2.1)

$$\forall x \forall y (x < y \Rightarrow \neg y < x) \quad \text{anti-symétrie stricte}$$

D'autre part on peut remarquer que l'ensemble des entiers naturels $\mathbb{N}$ muni de l'ordre usuel est un ordre strict. On montre ainsi par exemple qu'un ordre strict peut ne pas avoir de plus grand élément. En effet, tous les énoncés démontrés dans la théorie des ordres stricts sont vrais pour $(\mathbb{N}, <)$, qui n'a pas de plus grand élément. Ce genre d'usage est bien connu en algèbre. Pour montrer qu'un énoncé n'est pas conséquence de la théorie des groupes, il suffit d'exhiber ou de construire un groupe qui ne vérifie pas cet énoncé. On interprète donc une formule dans une structure. C'est ce genre d'usage que l'on appellera plus précisément sémantique du langage, même si cette sémantique n'épuise pas et de loin la signification des expressions du langage.

1.4.1 Signature et structure.

On généralise ici des notions bien connues en algèbre.

Étant donné une signature $\mathcal{S}$, une *$\mathcal{S}$-structure* $\mathcal{M}$ est la donnée :

– d'un ensemble *non vide*, soit M, appelé *ensemble de base* de la structure $\mathcal{M}$;
– d'un élément $\overline{c}^{\mathcal{M}}$ de M pour chaque symbole de constante c de $\mathcal{S}$;
– d'une fonction[4] $\overline{f}^{\mathcal{M}}$ de M^n dans M pour chaque symbole de fonction f d'arité n dans $\mathcal{S}$;
– d'un sous-ensemble $\overline{R}^{\mathcal{M}}$ de M^n pour chaque symbole de prédicat R d'arité n de $\mathcal{S}$.

En reprenant l'exemple ci-dessus, $(\mathbb{N}, <)$ est une $\mathcal{S}$-structure pour $\mathcal{S} = (<)$.

Si nous prenons le langage de l'arithmétique de signature $\mathcal{P} = (0, s, +, \cdot, \leqslant)$, on peut définir $\mathcal{N} = (\mathbb{N}, \overline{0}^{\mathcal{N}}, \overline{s}^{\mathcal{N}}, \overline{+}^{\mathcal{N}}, \overline{\cdot}^{\mathcal{N}}, \overline{\leqslant}^{\mathcal{N}})$ muni des constantes, opérations et relations, usuelles pour interpréter chacun des symboles de $\mathcal{P}$. Le symbole s est interprété par la fonction de $\mathbb{N} \to \mathbb{N}$ qui ajoute 1 à un entier, + par la fonction addition usuelle (à deux arguments) etc.

Des notations comme $\overline{s}^{\mathbb{N}}$, $\overline{+}^{\mathbb{N}}$ sont assez lourdes. S'il n'y a pas d'ambiguïté sur la structure concernée, on oubliera l'indication de celle-ci, par exemple on notera $\overline{s}$, $\overline{+}$, et nous noterons même de façon identique symbole et interprétation, s'il est clair d'après le contexte que l'on parle de l'interprétation.

Une autre $\mathcal{P}$-structure est $\mathbb{Z}$ muni des opérations et prédicats usuels, $\mathbb{Z}/2\mathbb{Z}$ en interprétant 0 par le 0 de $\mathbb{Z}/2\mathbb{Z}$, les opérations avec leur interprétation usuelle, et l'ordre par exemple par $\{(0,0), (0,1), (1,1)\}$ (ce qui signifie que dans cette structure l'on a $0 \leqslant 0$, $0 \leqslant 1$ et $1 \leqslant 1$, mais pas $1 \leqslant 0$)[5]. On pourrait tout aussi bien construire une structure $\mathcal{N}'$ d'ensemble de base N, où 0 est interprété par 1, s par la fonction constante nulle etc.

Quand il s'agit d'un modèle fini on représente souvent une opération binaire par sa table, un prédicat binaire par une table ou par un graphe.

Il est important que l'ensemble de base d'une structure soit non vide. Outre que le cas où l'ensemble de base est vide a assez peu d'intérêt, cela compliquerait les systèmes de preuves si l'on voulait que celles-ci restent correctes dans de telles structures.

Par contre nous n'avons pas éliminé les structures dont l'ensemble de base a un seul élément, un cas évidemment dégénéré puisque l'interprétation des symboles de fonctions et de constantes est imposée, et que les symboles de prédicats n'ont que deux interprétations possibles ($\emptyset$ ou M^n).

[4]Nous considérons que les fonctions sont *partout définies*.
[5]Cet ordre n'est pas compatible avec l'addition

1.4.2 Interprétation dans une structure.

Une fois donnée une $\mathcal{S}$-structure $\mathcal{M}$, l'interprétation des termes clos du langage de signature $\mathcal{S}$ $\mathcal{M}$ est fixée et se calcule naturellement à partir des interprétation des symboles de constantes et de de fonctions. Si le terme contient des variables libres, il faut commencer par affecter une valeur à chacune de ses variables, un objet de $\mathcal{M}$ pour déterminer l'interprétation du terme.

Dans une $\mathcal{S}$-structure $\mathcal{M}$ une formule du langage $\mathcal{S}$ est vraie ou fausse, on le décide suivant l'interprétation des symboles de prédicats de $\mathcal{S}$, dans $\mathcal{M}$, deux termes sont égaux s'ils désignent le même objet de $\mathcal{S}$. Pour interpréter une formule quelconque, il faut d'abord affecter une valeur à chacune de ses variables libres. La définition de ces interprétations suit l'intuition. On en verra plus tard une définition formelle.

Notons que l'ordre des quantificateurs est important. Par exemple $\exists x \forall y\ x \leqslant y$ est une formule vraie dans $\mathbb{N}$ est fausse dans $\mathbb{Z}$, qui affirme que $\leqslant$ a un "plus petit" élément, le x, qui vient en premier dans la lecture de la formule, ne peut dépendre de y, qui n'existe pas encore. $\forall y \exists x\ x \leqslant y$ est vraie dans $\mathbb{Z}$ et dans $\mathbb{N}$, là on peut choisir x en fonction de y, pour les deux structures $\mathbb{Z}$ et $\mathbb{N}$ (et les ordres larges en général), x convient.

Si une formule close du langage de signature $\mathcal{S}$ est vraie dans une $\mathcal{S}$-structure, on dit que cette structure *satisfait* la formule ou qu'elle est *modèle* de la formule. On étend ceci à une théorie – un ensemble de formules closes. Une théorie est satisfaite dans une structure, ou a cette structure pour modèle, si chacune de ses formules est satisfaite.

On ne s'intéresse bien-sûr qu'aux théories qui ont au moins un modèle, les théories dites cohérentes. Habituellement on se donne un ensemble d'axiomes, par exemple ceux de la théorie des groupes, et il s'agit d'en déduire des théorèmes qui seront donc satisfaits par tous les groupes (c.a.d. les modèles des axiomes de groupe). On peut définir la déduction ainsi, une formule close F se déduit d'une théorie T si tous les modèles de T sont modèles de F. C'est une définition sémantique qui n'est pas très opératoire. On va plutôt formaliser la déduction en donnant un certain nombre de règles de preuve : les figure communes du raisonnement. Ces règles respecterons bien-sûr la satisfaction, et mieux nous montrerons qu'elles capturent complètement la déduction sémantique : c'est le théorème de complétude de Gödel.

1.5 Langage et meta-langage.

Si on reprend les éléments, pour le moment très informels, que nous avons donné, on s'aperçoit que nous avons distingué le langage étudié, un langage du premier ordre de signature donnée, du reste du langage. Ne serait-ce que le signe "$\equiv$", certaines abréviations, comme "$a \neq b$" pour "$\neg a = b$", les quantifications bornées etc. ne font pas parties du langage étudié. Par ailleurs on présuppose des notions comme celles de structure, utilisées pour la sémantique de la logique du premier ordre qui ne sont pas complètement élémentaires. Quand on définira précisément la syntaxe d'un langage du premier ordre on verra également qu'il faut mettre en œuvre un attirail mathématique simple mais qui n'est pas non plus complètement élémentaire.

Pour éclaircir les choses on distingue le langage étudié du langage dans lequel on l'étudie que l'on appelle *meta-langage*. On ne se pose pas trop de question (en tout cas dans ce cours) sur la nature du meta-langage, mais clairement il utilise des notions mathématiques usuelles. La formalisation de la logique que nous avons commencé de décrire n'est pas une refondation de la logique à partir de rien.

La nécessité de séparer le langage du meta-langage apparait dans le paradoxe de Richard, simplifié ensuite par Berry qui est le suivant. Soit n le plus petit entier naturel tel que l'on ne puisse définir n en moins de 25 mots. On vient de définir l'entier en question en moins de 25 mots. Si maintenant on précise que cet entier ne peut être défini en moins de 25 symboles élémentaires dans le langage de signature $\mathcal{S}$, il est clair que cette expression est une expression du meta-langage et non du langage de signature $\mathcal{S}$, il n'y a plus de paradoxe.

2 Formalisation de la déduction.

2.1 Introduction.

La formalisation des preuves mathématiques que nous allons décrire est éloignée des preuves mathématiques réelles, à cause justement de son caractère formel. Elle en respecte cependant la structure.

Nous allons présenter la déduction naturelle, introduite par Gentzen en 1935. C'est le système de preuve sur lequel est fondé le logiciel Phox que vous utiliserez en TP. Avant de décrire ce système faisons quelques remarques préalables.

2.1.1 Structure arborescente des preuves.

Les preuves usuelles ont une structure arborescente. En effet même si une preuve est un texte, donc écrit linéairement, on est amené à distinguer différents cas : raisonnement par cas, raisonnement par récurrence, donc à faire deux preuves indépendantes l'une de l'autre pour prouver un résultat donné. Des mots comme «or» marquent aussi l'arborescence en faisant référence à un résultat déjà prouvé, dont ne dépendait pas jusqu'à présent le fragment de preuve en cours. Les numérotations de résultats sont également un moyen de faire référence à un résultat qui ne vient pas immédiatement d'être démontré.

2.1.2 Contexte et conclusion.

Ensuite on remarque qu'il y a implicitement dans une preuve une notion de *conclusion courante*, la formule que l'on cherche à démontrer ou que l'on vient de démontrer, et une notion de *contexte* courant, c'est à dire de résultats déjà démontrés, d'hypothèses faites au cours de la preuve.

Cette conclusion courante et ce contexte évolue au cours de la preuve. Par exemple pour le raisonnement par l'absurde, on ajoute au contexte la négation de la formule que l'on est en train de prouver (c'est à dire la conclusion courante), et on cherche à prouver une contradiction, la contradiction devient donc la conclusion courante.

Il est donc nécessaire, pour un système de preuve qui ne soit pas trop éloigné de la preuve ordinaire, de manipuler un couple constitué du contexte et de la conclusion, la conclusion se déduisant du contexte, c'est à dire une *relation de déduction* entre un nombre fini de formules (le contexte) et une formule (la conclusion) que l'on note

$$A_1, \ldots, A_n \vdash C$$

où $A_1, \ldots, A_n$ désignent les formules du contexte, et C la conclusion. Le signe $\vdash$ indique la déduction. Un tel couple est appelé aussi *séquent*, si on insiste plus sur l'aspect formel.

2.1.3 Un exemple de preuve.

Analysons une démonstration très simple. On a vu au paragraphe 1.4 les axiomes d'ordre strict. Une preuve de l'anti-symétrie d'un ordre strict est énoncée à la figure 1.

Montrons qu'un ordre strict est antisymétrique[1]*. Par l'absurde, supposons que $x < y$ et $y < x$,*[2] *par transitivité $x < x$,*[3] *ce qui est impossible par irreflexivité,*[4] *donc pour tous x et y, si $x < y$, alors on n'a pas $y < x$*[5]*.*

Fig. 1: Exemple de preuve

Cette preuve est analysée en identifiant à chacune des étapes (indiquée par un numéro sur la figure 1) la conclusion courante et le contexte courant, à la figure 2. On ne note dans le contexte que les formules qui apparaisent au cours de la preuve, et pas les axiomes de la théorie — la transitivité et l'irreflexivité.

Le système de preuves que nous allons introduire va manipuler explicitement des relations de déduction.

1. *Montrons qu'un ordre strict est antisymétrique*
   la relation de déduction initiale (que l'on cherche à prouver) est :

$$\vdash \forall x \forall y (x < y \Rightarrow \neg y < x)$$

2. *Par l'absurde, supposons que $x < y$ et $y < x$,*
   signifie que *l'on cherche à démontrer* une contradiction pour x et y quelconques vérifiant $x < y$ et $y < x$ la relation de déduction courante devient donc (la contradiction se note $\perp$) :

$$x < y, y < x \vdash \perp$$

3. *par transitivité $x < x$*
   signifie cette fois-ci que *l'on a démontré* $x < x$, en utilisant la transitivité :

$$\vdash \forall x \forall y \forall z (x < y \Rightarrow (y < z \Rightarrow x < z))$$

   la relation de déduction courante devient

$$x < y, y < x \vdash x < x$$

4. *ce qui est impossible par irreflexivité,*
   signifie que l'on a aboutit à une contradiction, en utilisant l'irreflexivité :

$$\vdash \forall x \; \neg x < x$$

   le séquent courant devient

$$x < y, y < x \vdash \perp$$

   celui que l'on souhaitait obtenir en 2.
5. *donc pour tous x et y, si $x < y$, alors on n'a pas $y < x$,*
   on réénonce le résultat démontré (voir 2).

Fig. 2: Première analyse de l'exemple

2.1.4 Règles d'introduction, règles d'élimination.

Il s'agit maintenant de dégager les *figures élémentaires du raisonnement*, pas exactement celles qui ont été utilisées dans l'exemple analysé à la figure 2, mais des règles plus élémentaires, qui par composition permettrons de retrouver celles utilisées pour l'exemple.

On va classer presque toutes les règles en deux familles, les *règles d'introduction* concernent la façon de prouver une formule, *les règles d'élimination* concernent la façon d'utiliser une formule dans la preuve[6].

On distingue suivant la forme des formules, plus précisément la dernière construction utilisée pour la formule, quantificateur ou connecteur.

Supposons que l'on veuille prouver une formule $\forall x \; F(x)$. Pour cela, on prend x "quelconque" (ce qui signifie que x est un nouveau nom de variable, c'est à dire que l'on ne sait rien de x pour le moment), et on prouve $F(x)$. La quantification est souvent associée à une implication. Pour prouver $F \Rightarrow G$, on suppose F (on l'ajoute au contexte) et on prouve G. Dans le cas de la quantification bornée $\forall x \in E \; F$, si on applique successivement les deux règles, on voit que pour prouver $\forall x \in E \; F$ $(\forall x (x \in E \Rightarrow F))$ on suppose que x est quelconque dans E, et on prouve F.

Ces deux règles sont des règles qui permettent de prouver une formule, ce sont des règles d'introduction. On peux décrire ces règles de deux façons qui correspondent à deux façons différentes de décrire la preuve, pour l'implication, celle que l'on a utilisé dans l'exemple « pour prouver

[6]Dans le logiciel Phox les commandes `intro` et `intros` utilisent les règles d'introduction, les commandes `elim`, `apply`, `left` et `lefts` utilisent les règles d'élimination.

$F \Rightarrow G$, on prouve G sous hypothèse F, c'est à dire $F \vdash G$ », mais aussi « on a prouvé G sous hypothèse F, on a donc prouvé $F \Rightarrow G$ » (dans la première formulation on décrit la preuve "en arrière", on procède par condition suffisante, dans le second cas on décrit la preuve "en avant", on procède par déduction directe).

Si on veut maintenant utiliser une formule $\forall x\ F(x)$, on en déduit $F(t)$ pour t un terme, c'est à dire t que désigne un objet du domaine sur lequel varie x. Si on veut utiliser une formule $F \Rightarrow G$, on prouve F et on en déduit G. Ces deux règles sont des règles d'élimination. Là encore on peut les décrire de plusieur façons, par exemple pour la règle d'élimination de $\forall$, « on a prouvé $\forall x\ F(x)$, on a donc $F(t)$ » (en "avant") ou encore « pour prouver $F(t)$ on prouve $\forall x\ F(x)$ » (en "arrière"). La règle d'élimination de $\Rightarrow$ se dit « on a $F \Rightarrow G$, on a F, on a donc prouvé G », mais aussi par exemple « on a $F \Rightarrow G$, pour prouver G, il suffit de prouver F ».

Pour analyser la démonstration donnée en exemple, il nous faut également les règles de la négation. On remarque que $\neg A$ est synonyme de $A \Rightarrow \perp$ — si A alors il y a contradiction. On a donc pour $\neg A$ la règle d'introduction : « pour prouver $\neg A$ on suppose A et on prouve $\perp$ (l'absurde) », et la règle d'élimination « on a prouvé $\neg A$ et on a prouvé A, on en déduit $\perp$ ».

Reprenons l'analyse de la démonstration donnée à la figure 2. L'étape 2 correspond à une suite de règles d'introduction utilisées "en arrière" sur les deux quantificateurs universels, puis sur l'implication, puis sur la négation. L'étape 3 correspond à une suite de règles d'élimination utilisées en avant (3 quantificateurs universels, deux implications). L'étape 4 correspond à une suite de règles d'élimination utilisées en avant (un quantificateur universel, une négation). L'étape 5 ne correspond pas à une règle, mais affirme que la preuve est terminée. Pour décomposer complètement la preuve les règles énoncées jusqu'ici ne sont pas tout à fait suffisantes. Il nous faut un point de départ, nous avons besoin en plus des axiomes de la théories d'axiomes purement logiques, les axiomes de la déduction, qui affirment que pour une formule F quelconque « de F on déduit F », c'est à dire «$F \vdash F$». La figure 3 analyse la preuve de la figure 1 en règles élémentaires de preuves.

1	$\vdash \forall x \forall y (x < y \Rightarrow \neg y < x)$	
2	$\vdash \forall y (x < y \Rightarrow \neg y < x)$	introduction de $\forall$ en arrière
	$\vdash x < y \Rightarrow \neg y < x$	introduction de $\forall$ en arrière
	$x < y \vdash \neg y < x$	introduction de $\Rightarrow$ en arrière
	$x < y, y < x \vdash \perp$	introduction de $\neg$ en arrière
3	$\vdash \forall x \forall y \forall z (x < y \Rightarrow (y < z \Rightarrow x < z))$	axiome de la théorie
	$\vdash \forall y \forall z (x < y \Rightarrow (y < z \Rightarrow x < z))$	élimination de $\forall$ en avant
	$\vdash \forall z (x < y \Rightarrow (y < z \Rightarrow x < z))$	élimination de $\forall$ en avant
	$\vdash x < y \Rightarrow (y < x \Rightarrow x < x)$	élimination de $\forall$ en avant
	$x < y \vdash x < y$	axiome de la déduction
	$x < y \vdash y < x \Rightarrow x < x$	élimination de l'implication en avant, à partir des deux séquents précédents
	$y < x \vdash y < x$	axiome de la déduction
	$x < y, y < x \vdash x < x$	élimination de l'implication en avant, à partir des deux séquents précédents
4	$\vdash \forall x\ \neg x < x$	axiome de la théorie
	$\vdash \neg x < x$	élimination de $\forall$ en avant
	$x < y, y < x \vdash x < x$	résultat de **3**
	$x < y, y < x \vdash \perp$	élimination de l'implication en avant, à partir des deux séquents précédents

FIG. 3: Analyse de l'exemple en règles élémentaires de preuves

On remarque à la figure 3 que les règles d'introduction sont utilisées en arrière, et les règles d'éliminations en avant. Il est également usuel de commencer par des règles d'introduction (en arrière). La structure de la preuve est bien arborescente, et on peut également la représenter comme à la figure 4. Les règles se lisent en avant de haut en bas et en arrière de bas en haut.

Dans ce cours d'introduction on ne s'occupera pas de la structure des preuves (c'est l'objet

$$
\begin{prooftree}
\AxiomC{$\vdash \forall x \,\neg x < x$}
\RightLabel{$\forall_e$}
\UnaryInfC{$\vdash \neg x < x$}
\AxiomC{$\vdash \forall x \forall y(x < y \Rightarrow (y < x \Rightarrow x < x))$}
\RightLabel{$\forall_e$}
\UnaryInfC{$\vdash \forall y(x < y \Rightarrow (y < x \Rightarrow x < x))$}
\RightLabel{$\forall_e$}
\UnaryInfC{$\vdash x < y \Rightarrow (y < x \Rightarrow x < x)$}
\AxiomC{$x < y \vdash x < y$}
\RightLabel{$\Rightarrow_e$}
\BinaryInfC{$x < y \vdash y < x \Rightarrow x < x$}
\AxiomC{$y < x \vdash y < x$}
\RightLabel{$\Rightarrow_e$}
\BinaryInfC{$x < y, y < x \vdash x < x$}
\RightLabel{$\neg_e$}
\BinaryInfC{$x < y, y < x \vdash \bot$}
\RightLabel{$\neg_i$}
\UnaryInfC{$x < y \vdash \neg y < x$}
\RightLabel{$\Rightarrow_i$}
\UnaryInfC{$\vdash x < y \Rightarrow \neg y < x$}
\RightLabel{$\forall_i$}
\UnaryInfC{$\vdash \forall y(x < y \Rightarrow \neg y < x)$}
\RightLabel{$\forall_i$}
\UnaryInfC{$\vdash \forall x \forall y(x < y \Rightarrow \neg y < x)$}
\end{prooftree}
$$

FIG. 4: Arbre de preuve de l'exemple

de la théorie de la démonstration), et on se contentera d'une description textuelle des arbres de preuves.

La façon la plus claire de comprendre la relation entre preuve formelle en déduction naturelle et preuve usuelle et de voir cette dernière comme une suite d'instructions pour la construction de la preuve formelle. C'est aussi ainsi que l'on peut comprendre les preuves dans le système Phox. Dans le cas d'une preuve mathématique usuelle il s'agirait souvent plutôt d'indications que d'instructions proprement dites.

2.2 La substitution.

Les règles de démonstration utilisent la substitution d'un terme à une variable. Dans les paragraphes précédents nous avons utilisé une notation fonctionnelle pour les formules $F(x)$, $F(t)$, et nous continuerons à le faire. Mais nous ne définirons pas formellement ces notions : s'il fallait formaliser, F désignerait une propriété et non plus une formule. On utilisera plutôt la substitution logique. On note $F[t/x]$ pour «la formule F dans laquelle t remplace x».

Il est entendu que la substitution ne remplace que les variables libres :

$$(\neg x < x)[y/x] \text{ est } \neg y < y \quad \text{mais} \quad (\forall x \; \neg x < x)[y/x] \text{ est } \forall x \; \neg x < x \; .$$

Il est également entendu que cette substitution renomme si nécessaire les variables liées de façon qu'une variable du terme substitué reste libre, c'est à dire qu'elle évite ce que l'on appelle la *capture de variable*, par exemple :

$$\forall x(x < y \to \neg y < x)[x/y] \text{ est } \forall z(z < x \to \neg x < z)$$

(on a dû renommer la variable liée x en z pour éviter la capture), en effet :

$$(x < y \to \neg y < x)[x/y] \text{ est } (x < x \to \neg x < x)$$

La substitution logique sera définie quand on formalisera les langages du premier ordre. Elle ne fait pas partie du langage étudié, c'est une notation du meta-langage.

2.3 Les règles de preuve.

On va énumérer les règles de preuves élémentaires, pour une signature L, et T une théorie dans le langage L.

Les règles manipulent des *relations de déduction dans la théorie* T entre un *multi-ensemble* de formules $A_1, \ldots, A_n$ et une formule C du langage L, on note :

$$A_1, \ldots, A_n \vdash_T C \; .$$

Un *multi-ensemble* est une suite non ordonnée de formules, ou encore un ensemble dans lequel on autorise les répétitions, on peut le formaliser comme une suite modulo permutation de ses termes, ou encore comme un ensemble de formules indexées. Concrètement on ne se soucie pas de l'ordre entre les formules $A_1, \ldots, A_n$, mais il pourrait y avoir des répétitions. Si Γ est un multi-ensemble, et A formule, on note Γ, A le multi-ensemble obtenu en ajoutant une occurrence de A. Si Γ et Δ sont des multi-ensembles, on note Γ, Δ l'union disjointe (la juxtaposition) des multi-ensembles Γ et Δ.[7]

Les règles que nous allons donner sont les seules que l'on puisse utiliser pour établir la relation de déduction $\Gamma \vdash_T C$. Pour chaque connecteur ou quantificateur on donne une règle dite *d'introduction*, qui donne la façon "standard" de prouver une formule construite avec ce connecteur ou quantificateur, et une règle dite *d'élimination* qui donne la façon "standard" d'utiliser une telle formule. Les règles sont énoncées "en avant".

La négation peut se définir à partir de l'implication et de l'absurde :

$$\neg A \equiv_{def} A \Rightarrow \perp \ .$$

On dérive immédiatement les règles de la négation, comme cas particuliers des règles de l'implication : voir figure 6.

La règle d'élimination de l'absurde est superflue : elle est en fait dérivable par affaiblissement et raisonnement par l'absurde.

Pratiquement, on intégrera le plus souvent aux règles de logiques (connecteurs et quantificateurs) les règles de contraction. C'est à dire que l'on peut relire ces règles en considérant que Γ, Δ désigne une union ensembliste sur certaines occurrences. On dira par exemple que de $A \vdash A$ et $A \vdash A$ on dérive $A \vdash A \wedge A$ par introduction de la conjonction, ce qui dissimule une contraction.

Remarquez que la substitution logique donne une façon "commode" de désigner des occurrences d'un terme dans une formule, et de gérer les problèmes de capture de variable.

Par exemple, pour la règle d'élimination de l'égalité, il faut se persuader que cela signifie que si $\Gamma \vdash B$ et $\Delta \vdash t = t'$, alors on a $\Gamma, \Delta \vdash B'$, où B' est une formule obtenue en remplaçant de façon cohérente (pas de capture de variables) une ou plusieurs occurrences du terme t par t'.

On a vu dans les paragraphes d'introduction que ces règles pouvaient se lire dans les deux sens, en avant, telles qu'elles sont énoncées dans la figure 5, ou en arrière. Au risque d'être très redondant, réénonçons les règles de la déduction naturelle de façon graphique, voir la figure 7. La lecture de haut en bas correspond à l'énoncé des règles "en avant", celle de bas en haut à la lecture "en arrière". On s'est restreint au cas purement logique (théorie sans axiome) et sans égalité.

Décrivons rapidement les règles que nous n'avons pas abordé dans les paragraphes précédents. Les règles de la conjonction sont particulièrement simples. La règle d'introduction affirme que pour prouver $A \wedge B$ il faut prouver A et il faut prouver B, les deux règles d'élimination affirme que si l'on a prouvé $A \wedge B$, on a prouvé A et l'on a prouvé B.

La première règle d'introduction de la disjonction affirme que pour prouver $A \vee B$ il suffit de prouver A, la seconde que pour prouver $A \vee B$ il suffit de prouver B. La règle d'élimination de la disjonction est celle du raisonnement par cas, si l'on a prouvé $A \vee B$, pour prouver C on il suffit de prouver C sous hypothèse A, puis de prouver C sous hypothèse B. Ceci correspond bien à, « si l'on a $A \vee B$, 1er cas : on suppose A, 2ième cas : on suppose B ».

La règles d'introduction du quantificateur existentiel affirme que pour prouver $\exists x\ F(x)$, on doit prouver $F(t)$ pour t désignant un objet du domaine sur lequel varie x. La règle d'élimination signifie que si l'on a $\exists x\ A(x)$, on peut poser x_0 vérifiant $A(x_0)$. Le fait que x_0 est libre dans le contexte signifie que, à cette étape, la seule propriété connue de x_0 est $A(x_0)$. Par exemple dans la phrase « On sait que x est pair, soit p tel que $x = 2p$ », « soit p tel que $x = 2p$ » correspond à la règle d'élimination de l'existentielle sur « $\exists p\ x = 2p$ », et p ne doit pas apparaître libre ailleurs (sinon on choisit un autre nom).

La règle d'élimination de l'absurde signifie que si l'on a une contradiction, on en déduit n'importe quelle formule. Elle est utilisée (souvent implicitement) par exemple dans un raisonnement par cas sur $A \vee B$: dans le contexte $\neg A$ on ne traite que le cas B.

[7] On pourrait formaliser la déduction avec la notion d'ensemble, plus familière que celle de multi-ensemble. La notion de multi-ensemble convient mieux pour des raisons qui tiennent à la théorie de la démonstration et que l'on n'exposera pas ici.

Affaiblissement et contraction.

Si $\Gamma \vdash_T C$, alors $\Gamma, A \vdash_T C$; si $\Gamma, A, A \vdash_T C$, alors $\Gamma, A \vdash_T C$;

Axiomes

Déduction. $\Gamma, A \vdash_T A$; **Théorie.** Si $A \in T$, alors $\vdash_T A$;

Connecteurs.

Implication.

Introduction. Si $\Gamma, A \vdash_T B$, alors $\Gamma \vdash_T A \Rightarrow B$;

Élimination. si $\Gamma \vdash_T A \Rightarrow B$ et $\Delta \vdash_T A$, alors $\Gamma, \Delta \vdash_T B$;
(cette règle est souvent nommée “Modus Ponens”)

Absurde.

Élimination. Si $\Gamma \vdash_T \bot$, alors $\Gamma \vdash_T A$;

Conjonction.

Introduction. Si $\Gamma \vdash_T A$ et si $\Delta \vdash_T B$, alors $\Gamma, \Delta \vdash_T A \wedge B$;

Élimination (gauche). Si $\Gamma \vdash_T A \wedge B$ alors $\Gamma \vdash_T A$;

Élimination (droite). Si $\Gamma \vdash_T A \wedge B$ alors $\Gamma \vdash_T B$;

Disjonction.

Introduction (gauche). Si $\Gamma \vdash_T A$, alors $\Gamma \vdash_T A \vee B$;

Introduction (droite). Si $\Gamma \vdash_T B$, alors $\Gamma \vdash_T A \vee B$;

Élimination. Si $\Gamma \vdash_T A \vee B$, Si $\Delta, A \vdash_T C$, si $\Delta', B \vdash_T C$, alors $\Gamma, \Delta, \Delta' \vdash_T C$;

Raisonnement par l’absurde.

si $\Gamma, \neg A \vdash_T \bot$, alors $\Gamma \vdash_T A$;

Quantificateurs.

Quantification universelle.

Introduction. Si $\Gamma \vdash_T A[y/x]$ où y est une variable qui n’apparaît pas libre dans Γ et A, alors $\Gamma \vdash \forall x A$; (cette règle est souvent nommée “généralisation”)

Élimination. Si $\Gamma \vdash_T \forall x A$, alors pour tout terme t de L on a $\Gamma \vdash_T A[t/x]$;

Quantification existentielle.

Introduction. Si $\Gamma \vdash_T A[t/x]$, alors $\Gamma \vdash_T \exists x A$;

Élimination. Si $\Gamma \vdash_T \exists x A$, Si $\Delta, A[y/x] \vdash_T C$ et si la variable y n’apparaît pas libre dans Δ, A et C, alors $\Gamma, \Delta \vdash_T C$;

Égalité.

Introduction. Pour tout terme t de L, $\vdash_T t = t$ (c’est un axiome) ;

Élimination. Pour tout terme t et t' de L, si $\Gamma \vdash_T A[t/x]$ et $\Delta \vdash_T t = t'$, alors $\Gamma, \Delta \vdash_T A[t'/x]$;

Aucune autre règle que les précédentes ne permet de dériver une relation $\Gamma \vdash C$.

Fig. 5: Règles de la déduction.

La seule règle qui ne rentre pas vraiment dans le schéma des règles d’introduction et d’élimination est celle du *raisonnement par l’absurde.* Cette règle ressemble fort à la règle d’introduction de

Négation.

Introduction. Si $\Gamma, A \vdash_T \bot$, alors $\Gamma \vdash_T \neg A$;

Élimination. Si $\Gamma \vdash_T \neg A$ et si $\Gamma \vdash_T A$ alors $\Gamma \vdash_T \bot$.

Fig. 6: Règles de la négation.

la négation, et en mathématique ces règles sont toutes deux appelées raisonnement par l'absurde. Ainsi le « par l'absurde » de l'exemple traité figure 1 correspond en fait à une règle d'introduction de la négation. Pour bien comprendre la différence, entre ces deux règles il faut d'abord noter que nous n'identifions pas $\neg\neg A$ et A. Bien-sûr, on prouve facilement en déduction naturelle que $A \vdash \neg\neg A$ (il faut l'introduction du $\neg$) et $\neg\neg A \vdash A$ (il faut le raisonnement par l'absurde).

Le raisonnement par l'absurde et l'introduction de la négation diffèrent d'une façon essentielle. Alors que la règle d'introduction de la négation ne fait qu'utiliser un composant de $\neg A$, à savoir A, pour prouver $\neg A$, la règle de raisonnement par l'absurde demande pour prouver A d'introduire la formule $\neg A$, qui n'est pas présente a priori. Sans cette règle la logique est plus faible, il s'agit de la *logique intuitionniste*, qui ne permet pas de démontrer par exemple le tiers-exclu $A \vee \neg A$ (ce sera démontré en TD).

2.4 La déduction.

On dit donc que la relation de déduction $A_1, \ldots, A_n \vdash_T C$ est *dérivable* quand elle est obtenue à partir des règles de preuves de la figure 5, et uniquement avec ces règles. Quand on affirme $A_1, \ldots, A_n \vdash_T C$ cela signifie que la relation est dérivable. Une formule close F est un *théorème* de la théorie T si $\vdash_T F$ est dérivable.

2.5 Quelques exemples de dérivation.

On a déjà vu un exemple dans les paragraphes précédents. On va montrer que la symétrie et la transitivité de l'égalité sont prouvables[8].

Lemme 2.1 *Pour tous termes u, v et w, on peut dériver seulement avec les règles de l'égalité :*

$$u = v, v = w \vdash u = w \qquad u = v \vdash v = u$$

Démonstration. Pour la transitivité : $u = v \vdash u = v$ et $v = w \vdash v = w$, (axiomes de la déduction), d'où par règle d'élimination pour l'égalité ($u = v$ est $u = x[v/x]$), $u = v, v = w \vdash u = w$.
Pour la symétrie : $\vdash u = u$ (introduction de l'égalité), et $u = v \vdash u = v$ (axiome de la déduction), d'où par règle d'élimination pour l'égalité ($u = u$ est $x = u[u/x]$), $u = v \vdash v = u$. ■

Ces relations sont démontrables pour n'importe quels termes u, v et w, en particulier des variables distinctes, soient x, y et z. On déduit alors par introduction pour l'implication puis pour le quantificateur universel :

$$\vdash \forall x \forall y \forall z (x = y \to y = z \to x = z) \qquad \vdash \forall x \forall y (x = y \to y = x) \ .$$

Vous avez d'autres exemples simples de dérivation dans le polycopié d'introduction à PhoX, dans les premières séances de TP et dans la feuille d'exercice sur la déduction. Des propriétés élémentaires comme la transitivité de la relation de déduction sont également montrées dans cette feuille.

[8]Cela sera utile pour la preuve de complétude

AFFAIBLISSEMENT

$$\frac{\Gamma \vdash B}{\Gamma, A \vdash B}\ w$$

CONTRACTION

$$\frac{\Gamma, A, A \vdash B}{\Gamma, A \vdash B}\ c$$

AXIOME

$$\frac{}{A \vdash A}\ \text{Ax.}$$

RÈGLES LOGIQUES

$$\frac{\Gamma \vdash A \quad \Delta \vdash B}{\Gamma, \Delta \vdash A \wedge B}\ \wedge_i \qquad \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash A}\ \wedge_{eg} \quad \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash B}\ \wedge_{ed}$$

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \Rightarrow B}\ \Rightarrow_i \qquad \frac{\Gamma \vdash A \Rightarrow B \quad \Delta \vdash A}{\Gamma, \Delta \vdash B}\ \Rightarrow_e$$

$$\frac{\Gamma, A \vdash \bot}{\Gamma \vdash \neg A}\ \neg_i \qquad \frac{\Gamma \vdash \neg A \quad \Delta \vdash A}{\Gamma, \Delta \vdash \bot}\ \neg_e$$

$$\frac{\Gamma \vdash A[y/x]}{\Gamma \vdash \forall x A}\ \forall_i\ (*) \qquad \frac{\Gamma \vdash \forall x A}{\Gamma \vdash A[t/x]}\ \forall_e$$

$$\frac{\Gamma \vdash A}{\Gamma \vdash A \vee B}\ \vee_{ig} \quad \frac{\Gamma \vdash B}{\Gamma \vdash A \vee B}\ \vee_{id} \qquad \frac{\Gamma \vdash A \vee B \quad \Delta, A \vdash C \quad \Delta', B \vdash C}{\Gamma, \Delta, \Delta' \vdash C}\ \vee_e$$

$$\frac{\Gamma \vdash A[t/x]}{\Gamma \vdash \exists x A}\ \exists_i \qquad \frac{\Gamma \vdash \exists x A \quad \Delta, A[y/x] \vdash C}{\Gamma, \Delta \vdash C}\ \exists_e\ (**)$$

(*) La règle $\forall_i$ ne vaut que si y n'a pas d'occurrences libres dans Γ, A.
(**) La règle $\exists_e$ ne vaut que si y n'a pas d'occurrences libres dans Δ, A, C.

ÉLIMINATION DE L'ABSURDE

$$\frac{\Gamma \vdash \bot}{\Gamma \vdash A}\ \bot_e$$

RAISONNEMENT PAR L'ABSURDE

$$\frac{\Gamma, \neg A \vdash \bot}{\Gamma \vdash A}\ \bot_c$$

Fig. 7: Règles de la déduction (2)

3 Définitions par induction.

Nous avons abordés très informellement un certain nombre de notions, termes, énoncés... Quand nous avons précisé par exemple ce que signifiait variables libres et liées, nous avons en fait défini les choses de façon circulaire. Pour traiter correctement ces notions on utilise les *définitions par induction*. Même la définition de la prouvabilité, décrite de façon beaucoup plus précise au chapitre 2, demandera en toute rigueur d'être définie par induction.

3.1 L'exemple des entiers.

La notion d'entiers est suffisamment intuitive pour que vous l'ayez utilisée sans la définir formellement, d'autant qu'elle est assez primitive en mathématique. Supposons cependant que nous voulions "définir" les entiers. Voyons deux façons de procéder.

Nous pouvons en donner une *axiomatisation* c'est à dire que les entiers sont définis implicitement, par leurs propriétés : l'ensemble des entiers muni d'une constante 0 et d'une opération unaire s (l'opération successeur, ajouter 1), qui vérifie un certain nombre de propriétés, dont une essentielle est la propriété de récurrence. L'*axiomatique de Peano* des entiers est la suivante :

successeur non nul $\forall x\ s\,x \neq 0$;

injectivité du successeur $\forall x \forall y\ (s\,x = s\,y \Rightarrow x = y)$;

Récurrence Pour toute propriété P sur les entiers :

$$P\,0 \Rightarrow \forall y(P\,y \Rightarrow P\,s\,y) \Rightarrow \forall x\ P\,x\ .$$

L'idée est que l'on obtient la suite des entiers en ajoutant 1 au précédent entier obtenu, ce en partant de 0. La propriété de récurrence exprime que l'on ne peut obtenir les entiers autrement. Les deux premiers axiomes expriment que l'on ne peut boucler ou revenir en arrière par cette opération, c'est à dire que la structure des entiers est *librement engendrée* par 0 et s : il n'y a pas d'autres relations entre les entiers que celles induites par la construction : $0,\ s\,0,\ ss\,0, \ldots$.

On peut également définir explicitement l'ensemble des entiers. On ne considère plus les entiers comme une notion primitive. On doit se placer dans un univers où existent un objet et une opération unaire qui vérifient les deux premiers axiomes, appelons les encore 0 et s (pensez aux entiers comme une suite de bâtons, 0 est le vide et indique le point de départ de la suite, s est l'opération "ajouter un bâton supplémentaire à la fin d'un entier"). On suppose également que l'on peut parler d'ensembles. On peut définir alors l'ensemble des entiers comme *le plus petit ensemble* $\mathbb{N}$ (plus petit au sens de l'inclusion) qui contient 0 et tel que si $x \in \mathbb{N}$ alors $s(x) \in \mathbb{N}$. Montrons que cette définition est correcte. Quand un ensemble A vérifie que si $x \in A$ alors $s(x) \in A$, on dit que A est *clos par application du successeur*. Appelons $Cl(A)$ le fait pour A de contenir 0 et d'être clos par successeur :

$$Cl(A) \equiv_d [0 \in A \text{ et } \forall x(x \in A \Rightarrow s(x) \in A)]\ .$$

On remarque que si chacun des ensembles d'une famille $(A_i)_{i \in I}$ vérifie $Cl(A_i)$ alors leur intersection également :

$$\text{si } \forall i \in I\ Cl(A_i), \quad \text{alors } Cl\left(\bigcap_{i \in I} A_i\right) \tag{$*$}$$

On peut donc définir l'ensemble des entiers comme l'intersection de tous les ensembles A qui contiennent 0 et qui vérifient que si $x \in A$ alors $s(x) \in A$:

$$\mathbb{N} = \bigcap_{Cl(A)} A\ .$$

Il s'agit bien d'un ensemble vérifiant $Cl(A)$ d'après $(*)$, et c'est forcément le plus petit par définition.

C'est ce que l'on appelle une *définition inductive*. C'est exactement comme cela que nous définirons les entiers en théorie des ensembles (il faudra donner explicitement 0 et s), ce qui permet de se ramener à la seule notion d'ensemble comme notion primitive.

La propriété ou principe de récurrence, dit également *principe d'induction* est alors une conséquence de la définition : pour une propriété P donnée il suffit de prendre $A = \{x \in \mathbb{N} \ / \ P\,x\}$. Par hypothèse cet ensemble contient 0 et est clos par successeur, c'est à dire vérifie $Cl(A)$, donc contient $\mathbb{N}$ et donc la propriété P est vérifiée sur $\mathbb{N}$.

On peut maintenant vouloir définir une suite par récurrence sur les entiers. C'est à dire qu'un entier a et une fonction binaire sur les entiers $f : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ étant donnée, on définit une suite $(u_n)_{n\in\mathbb{N}}$ par (on note $n+1$ plutôt que $s\,n$) :

$$\begin{aligned} &u_0 = a \\ &u_{n+1} = f(u_n, n) \end{aligned}$$

par exemple la fonction factorielle est définie à partir de 1 et de la multiplication par :

$$\begin{aligned} &0! = 1 \\ &(n+1)! = (n+1)\cdot n! \end{aligned}$$

On considère habituellement comme évident que l'on a bien défini ainsi une fonction, mais ceci constitue une nouvelle propriété que nous appelerons *principe de définition par induction*.

Remarquez que pour que cette définition soit correcte, il est essentiel que la structure des entiers soit librement engendrée. Ainsi, si l'on définit $\mathcal{N}$ en prenant pour 0 le 0 de $\mathbb{Z}/2\mathbb{Z}$ et pour s la fonction $x \mapsto x+1$ sur $\mathbb{Z}/2\mathbb{Z}$, la définition de $\mathcal{N} = \bigcap_{Cl(A)} A$ reste correcte (évidemment on ne définit pas les entiers, mais $\mathbb{Z}/2\mathbb{Z}$ lui-même qui est un ensemble fini et cela n'a pas grand intérêt). Par contre on ne peut pas donner une définition comme celle de $n!$ ci-dessus. Dans ce cas l'on aurait $1! = 1$, et $2 \mid 3!$, ce qui dans $\mathbb{Z}/2\mathbb{Z}$ donnerait $0 = 1$.

Dans le cadre d'une définition explicite des entiers, en toute rigueur il faut montrer ce principe. Voyons comment pour un principe de définition (ce n'est pas le plus général que l'on puisse énoncer).

Proposition 3.1 (Définition d'une fonction par induction) *Soit A un ensemble, $a \in A$ et $f : \mathbb{N} \times A \mapsto A$ une fonction. Alors il existe une et une seule suite u_n à valeur dans A telle que :*

$$\begin{aligned} &u_0 = a \\ &u_{n+1} = f(n, u_n) \end{aligned}$$

Démonstration. On définit inductivement le graphe G de la fonction. C'est le plus petit sous-ensemble de $\mathbb{N} \times A$ qui vérifie les deux clauses de définition de la fonction

$$\text{pour } X \subset \mathbb{N} \times A, \quad Cl(X) := (0, a) \in X \text{ et } \forall n \in \mathbb{N} \forall x \in A[(n, x) \in X \Rightarrow (n+1, f(n, x)) \in X]$$

$$G = \bigcap_{Cl(X)} X$$

Il est clair que la propriété $Cl(X)$ passe à l'intersection, et qu'elle est réalisée pour $X = \mathbb{N} \times A$. Montrons par récurrence que c'est bien le graphe d'une fonction de $\mathbb{N}$ dans A :

$$\forall n \in \mathbb{N} \exists! y \in A \ (n, y) \in G$$

$n = 0$: Par définition de G, $(0, a) \in G$. Si $(0, b) \in G$ pour $b \neq A$, $G - \{(0, b)\}$ satisfait la propriété de clôture Cl, (car $\forall n \in \mathbb{N}\ 0 \neq n+1\,!$) ce qui contredit la définition de G.

$n \to n+1$: Supposons que n ait une unique image et appelons la y_0. D'après la définition de G, $(n+1, f(n, y_0)) \in G$. Supposons que $(n+1, b) \in G$ pour $b \neq f(n, y_0)$. Alors l'ensemble $G - \{(n+1, b)\}$ a la propriété Cl. En effet d'une part on a bien $(0, a) \in G - \{(n+1, b)\}$ (car $0 \neq n+1\,!$). D'autre part de $(m, x) \in G - \{(n+1, b)\}$, on déduit $(m+1, f(m, x)) \in G - \{(n+1, b)\}$ pour $m \neq n$ (car $m+1 = n+1 \Rightarrow m = n$). Dans le cas où $m = n$, comme par hypothèse de récurrence, y_0 est le seul élément de A tel que $(n, y_0) \in G$, et que $b \neq f(n, y_0)$ la propriété on a encore si $(n, x) \in G - \{(n+1, b)\}$ alors $(n+1, f(n, x) \in G - \{(n+1, b)\}$. Mais $G - \{(n+1, b)\}$ satisfait Cl contredit $(n+1, b) \in G$, pour $b \neq f(n, y_0)$. on ne peut donc trouver un tel b, $n+1$ a bien une unique image par f. ■

On peut certe considérer comme plus intuitivement évident le principe de définition d'une fonction par induction sur les entiers que la démonstration qui en est faite ci-dessus ! Cette démonstration aura un sens dans le cadre de la théorie des ensembles (ou la notion d'entier n'est pas primitive). Pour construire les entiers il faudra donner des interprétations ensemblistes convenables de 0 et du successeur.

Dans le cadre d'une axiomatisation des entiers, on ne peut que poser comme axiome le principe de définition par induction (la proposition 3.1 où A est $\mathbb{N}$). Pour formuler cette axiomatisation au premier ordre, il faut ajouter au langage un symbole de fonction pour chaque fonction définie par induction, et à la théorie les axiomes qui définissent cette fonction. En pratique il suffit d'ajouter l'addition et la multiplication. On obtient alors l'arithmétique de Peano proprement dite, à savoir la théorie dans le langage $(0, s, +, \times)$ qui aux axiomes déjà donnés ajoute les axiomes définissant ces opérations par récurrence :[9]

$$\begin{array}{ll} \forall x\ x + 0 = x & \forall x\ x \times 0 = 0 \\ \forall x \forall y\ x + s\,y = s(x + y) & \forall x \forall y\ x \times s\,y = x + x \times y \end{array}$$

Ce sont les principes présentés dans le cas particulier des entiers : définition inductive d'un ensemble, principe d'induction, principe de définition d'une fonction (ou d'un prédicat) par induction sur une définition inductive librement engendrée, que l'on va utiliser pour définir les notions essentielles en logique. Les définitions inductives que nous utiliserons seront moins familières, mais finalement relativement intuitives. On admettra le principe de définition d'une fonction par induction pour les définitions inductives librement engendrées (la preuve suit à chaque fois le même schéma que celle donnée dans le cas particulier des entiers).

Mais voyons d'abord quelques exemples très simples de telles définitions.

3.2 Suites finies d'entiers.

Voyons un autre exemple de définition inductive d'une structure usuelle en mathématique (la définition et la notation ne sont pas usuelles). Supposons donnés une notion de couple, qui vérifie en particulier la propriété :

$$\forall x, y, x', y'\ [(x, y) = (x', y') \Rightarrow (x = x' \text{ et } y = y')]$$

et un objet () pour la suite vide qui n'est jamais égal à un couple.

On peut définir l'ensemble $\mathcal{S}$ suites finies d'entiers par induction, $\mathcal{S}$ est le plus petit ensemble qui vérifie :

Suite vide $() \in \mathcal{S}$

adjonction Si $U \in \mathcal{S}$ et $v \in \mathbb{N}$, alors $(U, v) \in \mathcal{S}$.

(la notation n'est pas la notation usuelle)

Ces propriétés sont bien stables par intersection.

D'après ce que l'on a supposé sur les couples et sur () :

- Pour toute suite finie U et tout entier v, $(U, v) \neq ()$.
- Pour toutes suites finies U, U', tous entiers v, v',

$$(U, v) = (U', v') \Rightarrow (U = U' \text{ et } v = v')\ .$$

[9]On peut alors d'une certaine façon définir implicitement les autres fonctions définies par induction sur les entiers et à image dans les entiers.Mais la démonstration nous entraînerais trop loin. Ce résultat constitue une étape importante de la preuve du premier théorème d'incomplétude de Gödel.

On peut définir par induction le produit fini, à partir de la multiplication sur les entiers :

$$\begin{array}{l} \Pi() = 1 \\ \Pi(U, v) = \Pi\ U \cdot v \end{array}$$

ainsi que le prédicat noté ϵ d'apparition dans la suite :

$$\begin{array}{l} \forall n \in \mathbb{N}\ n \not\epsilon () \\ n \epsilon (U, v) \Leftrightarrow (n \epsilon U \text{ ou } n = v) \end{array}$$

On peut alors démontrer par induction par exemple que

$$\forall U \in \mathcal{S}\ (\Pi\, U = 0 \Leftrightarrow 0 \epsilon U)$$

suite vide On a $\Pi() = 1$ et $0 \not\epsilon ()$.

adjonction On suppose le résultat pour U, à savoir

$$(\Pi\, U = 0 \Leftrightarrow 0 \not\epsilon U) \qquad \text{(hypothèse d'induction)}$$

Soit un entier v. On a $\Pi(U, v) = \Pi\, U \cdot v$ et $0 \epsilon (U, v) \Leftrightarrow (0 \epsilon U \text{ ou } 0 = v)$. Si $v = 0$ alors $\Pi(U, v) = 0$ et $0 \epsilon (U, v)$. Si $v \neq 0$ alors $\Pi(U, v) = 0 \Leftrightarrow \Pi\, U = 0$, et $0 \epsilon (U, v) \Leftrightarrow 0 \epsilon U$, ces deux énoncés sont donc équivalents par hypothèse d'induction.

3.3 Ensembles finis d'entiers.

Voyons un dernier exemple assez artificiel mais qui présente une structure inductive qui n'est pas librement engendrée : on va définir par induction l'ensemble $\mathcal{F}$ des ensembles finis non vides d'entiers. On suppose connues les notions d'ensembles, d'appartenance, de réunion etc. L'ensemble $\mathcal{F}$ est le plus petit ensemble tel que :

Singleton Pour tout entier n, $\{n\} \in \mathcal{F}$;

Réunion Si $A \in \mathcal{F}$ et $B \in \mathcal{F}$ alors $A \cup B \in \mathcal{F}$.

Ces propriétés sont stables par intersection. Cette définition est correcte. On a donc un principe d'induction qui peut, pour prendre un exemple trivial, nous permettre de montrer qu'un élément de F contient au moins un élément :

Singleton $n \in \{n\}$;

Réunion Si $A \in \mathcal{F}$ et $B \in \mathcal{F}$ alors, par hypothèse d'induction sur A, il existe un entier n tel que $n \in A$ et donc $n \in A \cup B$.

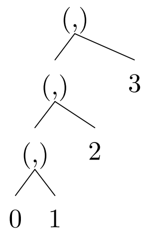
Par contre on peut obtenir un même ensemble de plusieurs façons différentes. Par exemple $\{0\}$ peut être obtenu directement comme singleton, ou comme $\{0\} \cup \{0\}$. On ne pourra donc pas donner directement de définition par induction sur F comme celles données dans les exemples précédents.

3.4 Arbres de dérivations.

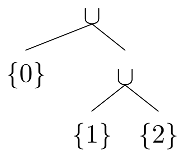
Étant donné un ensemble $\mathcal{E}$ défini par induction, et e un élément de $\mathcal{E}$ on appelle *dérivation* de e la suite des clauses de la définition qui ont permis de montrer que $e \in \mathcal{E}$. La dérivation se représente naturellement sous forme arborescente. Dans le cas des entiers cet arbre est un simple fil :

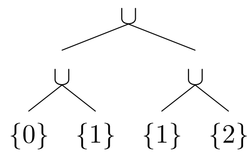
$$\begin{array}{cccc} 0 & 1 & 2 & \cdots \\ & & & \\ 0 & s & s & \\ & | & | & \cdots \\ & 0 & s & \\ & & | & \\ & & 0 & \end{array}$$

Dans le cas des suites d'entiers tels que définis au paragraphe 3.2 ci-dessus, les arbres de dérivations ont une forme de “peigne”, on a par exemple pour la suite $(((0,1),2),3)$



Dans le cas des ensembles finis tels que définis au paragraphe 3.2, $\{0,1,2\}$ a plusieurs arbres de dérivation, par exemple :





Pour la définition du paragraphe 3.1 un entier a un seul arbre de dérivation, de même une suite pour la définition du paragraphe 3.2. Une structure est librement engendrée si et seulement si tout élément défini possède un seul arbre de dérivation, et alors on peut utiliser le principe d'induction par définition.

3.5 Complexité d'une dérivation.

On appelle *complexité d'une dérivation* la hauteur de son arbre de dérivation, c'est à dire la longueur de la plus longue branche de cet arbre (comptée en arêtes). Si la structure est librement engendrée la complexité se définit par induction sur la structure. On peut déduire le principe d'induction sur la structure par récurrence sur la complexité. On verra que pour les formules du calcul des prédicats, on aura vraiment besoin dans la preuve de complétude de raisonner par récurrence sur la complexité de la formule.

4 Calcul propositionnel.

4.1 introduction.

On s'intéresse dans ce chapitre aux formules propositionnelles, c'est à dire sans quantificateurs. L'interprétation d'une formule propositionnelle close d'un langage $\mathcal{L}$ dans une $\mathcal{L}$-structure, sa vérité ou fausseté, ne dépend que de l'interpétation des formules atomiques qui la composent. Pour une formule propositionnelle quelconque, l'interprétation, une fois fixée la valeur des variables libres, ne dépend également que de celle des formules atomiques qui la composent. Pour étudier ces formules, on peut donc "oublier" la structure des formules atomiques : c'est le calcul propositionnel.

4.2 Syntaxe.

On définit la syntaxe du calcul propositionnel par induction. Soit un ensemble de constantes propositionnelles $\mathcal{P}$, l'ensemble des formules du calcul propositionnel sur $\mathcal{P}$ – nous dirons formules propositionnelles – est défini inductivement par les clauses suivantes :

atomes si $p \in \mathcal{P}$, p est une formule propositionnelle.

absurde $\perp$ est une formule propositionnelle.

négation Si F est une formule propositionnelle $\neg F$ est une formule propositionnelle.

implication Si F et G sont des formules propositionnelles $(F \Rightarrow G)$ est une formule propositionnelle.

conjonction Si F et G sont des formules propositionnelles $(F \wedge G)$ est une formule propositionnelle.

disjonction Si F et G sont des formules propositionnelles $(F \vee G)$ est une formule propositionnelle.

On utilisera également comme abréviations $\top$ pour $\perp \Rightarrow \perp$ (le "vrai") et $(F \Leftrightarrow G)$ pour $(F \Rightarrow G) \wedge (G \Rightarrow F)$ (l'équivalence).

Si l'on veut formaliser un problème en calcul propositionnel, on prendra pour $\mathcal{P}$ un ensemble qui a une certaine structure, des formules atomiques d'une certaine signature par exemple (voir la suite, paragraphe 4.4) mais on n'a pas besoin de connaître cette structure pour étudier la sémantique.

Une propriété importante et la propriété de lecture unique qui dit que la structure des formules du calcul propositionnel est librement engendrée, plus précisément

Proposition 4.1 (lecture unique) *Pour toute formules F, F', G et G' du calcul propositionnel (on utilise dans ce qui suit le signe "$=$" pour l'égalité syntaxique) :*

1. *si p est une variable propositionnelle, $\neg F \neq p$, $(F \Rightarrow G) \neq p$, $(F \vee G) \neq p$, $(F \wedge G) \neq p$;*
2. *$\neg F \neq (F \Rightarrow G)$, $\neg F \neq (F \vee G)$, $\neg F \neq (F \wedge G)$;*
3. *$(F \Rightarrow G) \neq (F \vee G)$, $(F \Rightarrow G) \neq (F \wedge G)$, $(F \vee G) \neq (F \wedge G)$.*
4. *si $\neg F = \neg F'$ alors $F = F'$;*
5. *si $(F \Rightarrow G) = (F' \Rightarrow G')$ alors $F = F'$ et $G = G'$;*
6. *si $(F \vee G) = (F' \vee G')$ alors $F = F'$ et $G = G'$;*
7. *si $(F \wedge G) = (F' \wedge G')$ alors $F = F'$ et $G = G'$;*

Ces propriétés sont l'analogue des deux propriétés : le successeur est non nul et l'injectivité du successeur, sur les entiers (voir paragraphe 3.1).

Si les formules sont vues comme des mots sur un alphabet (suites finies de lettres de l'alphabet), certaines de ces propriétés sont évidentes (1, 2, 4), les autres résultent de propriétés des parenthésages, et sont couramment admises, ce que l'on fera ici.

Il n'y a pas de notion de liaison en calcul propositionnelle. Une *théorie propositionnelle* est donc un ensemble arbitraire de formules propositionnelles.

4.3 Sémantique.

La sémantique Pour interpréter une formule du calcul propositionnel, il suffit de donner une valeur de vérité à ses constantes propositionnelles, pour calculer ensuite la valeur de vérité de la formule[10]. On appelle donc *valuation* une fonction de $\mathcal{P}$ dans $\{0,1\}$ (0 pour "Faux", 1 pour "Vrai").

On va donc formuler la définition de l'interprétation d'une formule propositionnelle à l'aide des valuations. Étant donnée une valuation v de $\mathcal{P}$ dans $\{0,1\}$, on définit l'interprétation induite sur les formules du calcul propositionnel, que l'on note $\overline{v}$ et que l'on notera ensuite (abusivement) v, par induction sur la définition des formules propositionnelles :

atome $\overline{v}(p) = v(p)$.

absurde $\overline{v}(\perp) = 0$.

négation $\overline{v}(\neg G) = 1$ ssi $\overline{v}(G) = 0$.

conjonction $\overline{v}(G \wedge H) = 1$ ssi $\overline{v}(G) = 1$ et $\overline{v}(H) = 1$.

disjonction $\overline{v}(G \vee H) = 0$ ssi $\overline{v}(G) = 1$ et $\overline{v}(H) = 1$.

implication $\overline{v}(G \Rightarrow H) = 0$ ssi $\overline{v}(G) = 1$ et $\overline{v}(H) = 0$.

De façon équivalente, on peut aussi définir $\overline{v}$ en utilisant les opérations usuelles "+" et "·" de $\mathbb{Z}/2\mathbb{Z}$:

négation $\overline{v}(\neg G) = 1 + \overline{v}(G)$.

conjonction $\overline{v}(G \wedge H) = \overline{v}(G) \cdot \overline{v}(H)$.

disjonction $\overline{v}(G \vee H) = \overline{v}(G) + \overline{v}(H) + \overline{v}(G) \cdot \overline{v}(H)$.

implication $\overline{v}(G \Rightarrow H) = 1 + \overline{v}(G) + \overline{v}(G) \cdot \overline{v}(H)$.

On voit que pour chaque connecteur n-aire la sémantique est définie par une fonction de $\{0,1\}^n$ dans $\{0,1\}$. On a déjà décrit ces fonctions, de deux façons différentes. On peut également les décrire par des tableaux de 2^n lignes :

G	$\neg G$
0	1
1	0

G	H	$G \wedge H$	$G \vee H$	$G \Rightarrow H$
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	1

Il faut se persuader que pour chacun de ces connecteurs, qui sont empruntés à la langue courante, la sémantique dérive bien de la logique intuitive (non spécifiquement mathématique), mais simplifiée drastiquement par le tiers-exclu (seulement deux valeurs de vérité). Cette sémantique est bien-sûr insuffisante pour analyser la logique de la langue naturelle, à laquelle elle ne correspond, en particulier dans le cas de l'implication, que dans des situations "binaires".[11]

On peut voir également les tableaux ci-dessus comme des fonctions de l'ensemble des valuations sur un ensemble fini de constantes propositionnelles dans $\{0,1\}$. De telles fonctions sont appellées *tables de vérité*. L'ensemble des tables de vérités sur les variables propositionnelles $p_1, \ldots, p_n$ est donc $\{0,1\}^{\{p_1,\ldots,p_n\}}$.

[10] Formellement, on peut également considérer que les constantes propositionnelles de $\mathcal{P}$ sont des constantes de prédicat à 0-arguments. L'ensemble des formules du calcul propositionnel est alors un sous-ensemble de l'ensemble des formules du premier ordre pour la signature qui énumère toutes les constantes de $\mathcal{P}$ (remarquez que dans ce cas les quantificateurs sont sans utilité). La sémantique est qui sera définie au chapitre 7 est alors celle choisie ici pour le calcul propositionnel. Une constante propositionnelle est soit vraie, soit fausse : formellement elle s'interprète dans une structure d'ensemble de base M par un sous ensemble de M^0, ensemble réduit à un élément, $\{\emptyset\}$, et n'a donc que deux interprétations possibles $\emptyset$ que l'on note 0, et $\{\emptyset\}$ que l'on note 1. Évidemment l'ensemble de base de la structure ne joue aucun rôle

[11] Par exemple la phrase « si vous avancez je tire » peut se transformer en « n'avancez pas ou je tire » clairement synonyme. Ce n'est pas le cas d'une autre implication comme, « si Sophie était plus grande, elle attraperait les confitures » (dont la sémantique devrait faire intervenir divers mondes possibles).

Il est clair que l'interprétation d'une formule F pour une valuation v ne dépend que des valeurs de v pour les constantes propositionnelles qui apparaissent effectivement dans F et qui sont donc en nombre fini.

Étant donnée une formule qui n'utilise que les constantes propositionnelles $p_1, \ldots, p_n$, la *table de vérité* de la formule F pour $p_1, \ldots, p_n$ est la fonction de l'ensemble des valuations sur $p_1, \ldots, p_n$, soit $\{0,1\}^{\{p_1,\ldots,p_n\}}$, dans $\{0,1\}$, qui à chaque valuation v associe $\overline{v}(F)$. On peut la représenter comme ci-dessus par un tableau de 2^n lignes. Par exemple voici un calcul de la table de vérité de la formule $p \Leftrightarrow q$ définie par $(p \Rightarrow q) \wedge (q \Rightarrow p)$ (pour chacune des 4 valuations v on utilise bien la définition du prolongement $\overline{v}$) :

p	q	$p \Rightarrow q$	$q \Rightarrow p$	$p \Leftrightarrow q$
0	0	1	1	1
0	1	1	0	0
1	0	0	1	0
1	1	1	1	1

La table de vérité d'une formule décrit entièrement la sémantique d'une formule propositionnelle, puisqu'elle décrit toutes les interprétations possibles. Voyons quelques définitions.

Une valuation v *satisfait* une formule F signifie que $\overline{v}(F) = 1$. Une formule propositionnelle F est *satisfaisable* s'il existe au moins une valuation qui la satisfait.Une *théorie propositionnelle est satisfaisable* s'il existe une valuation qui satisfait toutes les formules de la théorie.

Une formule propositionnelle F est une *tautologie* signifie que pour toute valuation v on a $\overline{v}(F) = 1$.

Deux formules propositionnelles F et G sont *équivalentes* ssi elles sont satisfaites par les mêmes valuations :

$$F \equiv G \text{ ssi pour toute valuation } v\ v(F) = v(G).$$

et l'on vérifie que :

$$F \equiv G \text{ ssi } F \Leftrightarrow G \text{ est une tautologie.}$$

en utilisant la table de vérité de $\Leftrightarrow$.

On remarque également (c'est juste une reformulation de ce qui précède) que deux formules sont équivalentes ssi elles ont les mêmes tables de vérité.

La substitution simultanée de formules du langage du calcul des prédicats de signature $\mathcal{S}$, soient $H_1, \ldots, H_n$, aux constantes propositionnelles $p_1, \ldots, p_n$ d'une formule propositionnelle F s'écrit $F[H_1/p_1, \ldots, H_n/p_n]$ et se définit par induction sur F sans difficultés (les accolades { et } indiquent la portée de la substitution quand il y a ambiguïté ; comme la substitution, elles ne font pas partie du langage) :

atome $p_i[H_1/p_1, \ldots, H_n/p_n] := H_i$, pour $p \notin \{p_1, \ldots, p_n\}$, $p[H_1/p_1, \ldots, H_n/p_n] := p$;

absurde $\perp[H_1/p_1, \ldots, H_n/p_n] := \perp$;

négation $\{\neg G\}[H_1/p_1, \ldots, H_n/p_n] := \neg\{G\}[H_1/p_1, \ldots, H_n/p_n]$;

conjonction $(G \wedge H)[H_1/p_1, \ldots, H_n/p_n] := (G[H_1/p_1, \ldots, H_n/p_n] \wedge H[H_1/p_1, \ldots, H_n/p_n])$;

disjonction $(G \vee H)[H_1/p_1, \ldots, H_n/p_n] := (G[H_1/p_1, \ldots, H_n/p_n] \vee H[H_1/p_1, \ldots, H_n/p_n])$;

implication $(G \Rightarrow H)[H_1/p_1, \ldots, H_n/p_n] := (G[H_1/p_1, \ldots, H_n/p_n] \Rightarrow H[H_1/p_1, \ldots, H_n/p_n])$;

La propriété de substitution énoncée ci-dessous permet de voir les tautologies propositionnelles commes des schémas de formules universellement valides du calcul des prédicats.

Proposition 4.2 (substitution) *Soit F et G des formules propositionnelles où n'apparaissent que les constantes propositionnelles $p_1, \ldots, p_n$. Soit $H_1, \ldots, H_n$ sont des formules du langage $\mathcal{S}$ dont les variables libres sont parmi $x_1, \ldots, x_k$.*

Si F est une tautologie, dans toute $\mathcal{S}$-structure la formule

$$\forall x_1 \dots \forall x_k \; F[H_1/p_1, \dots, H_n/p_n] \text{ est valide.}$$

Si $F \equiv G$ alors

$$F[H_1/p_1, \dots, H_n/p_n] \equiv G[H_1/p_1, \dots, H_n/p_n]$$

En particulier si H_1,…,H_n sont des formules du calcul propositionnel, si F est une tautologie la formule

$$F[H_1/p_1, \dots, H_n/p_n] \text{ est une tautologie.}$$

Démonstration. La deuxième partie de la proposition se déduit de la première en utilisant que $F \equiv G$ ssi $\forall x_1 \dots \forall x_k (F \Leftrightarrow G)$ est une tautologie.

On ne va pas exposer la preuve pour le calcul des prédicats puisque l'on a pas encore vu la définition formelle de la sémantique du calcul des prédicats mais elle très simple et est essentiellement la même que celle pour le calcul propositionnel. On prouve par induction sur F qu'étant donnée une valuation v telle que $v(H_1) = \alpha_1, \; \dots, \; v(H_n) = \alpha_n$, et α une valuation telle que $\alpha(p_1) = \alpha_1, \dots, \alpha(p_n) = \alpha_n$, on a :

$$v(F[H_1/p_1, \dots, H_n/p_n]) = \alpha(F)$$

Le résultat suit par définition de v et α pour les variables propositionnelles (toutes parmi p_i). Il suit de la définition de la sémantique pour chacun des connecteurs. ■

Cette propriété explique que l'on parle parfois de variable propositionnelle pour les constantes propositionnelles.

4.4 Exemples de formalisation en calcul propositionnel.

Il n'est pas très confortable de devoir formaliser en calcul propositionnel. Quand cela est possible on y gagne un langage beaucoup plus simple. En particulier on peut vérifier mécaniquement (mais pas de façon très efficace) la satisfaisabilité d'une formule propositionnelle.

4.4.1 Calcul propositionnel sur une structure.

Étant donné une $\mathcal{S}$-structure $\mathcal{M}$ on choisi comme constantes propositionnelles l'ensemble des formules atomiques de $\mathcal{M}$ pour le langage de signature $\mathcal{S}_\mathcal{M}$ *sans égalité*, où $\mathcal{S}_\mathcal{M}$ est la signature obtenue en ajoutant à $\mathcal{S}$ *chaque élément de l'ensemble de base de $\mathcal{M}$ comme symbole de constante.* Pour prendre un exemple très simple, pour la structure $(\{0,1\}, <)$, notre ensemble de constantes propositionnelles a 4 éléments, les formules atomiques du langage $(<, 0, 1)$:

$$0 < 0, \; 0 < 1, \; 1 < 0, \; 1 < 1$$

Le calcul propositionnel sur $\mathcal{M}$ est l'ensemble des formules propositionnelles construit sur ces constantes propositionnelles.

On peut exprimer par exemple que $(\mathbb{N}, \prec)$ est un ordre strict en utilisant l'ensemble infini de formules propositionnelles suivant :

$$O_s = \{m \prec n \Rightarrow n \prec p \Rightarrow m \prec p \; / \; m,n,p \in \mathbb{N}\} \cup \{\neg m \prec m \; / \; m,n \in \mathbb{N}\}$$

c'est à dire que $(\mathbb{N}, \prec)$ est un ordre strict ssi $(\mathbb{N}, \prec)$ satisfait O_s.

On peut exprimer de façon analogue $(\mathbb{N}, \preccurlyeq)$ est un ordre large :

$$\begin{aligned} O_l = \; & \{m \preccurlyeq n \Rightarrow n \preccurlyeq p \Rightarrow m \preccurlyeq p \; / \; m,n,p \in \mathbb{N}\} \cup \{m \preccurlyeq m \; / \; m \in \mathbb{N}\} \\ & \cup \{\neg(m \preccurlyeq n \wedge n \preccurlyeq m) \; / \; m,n \in \mathbb{N}, \; m \neq n\} \; . \end{aligned}$$

Bien entendu le calcul propositionnel est moins expressif que la logique du premier ordre, les formules ci-dessus n'ont la signification souhaitée que si l'ensemble de base est $\mathbb{N}$, et on n'a aucun moyen d'exprimer par exemple qu'il existe un plus petit élément pour l'ordre. Par contre on pourra exprimer que la structure finie $(\{0, \ldots, n\}, \preccurlyeq)$ (n étant un entier) a un plus petit élément :

$$\bigvee_{i=0}^{n} \bigwedge_{j=0}^{n} i \preccurlyeq j \ .$$

De façon générale on peut facilement exprimer grâce aux formules du calcul propositionnel sur la $\mathcal{S}$-structure $\mathcal{M}$:

- si $\mathcal{M}$ est fini, les propriétés de $\mathcal{M}$ exprimables au premier ordre (les quantifications universelles deviennent des conjonctions, les quantifications existentielles des disjonctions, les égalités vraies $\top$, les égalités fausses $\bot$).
- pour $\mathcal{M}$ quelconque, les propriétés universelles (des quantificateurs universels seulement en tête) exprimable au premier ordre de $\mathcal{M}$, en les remplaçant par une infinité de formules comme dans les exemples ci-dessus (on élimine ensuite l'égalité comme dans le cas où $\mathcal{M}$ est fini).

4.5 Déduction en calcul propositionnel.

On peut utiliser les deux notions de déduction étudiées (syntaxique et sémantique). Voyons les définitions en calcul propositionnel.

Si l'on restreint les règles de la déduction de la figure 5 page 18 aux règles propositionnelles (on omet les règles de l'égalité et les règles sur les quantificateurs), on obtient un système de déduction pour le calcul propositionnel, on notera de même $\Gamma \vdash_T C$, pour Γ un multi-ensemble de formules propositionnelles a pour conséquence C dans la théorie (propositionnelle) T. Les règles de la figure 5 page 18 sont les clauses d'une définition inductive. Formellement, on définit la déduction par induction. La relation $\Gamma \vdash_T C$ entre un multi-ensemble de formules propositionnelles Γ et une formule C est donc la relation dont la définition inductive est donnée par les règles de la figure 5 page 18, sauf les 4 règles qui concernent le quantificateurs, et par les règles de la figure 6 page 19. On remarque qu'une relation de déduction peut manifestement avoir des dérivations différentes. Il n'est donc pas question de définir une fonction par induction sur la définition d'une relation de déduction.

Si Γ est un ensemble de formules, on dira que $\Gamma \vdash_T C$, quand $\Gamma' \vdash_T C$ où Γ' est un multi-ensemble de formules qui contient les formules de Γ et seulement celles-ci.

La *déduction sémantique*, que l'on va noter temporairement $\Gamma \mid\!\sim_T C$, est définie pour Γ un ensemble de formules propositionnelles et C une formule propositionnelle par :

Si pour toute valuation v pour toute formule A de $\Gamma \cup T$ $v(A) = 1$, alors $v(C) = 1$.

Le système vérifie la *propriété d'adéquation*, à savoir que :

Proposition 4.3 (adéquation) *Pour toute théorie propositionnelle T, tout ensemble fini de formules propositionnelles Γ, toute formule propositionnelle C :*

$$\text{si } \Gamma \vdash_T C, \quad \text{alors } \Gamma \mid\!\sim_T C \ .$$

Démonstration. La preuve se fait par induction sur la définition de la déduction, et ne pose aucune difficulté. Détaillons quelques cas.

Axiomes de la déduction, axiomes de la théorie T : par définition de $\mid\!\sim_T$.

Affaiblissement et contraction : par définition de $\mid\!\sim_T$.

Élimination de l'implication : On suppose que $\Gamma \mid\!\sim_T A \to B$ et $\Gamma' \mid\!\sim_T A$ (hypothèses d'induction). Soit v une valuation telle que $v(X) = 1$ pour $X \in \Gamma \cup \Gamma'$. On a alors par hypothèse d'induction $v(A \to B) = 1$ et $v(A) = 1$. D'après la table de vérité de l'implication, $v(B) = 1$. On a bien montré $\Gamma, \Gamma' \mid\!\sim_T B$.

Introduction de l'implication : on suppose que $\Gamma, A \mid\!\sim_T B$ (hypothèse d'induction) et l'on veut montrer que $\Gamma \mid\!\sim_T A \Rightarrow B$. La relation $\Gamma, A \mid\!\sim_T B$ signifie que pour toute valuation v telle que $v(X) = 1$ si $X \in \Gamma$ et $v(A) = 1$ on a $v(B) = 1$.

Soit v une valuation telle que $v(X) = 1$ si $X \in \Gamma$. Deux cas sont possibles suivant que $v(A) = 1$ ou $v(A) = 0$.
Si $v(A) = 1$, alors $v(B) = 1$ par hypothèse d'induction d'ou $v(A \Rightarrow B) = 1$.
Si $v(A) = 0$, alors $v(A \Rightarrow B) = 1$.
Dans les deux cas on a le résultat voulu.

raisonnement par l'absurde : On suppose que $\Gamma, \neg A \mid\!\sim \perp$ (hypothèse d'induction). Soit v une valuation telle que $v(X) = 1$ pour $X \in \Gamma$. Comme $v(\perp) = 0$, l'hypothèse d'induction entraîne que l'on ne peut avoir de plus $v(\neg A) = 1$, c'est à dire que forcément $v(A) = 1$. On a bien montré $\Gamma \mid\!\sim_T A$.

Les autres règles se traitent de façon tout aussi élémentaire. ■

La déduction syntaxique nous donne donc un moyen (en prenant une théorie T vide) de montrer que des propositions sont des tautologies.

4.6 Quelques équivalences logiques et tautologies usuelles.

Dans la suite A, B et C désignent des formules quelconques. Voici quelques équivalences usuelles. Nous laissons au lecteur le soin de les vérifier en utilisant la déduction ou les tables de vérité (voir également le paragraphe 4.8).

4.6.1 Négation des connecteurs usuels.

$$\neg\neg A \equiv A$$

Lois de de Morgan :

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

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

$$\neg(A \Leftrightarrow B) \equiv (A \Leftrightarrow \neg B)$$
$$\neg(A \Leftrightarrow B) \equiv (\neg A \Leftrightarrow B)$$

4.6.2 Expression des connecteurs avec $\neg$, $\wedge$ et $\vee$.

$$\perp \equiv (A \wedge \neg A)$$
$$\top \equiv (A \vee \neg A)$$

$$(A \Rightarrow B) \equiv (\neg A \vee B)$$

$$(A \Leftrightarrow B) \equiv ((A \Rightarrow B) \wedge (B \Rightarrow A)) \equiv ((\neg A \vee B) \wedge (\neg B \vee A))$$
$$(A \Leftrightarrow B) \equiv ((A \wedge B) \vee (\neg A \wedge \neg B))$$

4.6.3 Propriétés de la disjonction et de la conjonction.

commutativité :

$$(A \wedge B) \equiv (B \wedge A)$$
$$(A \vee B) \equiv (B \vee A)$$

associativité :

$$(A \wedge (B \wedge C)) \equiv ((A \wedge B) \wedge C)$$
$$(A \vee (B \vee C)) \equiv ((A \vee B) \vee C)$$

distributivité :

$$(A \wedge (B \vee C)) \equiv ((A \wedge B) \vee (A \wedge C))$$
$$(A \vee (B \wedge C)) \equiv ((A \vee B) \wedge (A \vee C))$$

idempotence :

$$(A \wedge A) \equiv A$$
$$(A \vee A) \equiv A$$

absorption :

$$(A \wedge \bot) \equiv \bot$$
$$(A \wedge (A \vee B)) \equiv A$$
$$(A \vee \top) \equiv \top$$
$$(A \vee (A \wedge B)) \equiv A$$

neutre :

$$(A \wedge \top) \equiv A$$
$$(A \vee \bot) \equiv A$$

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

Remarque : ces deux dernières équivalences ainsi que celles d'absorption et d'idempotence permettent de simplifier les formes normales.

4.6.4 Propriétés de l'implication et de l'équivalence.

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

contraposée :

$$(A \Rightarrow B) \equiv (\neg B \Rightarrow \neg A)$$
$$(A \Leftrightarrow B) \equiv (\neg B \Leftrightarrow \neg A)$$

Distributivité à droite :

$$(A \Rightarrow (B \wedge C)) \equiv ((A \Rightarrow B) \wedge (A \Rightarrow C))$$
$$(A \Rightarrow (B \vee C)) \equiv ((A \Rightarrow B) \vee (A \Rightarrow C))$$

Pseudo-distributivité à gauche :

$$((A \wedge B) \Rightarrow C) \equiv ((A \Rightarrow C) \vee (B \Rightarrow C))$$
$$((A \vee B) \Rightarrow C) \equiv ((A \Rightarrow C) \wedge (B \Rightarrow C))$$

Remarque : il n'y a pas de propriété analogue à ces 4 dernières pour l'équivalence.

$\bot$ et $\top$:

$$(A \Rightarrow \bot) \equiv \neg A \quad (\bot \Rightarrow A) \equiv \top \quad (A \Rightarrow \top) \equiv \top \quad (\top \Rightarrow A) \equiv A$$

$$(A \Leftrightarrow \bot) \equiv \neg A \quad (A \Leftrightarrow \top) \equiv A$$

4.6.5 Encore quelques équivalences.

$$(\neg A \Rightarrow A) \equiv A \qquad ((A \Rightarrow B) \Rightarrow A) \equiv A \qquad ((A \Rightarrow B) \Rightarrow B) \equiv (A \vee B)$$

Beaucoup des équivalences vues ci dessus, commutativité, associativité, lois de de Morgan... sont utilisées couramment dans le raisonnement usuel, en plus des règles de la déduction naturelle.

4.7 Formes normales.

4.7.1 Définitions.

On appelle *littéral* une constante propositionnelle ou une négation de constante propositionnelle.

Une formule est sous *forme normale disjonctive* (on dit parfois forme normale disjonctive-conjonctive) si elle s'écrit comme une disjonction de conjonctions de littéraux. Une formule est sous *forme normale conjonctive* (on dit parfois forme normale conjonctive-disjonctive) si elle s'écrit comme une conjonction de disjonctions de littéraux.

Donnons quelques exemples. Supposons que p, q, r soient des constantes propositionnelles, alors $p, q, r, \neg p, \neg q, \neg r$ sont des littéraux.

La formule $(p \wedge q \wedge r) \vee (\neg p \wedge r) \vee \neg r$ est sous forme normale disjonctive (la troisième conjonction est réduite à un élément).

La formule $(p \vee \neg q) \wedge \neg p \wedge (p \vee q \vee \neg r)$ est sous forme normale conjonctive (la deuxième disjonction est réduite à un élément).

La formule $p \vee \neg q$ est à la fois sous forme normale disjonctive (deux conjonctions réduites à un élément) et disjonctive (une conjonction réduite à une disjonction de deux littéraux), de même la formule $p \wedge q \wedge \neg r$.

4.7.2 Table de vérité et formes normales.

Voyons tout d'abord l'idée de la démonstration sur un exemple. Il s'agit tout simplement de décrire la table de vérité ligne par ligne, on obtient naturellement ainsi une forme normale disjonctive ou conjonctive suivant la façon dont on procède. Prenons la table de vérité de l'équivalence :

p	q	$p \Leftrightarrow q$
0	0	1
0	1	0
1	0	0
1	1	1

On peut la décrire de deux façons. En énumérant les valuations qui rendent la formule vraie :

$$\begin{array}{ll} v(p \Leftrightarrow q) = 1 & \text{ssi } [v(p) = 0 \text{ et } v(q) = 0] \text{ ou } [v(p) = 1 \text{ et } v(q) = 1] \\ & \text{ssi } [v(\neg p) = 1 \text{ et } v(\neg q) = 1] \text{ ou } [v(p) = 1 \text{ et } v(q) = 1] \\ & \text{ssi } v(\neg p \wedge \neg q) = 1 \text{ ou } (v(p \wedge q) = 1 \\ & \text{ssi } v(\ (\neg p \wedge \neg q) \vee (p \wedge q)\ ) = 1 \end{array}$$

on obtient une forme normale disjonctive de $p \Leftrightarrow q$ qui est $(\neg p \wedge \neg q) \vee (p \wedge q)$.

En énumérant les valuations qui rendent la formule fausse :

$$\begin{array}{ll} v(p \Leftrightarrow q) = 0 & \text{ssi } [v(p) = 0 \text{ et } v(q) = 1] \text{ ou } [v(p) = 1 \text{ et } v(q) = 0] \\ & \text{ssi } [v(p) = 0 \text{ et } v(\neg q) = 0] \text{ ou } [v(\neg p) = 0 \text{ et } v(q) = 0] \\ & \text{ssi } v(p \vee \neg q) = 0 \text{ ou } (v(\neg p \vee q) = 0 \\ & \text{ssi } v(\ (p \vee \neg q) \wedge (\neg p \vee q)\ ) = 0 \end{array}$$

on obtient une forme normale conjonctive de $p \Leftrightarrow q$ qui est $(p \vee \neg q) \wedge (\neg p \vee q)$.

On montre maintenant que ce procédé fonctionne pour n'importe quelle table de vérité.

Les deux propriétés suivantes sont des conséquences immédiates de la définition de l'interprétation des conjonctions et disjonctions :

Lemme 4.4 *Soient $F_1, \ldots, F_n$ des formules propositionnelles. On a :*

$$\begin{array}{lll} v(F_1 \wedge \ldots \wedge F_n) = 1 & ssi & v(F_1) = 1 \text{ et } \ldots \text{ et } v(F_n) = 1 \\ v(F_1 \vee \ldots \vee F_n) = 0 & ssi & v(F_1) = 0 \text{ et } \ldots \text{ et } v(F_n) = 0 \end{array}$$

On utilise ces propriétés pour le lemme suivant :

Lemme 4.5 *Soit a une valuation à support fini sur n constantes propositionnelles $p_1, \ldots, p_n$. Il existe des formules F^1 et F^0 telles que pour toute valuation v :*

F^1 est une conjonction de littéraux et $v(F^1) = 1$ ssi pour tout $i \in \{1, \ldots, n\}$ $v(p_i) = a(p_i)$
F^0 est une disjonction de littéraux et $v(F^1) = 0$ ssi pour tout $i \in \{1, \ldots, n\}$ $v(p_i) = a(p_i)$

Démonstration. On pose $p_i^1 = p_i$ si $a(p_i) = 1$, $p_i^1 = \neg p_i$ si $a(p_i) = 0$. On pose $F^1 = p_1^1 \wedge \ldots \wedge p_n^1$. On a bien que $a(p_i^1) = 1$ pour tout i, donc $a(F^1) = 1$.

On a que $v(F^1) = 1$ ssi $v(p_i^1) = 1$ pour chaque $i \in \{1, \ldots, n\}$, d'où le résultat.

De même en posant $p_i^0 = p_i$ si $a(p_i) = 0$, $p_i^0 = \neg p_i$ si $a(p_i) = 1$, et$F^0 = p_1^0 \vee \ldots \vee p_n^0$, on a bien, $a(F^0) = 0$ et $v(F^0) = 0$ ssi $v(p_i^0) = 0$ pour chaque $i \in \{1, \ldots, n\}$, d'où le résultat. ∎

Pour en déduire le théorème de mise sous forme normale, on utilise maintenant les propriétés duales du lemme 4.5 :

Lemme 4.6 *Soient $F_1, \ldots, F_n$ des formules propositionnelles. On a :*

$$\begin{array}{lll} v(F_1 \vee \ldots \vee F_n) = 1 & ssi & v(F_1) = 1 \text{ ou } \ldots \text{ ou } v(F_n) = 1 \\ v(F_1 \wedge \ldots \wedge F_n) = 0 & ssi & v(F_1) = 0 \text{ ou } \ldots \text{ ou } v(F_n) = 0 \end{array}$$

Théorème 4.7 *Toute table de vérité, est la table de vérité d'une formule. Qui plus est on peut choisir cette formule sous forme normale conjonctive, ou sous forme normale disjonctive.*

Plus précisément, pour tout ensemble de n constantes propositionnelles $\{p_1, \ldots, p_n\}$, toute fonction de $\{0, 1\}^{\{p_1, \ldots, p_n\}}$ dans $\{0, 1\}$, il existe une formule sous forme normale conjonctive et une formule sous forme normale disjonctive dont c'est la table de vérité.

Démonstration. Soit t une table de vérité sur $p_1, \ldots, p_n$. Soit $v_0, \ldots, v_q$ toutes les valuations dont l'image est 1 par t. A chacune de ces valuations v_i on associe la formule F_i^1 donnée par le lemme 4.5. La formule ${F_0}^1 \vee \ldots \vee {F_q}^1$ est sous forme normale disjonctive et répond à la question. En effet $v({F_0}^1 \vee \ldots \vee {F_q}^1) = 1$ ssi $v({F_0}^1) = 1$ ou ...ou $v({F_q}^1) = 1$, et donc ssi v est l'une des valuation $v_0, \ldots, v_q$ d'après le lemme 4.5.

De façon analogue si $u_0, \ldots, u_r$ sont les valuations dont l'image est 0 par t, on associe à chacune d'entre elles la formule F_i^0 donnée au lemme 4.5. La formule $F_0^0 \wedge \ldots \wedge F_q^0$ est sous forme normale conjonctive et répond à la question, car $v(F_0^0 \wedge \ldots \wedge F_q^0) = 0$ ssi $v(F_0^0) = 0$ ou ...ou $v(F_0^0) = 0$. ∎

Là encore on s'aperçoit que l'on a essentiellement utilisé les propriétés de la conjonction et de la disjonction dans le meta-langage. C'est à dire que l'on peut décrire une table de vérité à l'aide uniquement de conjonctions de disjonctions et de négations en décrivant toutes les valuations qui prennent la valeur "vrai" (forme normale disjonctive) ou la valeur "faux" (forme normale conjonctive).

La forme normale que l'on trouve à la lecture de la table de vérité, en suivant la méthode indiquée par la preuve du théorème précédent, n'est en général pas la plus courte ! Ainsi $p \wedge q$ est sous forme normale conjonctive et disjonctive. La lecture de la table de vérité donnera bien $p \wedge q$ comme forme normale disjonctive, mais $(p \vee q) \wedge (p \vee \neg q) \wedge (\neg p \vee q)$ comme forme normale conjonctive.

4.7.3 Systèmes complets de connecteurs.

On dit qu'un système de connecteurs est *fonctionnellement complet*, ou parfois plus simplement *complet* si toute table de vérité est représentée par une formule utilisant ces seuls connecteurs. Un corollaire immédiat du théorème de mise sous-forme normale est que :

Corollaire 4.8 *Le système de connecteurs* $\{\neg, \wedge, \vee\}$ *est fonctionnellement complet.*

Des diverses équivalences parmi celles du paragraphe 4.6 on déduit que :

Corollaire 4.9 *Les systèmes de connecteurs* $\{\neg, \wedge\}$, $\{\neg, \vee\}$, $\{\neg, \Rightarrow\}$, $\{\perp, \Rightarrow\}$ *sont fonctionnellement complets.*

Démonstration. Pour chaque système S il suffit de montrer que chacun des connecteurs d'un système complet connu s'exprime avec les connecteurs du système S. Pour le moment le seul système complet connu est $\{\neg, \wedge, \vee\}$.

Le système $\{\neg, \wedge\}$ est complet car $A \vee B \equiv \neg(\neg A \wedge \neg B)$ (loi de de Morgan).

Le système $\{\neg, \vee\}$ est complet car $A \wedge B \equiv \neg(\neg A \vee \neg B)$ (loi de de Morgan).

Le système $\{\neg, \Rightarrow\}$ est complet car $A \vee B \equiv \neg A \Rightarrow B$ et $\{\neg, \vee\}$ est complet.

Le système $\{\perp, \Rightarrow\}$ est complet car $\neg A \equiv A \Rightarrow \perp$ et $\{\neg, \Rightarrow\}$ est complet. ■

4.7.4 Mise sous forme normale.

Étant donnée une formule, la mettre sous forme normale conjonctive, resp. disjonctive, consiste à trouver une formule équivalente sous forme normale conjonctive, resp. disjonctive.

Une première méthode est de chercher la table de vérité puis d'en déduire la forme normale cherchée comme dans la preuve du théorème 4.7.

Une seconde méthode un peu plus directe est d'utiliser convenablement les équivalences logiques des paragraphes 4.6.2, 4.6.1 et 4.6.3.

Ces méthodes ne sont pas très efficaces algorithmiquement, et l'on n'en connait pas qui le soit vraiment. Pratiquement on a souvent besoin seulement d'une forme normale non pas équivalente mais *equisatisfaisable* à une formule donnée (l'une est satisfaite ssi l'autre l'est). Là il existe des algorithmes simples et efficaces que l'on ne décrira pas ici.

4.8 Quelques méthodes pour tester les tautologies, la satisfaisabilité etc.

4.8.1 Un peu de complexité.

Donnons de façon très informelle une idée de la complexité des problèmes rencontrés dans ce chapitre, c'est à dire du temps de calcul théorique pour résoudre ces problèmes.

Le problème le plus simple que nous ayons à résoudre est celui de la satisfaction d'une formule donnée par une valuation donnée. Il s'agit d'un calcul booléen (c'est à dire sur $\mathbb{Z}/2\mathbb{Z}$), comme nous l'avons vu lors de la définition de la satisfaction paragraphe 4.3. Ce calcul très simple se fait en temps linéaire en la taille de la formule. On dit qu'un tel problème est P (pour polynomial).

Un problème nettement plus ardu est celui de savoir si une formule donnée est satisfaisable. Une solution (pas très efficace) serait de calculer la table de vérité de la formule. Si la formule possède n variables propositionnelles, il faut donc procéder à 2^n calculs (eux-mêmes) linéaires. Un tel calcul est exponentiel en n et donc (on s'en convainc facilement) en la taille de la formule. Une question naturelle est de savoir s'il existe des méthodes vraiment plus efficaces, si ce n'est linéaire au moins polynomiale en la taille de la formule. C'est un problème posé depuis les années 60 et auquel on ne sait toujours pas répondre actuellement.

Pour essayer d'être un peu plus précis : un problème tel que celui de la satisfaisabilité à ceci de particulier que si un "oracle" fournissait la bonne valuation à vérifier (s'il y en a une c'est celle-ci), le problème se résoudrait en temps polynomial. Un tel problème est dit NP, pour polynomial non-déterministe. Sans entrer dans le détail une autre façon de caractériser ces problèmes est de dire qu'ils se résolvent en temps polynomial quand on remplace la notion traditionnel de calcul qui est *déterministe* – l'état de la machine et l'avancement du calcul étant connus, il n'y a qu'une possibilité pour l'étape suivante – par une relation non-déterministe – dans les mêmes conditions il y a maintenant plusieurs possibilités. Le calcul non-déterministe est une notion toute théorique.

On montre que le problème de la satisfaction d'une formule est NP, et que de plus il peut coder de façon polynomiale tous les problèmes NP, c'est à dire que s'il était polynomial, tous les problèmes NP le seraient.

La conjecture la plus courante est que $\mathsf{P} \neq \mathsf{NP}$, mais le problème reste ouvert.

Un autre problème est celui de savoir si une formule donnée est ou non une tautologie. On sait que F est une tautologie ssi $\neg F$ n'est pas satisfaisable. Cela indique que le problème de la tautologie pour F est le complémentaire du problème de la satisfaisabilité pour $\neg F$ qui est NP. On dit qu'un tel problème est co-NP. Évidemment si les problèmes NP étaient polynomiaux, les problèmes co-NP le seraient également.

Nous n'irons pas plus loin dans cette voie. La conclusion est que, dans l'état actuel des choses, il ne faut pas s'attendre à des algorithmes très efficaces pour déterminer si une formule est satisfaisable ou si c'est une tautologie. On peut cependant essayer de donner quelques méthodes un peu plus efficaces dans certains cas que le simple calcul de la table de vérité. Voyons en rapidement quelques unes, présentées sur des exemples.

4.8.2 La déduction.

Une première méthode à ne pas négliger est la déduction qui permet dans les cas les plus simples de se persuader rapidement qu'une formule est une tautologie, même si il peut être assez long de rédiger formellement la preuve. En effet on sait d'après le propriété d'adéquation 4.3 que si $\vdash F$ est prouvable, F est une tautologie.

Par exemple les propriétés d'associativité ou de distributivité entre conjonction et disjonction sont évidentes par déduction.

4.8.3 Satisfaction, Réfutation.

La table de vérité de vérité d'une formule permet à la fois de savoir si celle-ci est une tautologie et si elle est satisfaisable. Si l'on rompt la symétrie, en s'intéressant seulement à l'un des deux problèmes, on peut espérer arriver plus rapidement au résultat.

Pour savoir si une formule est satisfaisable, il peut être plus rapide d'analyser celle-ci, en cherchant à quelle condition elle est satisfaite, plutôt que d'énumérer toutes les valuations jusqu'à en trouver une convenable. Par exemple supposons que l'on souhaite savoir si $\neg((A \Rightarrow B) \Rightarrow B) \Rightarrow B$ est satisfaisable. On écrit :

$$\begin{array}{ll} & v(\neg(((A \Rightarrow B) \Rightarrow B) \Rightarrow B)) = 1 \\ \text{ssi} & v(((A \Rightarrow B) \Rightarrow B) \Rightarrow B) = 0 \\ \text{ssi} & v((A \Rightarrow B) \Rightarrow B) = 1 \text{ et } v(B) = 0 \\ \text{ssi} & v(A \Rightarrow B) = 0 \text{ et } v(B) = 0 \\ \text{ssi} & v(A) = 1 \text{ et } v(B) = 0 \end{array}$$

et on a trouvé une valuation qui satisfait la formule.

Cette méthode permet également de savoir si une formule est une tautologie, et prend alors le nom de *méthode de réfutation*. En effet il suffit de savoir si la négation d'une formule F donnée est ou non satisfaisable. L'exemple ci-contre nous permet par exemple de conclure que $((A \Rightarrow B) \Rightarrow B) \Rightarrow B$ n'est pas une tautologie.

Montrons par cette méthode que la loi de Peirce $((A \Rightarrow B) \Rightarrow A) \Rightarrow A$ est une tautologie.

$$\begin{array}{ll} & v(((A \Rightarrow B) \Rightarrow A) \Rightarrow A) = 0 \\ \text{ssi} & v((A \Rightarrow B) \Rightarrow A) = 1 \text{ et } v(A) = 0 \\ \text{ssi} & v(A \Rightarrow B) = 0 \text{ et } v(A) = 0 \\ \text{ssi} & v(A) = 1 \text{ et } v(B) = 0 \text{ et } v(A) = 0 \end{array}$$

il n'existe donc pas de valuation qui satisfait la loi de Peirce, c'est bien une tautologie.

On ne donne pas de présentation systématique de cette méthode, qui a d'ailleurs plusieurs variantes. Une façon synthétique de procéder est de la présenter sous forme d'arbre, un embranchement correspond à l'étude de plusieurs possibilités. Dans le premier des deux exemples précédents on est amené à :

$$\begin{array}{rl} & v((A \Rightarrow B) \Rightarrow B) = 1 \\ \text{ssi} & v(A \Rightarrow B) = 0 \text{ ou } v(B) = 1 \end{array}$$

et on a éliminé immédiatement la possibilité $v(B) = 1$ car par ailleurs on sait qu'il faut $v(B) = 0$, ce qui a permis de simplifier la présentation pour cet exemple. Bien sûr ce n'est pas toujours possible et on peut avoir plusieurs possibilités à explorer. Une formalisation de cette méthode est connue sous le nom de *méthode des tableaux*.

4.8.4 Tables de vérité réduites.

On peut tirer partie de ce que dans certains cas, la connaissance de la valeur de vérité d'une sous-formule peut permettre de déterminer la valeur de la formule. Par exemple si $v(B) = 1$, on sait sans connaître $v(A)$ que $v(A \Rightarrow B) = 1$. On peut donc éviter d'énumérer tous les cas (cependant il devient moins évident que tous les cas sont énumérés et cela doit être vérifié). Voici un exemple.

A	B	C	$A \wedge B$	$(A \wedge B) \Rightarrow C$	$B \Rightarrow C$	$A \Rightarrow (B \Rightarrow C)$
?	?	1	?	1	1	1
0	?	0	0	1	?	1
1	0	0	0	1	1	1
1	1	0	1	0	0	0

On vérifie facilement que tous les cas sont bien énumérérés (le "?" désigne 0 ou 1). On a donc montré que $(A \wedge B) \Rightarrow C \equiv A \Rightarrow (B \Rightarrow C)$.

On peut également introduire des variables et utiliser les propriétés d'élément neutre de 0 et 1 en calcul booléen. Ainsi, en notant (abusivement) l'opération booléenne associée au connecteur comme le connecteur lui-même on obtient

A	B	C	$B \vee C$	$(A \wedge (B \vee C)$	$A \wedge B$	$A \wedge C$	$(A \wedge B) \vee (A \wedge C)$
0	?	?	?	0	0	0	0
1	b	c	$b \vee c$	$b \vee c$	b	c	$b \vee c$

5 Complétude du calcul propositionnel.

Nous aurons besoin de quelques propriétés de la relation de déduction que nous n'avions pas enocre énoncé et qui seront tout aussi valides en calcul des prédicats.

5.1 Quelques préliminaires sur la relation de déduction.

Voyons quelques propriétés de la relation de déduction (voir chapitre 2).

Proposition 5.1 *Pour toutes théories T et T', toute formule C,*

$$\textit{si}\ \ T \subset T',\ \ \textit{et}\ \ \Gamma \vdash_T C \ \ \textit{alors}\ \ \Gamma \vdash_{T'} C\ .$$

Démonstration. C'est une conséquence immédiate de la définition, la théorie n'intervenant que dans les axiomes (formellement c'est une induction). ■

Proposition 5.2 *Pour toute théorie T, toute formule A, toute formule C,*

$$\Gamma \vdash_{T\cup\{A\}} C,\ \ \textit{ssi}\ \ \Gamma, A \vdash_T C\ .$$

Démonstration. Supposons $\Gamma \vdash_{T\cup\{A\}} C$. Si cette relation a été dérivée sans utiliser l'axiome $\vdash A$, on a $\Gamma \vdash_T C$ et donc par affaiblissement $\Gamma, A \vdash_T C$. Si la relation a été dérivée en utilisant une ou plusieurs fois $\vdash A$, en remplaçant cet axiome par $A \vdash A$ on obtient une dérivation de la relation $\Gamma, \underbrace{A, \ldots, A}_{n} \vdash C$, où l'entier n vérifie $n \geqslant 1$. En utilisant la contraction si $n > 1$ on obtient $\Gamma, A \vdash C$ (remarquez que formellement, il faut faire un raisonnement par induction).

Réciproquement, supposons $\Gamma, A \vdash_T C$, alors $\Gamma \vdash_T A \Rightarrow C$ (introduction), d'où $\Gamma \vdash_{T\cup\{A\}} A \Rightarrow C$, or $\vdash_{T\cup\{A\}} A$, donc (élimination) $\Gamma \vdash_{T\cup\{A\}} C$. ■

5.2 Énoncés du théorème de complétude.

Le *théorème de complétude* du calcul propositionnel est la réciproque de la propriété d'adéquation 4.3 page 30.

Théorème 5.3 (complétude) *Pour toute théorie T, tout ensemble fini de formules closes Γ, toute formule close C dans le langage L,*

$$\textit{Si}\ \Gamma \models_T C \quad \textit{alors}\ \ \Gamma \vdash_T C\ .$$

Donnons tout d'abord quelques énoncés équivalents. Tout d'abord remarquons que les deux cas particuliers suivants sont équivalents au cas général.

Corollaire 5.4 (complétude) *Pour toute théorie T, toute formule close C dans le langage L,*

$$\textit{Si}\ \ \models_T C \quad \textit{alors}\ \ \vdash_T C\ .$$

Corollaire 5.5 (complétude) *Dans toute théorie T, tout ensemble fini de formules closes Γ dans le langage L*

$$\textit{Si}\ \ \models_T \bot \quad \textit{alors}\ \ \vdash_T \bot\ ,$$

ou encore par contraposée :

$$\textit{Si}\ \ \nvdash_T \bot\ \textit{alors}\ T\ \textit{est satisfaisable}$$

Il s'agit des cas particuliers du théorème précédent où Γ est vide, puis où $C = \bot$.

Proposition 5.6 *On peut déduire le théorème de complétude 5.3 des corollaires 5.4 et 5.5.*

Démonstration (cor 5.4 $\Rightarrow$ complétude). Si l'on pose $\Gamma = \{A_1, \ldots, A_n\}$. On remarque tout d'abord que, vu la définition de la déduction sémantique "$\mid\!\sim$" :

$$\text{Si } \Gamma \mid\!\sim_T C \quad \text{alors} \quad \mid\!\sim_{T \cup \Gamma} C$$

On a donc $\vdash_{T \cup \Gamma} C$ d'après le corollaire 5.4, et on en déduit $\Gamma \vdash_T C$ par récurrence sur la longueur de Γ en appliquant la proposition 5.2. ∎

Démonstration (cor 5.5 $\Rightarrow$ cor 5.4). On vérifie que

$$\text{Si } \mid\!\sim_T C \quad \text{alors} \quad \mid\!\sim_{\{T \cup \{\neg C\}} \perp .$$

par définition de l'interprétation : si une valuation v satisfait T, elle satisfait C donc ne satisfait pas $\neg C$.

On déduit ensuite du corollaire que $\vdash_{\{T \cup \{\neg C\}} \perp$, puis que $\neg C \vdash_T \perp$ d'après la proposition 5.2. D'après la règle du raisonnement par l'absurde (voir figure 5 page 18) on a $\Gamma \vdash_T C$. ∎

5.3 Le théorème de compacité.

Avant de démontrer ce théorème voyons en une conséquence. La définition de la déduction a pour conséquence immédiate qu'une formule utilise dans sa démonstration un nombre fini d'axiomes de la théorie :

Proposition 5.7 (finitude) *Si $\vdash_T C$, il existe une partie finie de T, soit T_0, telle que $\vdash_{T_0} C$.*

Appliquée à la déduction sémantique la proposition de finitude devient beaucoup moins immédiate. Énonçons la dans le cas $C = \perp$.

Proposition 5.8 (compacité) *Si une théorie T n'est pas satisfaisable, alors l'une de ses parties finies n'est pas satisfaisable (et réciproquement).*

ou encore par contraposée

Une théorie propositionnelle T est satisfaisable si (et seulement si) toutes ses parties finies sont satisfaisables.

Ce théorème prend le nom de *théorème de compacité*. Voyons pourquoi[12]. Soit $\mathcal{P}$ l'ensemble des variables propositionnelles. L'ensemble des valuations est $\{0,1\}^{\mathcal{P}}$. On peut munir $\{0,1\}$ de la topologie discrète (chauqe sous-ensemble est un ouvert, donc un fermé). On muni alors $\{0,1\}^{\mathcal{P}}$ de la topologie produit. Les ouverts élémentaires de $\{0,1\}^{\mathcal{P}}$ sont donc les produits de sous-ensembles de $\{0,1\}$ dont tous, sauf un nombre fini, sont égaux à $\{0,1\}$. Il s'agit en quelque sorte de valuations partielles à support fini, un ouvert élémentaire est défini par une valuation a sur k constantes propositionnelles $p_1, \ldots, p_k$:

$$O_a = \{v \in \{0,1\}^{\mathcal{P}} \ / \ \forall p \in \{p_1, \ldots, p_k\} \ v(p) = a(p)\}$$

Comme une réunion d'ouverts est un ouvert, le complémentaire d'un tel ensemble est un ouvert, et donc ces ouverts élémentaires sont égalements des fermés de $\{0,1\}^{\mathcal{P}}$.

A chaque formule A on peut faire correspondre l'ensemble des valuations qui la satisfont. Comme on l'a remarqué au lemme 4.5 page 34, on peut associer à a une formule, que nous appelons maintenant A_a qui est une conjonction de littéraux est telle que :

$$O_a = \{v \in \{0,1\}^{\mathcal{P}} \ / \ v(A_a) = 1\}$$

A une formule quelconque X sur $p_1, \ldots, p_k$ on associe sa forme normale disjonctive, et donc une réunion finie d'ouverts-fermés élémentaires qui est exactement :

$$O_X = \{v \in \{0,1\}^{\mathcal{P}} \ / \ v(X) = 1\} \ .$$

[12]la lecture de la fin de ce paragraphe utilise un peu de topologie, elle n'est pas nécessaire pour la suite.

et qui est donc un ouvert-fermé.

Dire qu'une théorie T est satisfaisable c'est dire que :

$$\bigcap_{X \in T} O_X \neq \emptyset$$

Le théorème de compacité affirme donc que toute famille de fermés de la forme O_X d'intersection non vide a une sous-partie finie d'intersection non vide. C'est bien un résultat de compacité. On peut le déduire ce que $\{0,1\}^{\mathcal{P}}$ est un espace compact, ce qui est une conséquence du théorème de Tychonoff qui énonce que tout produit d'espaces compacts est compact (le théorème de Tychonoff nécessite l'axiome du choix).

5.4 La preuve du théorème de complétude.

5.4.1 Quelques préliminaires.

On ne va faire la preuve du théorème de complétude que pour une théorie dénombrable, donc sur un ensemble $\mathcal{P}$ de constantes propositionnelles dénombrables. On montre alors que :

Proposition 5.9 *L'ensemble des formules du calcul propositionel sur un ensemble dénombrable $\mathcal{P}$ est dénombrable.*

Démonstration. L'écriture d'une formule propositionnelle est une suite finie de symboles de $\mathcal{P} \cup \{\perp, \Rightarrow, \neg, \wedge, \vee,), (\}$. On conclut avec la proposition suivante. ∎

proposition. *L'ensemble des suites finies d'éléments d'un ensemble dénombrable est dénombrable.*

Démonstration. On verra ce résultat en théorie des ensembles. Voyons une justification rapide. On peut en effet énumérer les suites finies d'entiers, en commençant par la suite vide, puis les suite de longueur 1 d'entiers plus petits ou égaux à 1, puis les suites de longueur $\leqslant 2$ d'entiers $\leqslant 2$ non encore énumérées etc. À l'étape n on énumère les suite de longueur $\leqslant n$ d'entiers $\leqslant n$ qui n'ont pas encore été énumérées. A chaque étape on énumère un nombre fini de suites finies. ∎

On peut donc supposer l'existence d'une énumération des formules du calcul propositionnel, $\{F_i \;/\; i \in \mathbb{N}\}$.

5.4.2 Plan de la preuve.

On va démontrer directement le corollaire 5.5, dont on a vu qu'il était équivalent au théorème de complétude 5.3. La démarche sera la même en calcul des prédicats. Elle est la suivante. On rappelle que l'on a une théorie T telle que $\nvdash_T \perp$, et que l'on veut construire une valuation qui valide chaque formule de T.

1. étant donnée une théorie non contradictoire T l'*étendre*, en utilisant la déduction, en une théorie T^s :
$$T \subset T^s$$
qui doit être *complète*[13], c'est à dire telle que
$$\text{pour toute formule close } \; C\,, C \in T^s \text{ ou } \neg C \in T^s\ .$$
qui doit être *non contradictoire* c'est à dire telle que :
$$\nvdash_{T^s} \perp\ .$$
et qui doit être *saturée*, c'est à dire telle que :
$$\text{pour toute formule propositionnelle } \; C\ (\vdash_{T^s} C \Rightarrow C \in T^s)\ .$$
2. Construire une valuation qui valide la théorie T à partir de la théorie T^s.

[13]habituellement une théorie Θ complète vérifie seulement que pour toute formule C, $\vdash_\Theta C$ ou $\vdash_\Theta \neg C$. Pour une théorie saturée cela revient bien-sûr à la formulation adoptée ici.

5.4.3 Une théorie complète et saturée T^s.

On se donne une énumération des formules propositionnelles, soit $(F_i)_{i\in\mathbb{N}}$.

Supposons T non contradictoire : $\nvdash_T \perp$. On va construire une suite de théories $(T_n)_{n\in\mathbb{N}}$ par récurrence sur $\mathbb{N}$.

i. $T_0 = T$

ii. Si $\vdash_{T_n} F_n$, alors $T_{n+1} = T_n \cup \{F_n\}$, sinon $T_{n+1} = T_n \cup \{\neg F_n\}$.

On pose ensuite $T^s = \bigcup_{i\in\mathbb{N}} T_n$.

Lemme 5.10 *La théorie T^s est complète et contient T.*

Démonstration. La théorie T^s contient $T = T_0$ par construction. Elle est complète car pour tout entier n, $F_n \in T_{n+1}$ ou $\neg F_{n+1} \in T_{n+1}$ par construction des T_n. Or pour tout entier n, $T^s \supset T_n$ d'où le résultat (axiome de la théorie). ■

Lemme 5.11 *La théorie T^s n'est pas contradictoire : $\nvdash_{T^s} \perp$.*

Démonstration. En vertu de la finitude 5.7 page 39, si T^s était contradictoire, l'une de ses parties finies le serait. Toute partie finie de T^s est inclue dans T_n pour un entier n suffisamment grand. Il suffit donc de montrer que pour tout entier n, T_n n'est pas contradictoire, ce que l'on fait par récurrence sur n.

$n = 0$. $T_0 = T$ n'est pas contradictoire par hypothèse.

$n \to n+1$. On procède par contraposée. Supposons que T_{n+1} est contradictoire, c'est à dire $\vdash_{T_{n+1}} \perp$, et montrons qu'alors T_n l'est. Par construction de T_{n+1}, deux cas se présentent :

$T_{n+1} = T_n \cup \{F_n\}$. Cela signifie que $\vdash_{T_n} F_n$. On a $\vdash_{T_n\cup\{F_n\}} \perp$, donc $F_n \vdash_{T_n} \perp$ (proposition 5.2 page 38), donc $\vdash_{T_n} \neg F_n$ (introduction). Or $T_n \vdash F_n$, donc $T_n \vdash \perp$ (élimination), c'est à dire T_n contradictoire.

$T_{n+1} = T_n \cup \{\neg F_n\}$. Cela signifie que $\nvdash_{T_n} F_n$. Or on a $\vdash_{T_n\cup\{\neg F_n\}} \perp$, donc $\neg F_n \vdash_{T_n} \perp$ (proposition 5.2 page 38), donc $\vdash_{T_n} F_n$ (raisonnement par l'absurde). Mais dans ce cas $T_{n+1} = T_n \cup \{F_n\}$. Ce cas ne peut donc se produire. ■

Lemme 5.12 *La théorie T^s est saturée.*

Démonstration. Supposons $\vdash_{T^s} C$. Comme T^s est complète $C \in T^s$ ou $\neg C \in T^s$. Si $\neg C \in T^s$, $\vdash_{T^s} \neg C$, et donc $\vdash_{T^s} \perp$ (élimination), or T^s n'est pas contradictoire (lemme 5.11). On a donc bien $C \in T^s$. ■

5.4.4 Une valuation validant T.

On définit la valuation v, pour toute formule atomique (close) p différente de $\perp$:

$$\text{si } \ p \in T^s \ \text{ alors } \ v(p) = 1 \ \text{ sinon } \ v(p) = 0 \ .$$

On montre par induction sur la structure des formules propositionnelles le lemme suivant.

Lemme 5.13 *Pour toute formule C :*

$$\textit{si } \ C \in T^s \ \textit{ alors } \ v(C) = 1 \ \textit{ sinon } \ v(C) = 0 \ .$$

Démonstration. Par induction sur la définition des formules.

Atomes. C'est la définition de v.

Absurde. Pour toute valuation v, $v(\perp) = 0$. Or d'après le lemme 5.11, la théorie T^s n'est pas contradictoire donc $\perp \notin T^s$.

Implication. Supposons le résultat pour A et B, et montrons le pour $A \Rightarrow B$. Distinguons deux cas.

Supposons $A \Rightarrow B \in T^s$. Si $A \notin T^s$, on a $v(A) = 0$ par hypothèse d'induction et donc $v(A \Rightarrow B) = 1$. Si $A \in T^s$, on a $\vdash_{T^s} A \Rightarrow B$ et $\vdash_{T^s} A$. On en déduit (élimination) $\vdash_{T^s} B$. Par saturation $B \in T^s$, et donc $v(B) = 1$ par hypothèse d'induction, donc $v(A \Rightarrow B) = 1$.

Supposons $(A \Rightarrow B) \notin T^s$. Les deux cas $A \notin T^s$ et $B \in T^s$ sont exclus car ils contredisent l'hypothèse $(A \Rightarrow B) \notin T^s$, comme le montre ce qui suit.

Supposons $A \notin T^s$. Comme la théorie est complète on a que $\neg A \in T^s$. On en déduit (axiome de la théorie et élimination) $A \vdash_{T^s} \bot$. On a donc $A \vdash_{T^s} B$ (élimination de $\bot$). On en déduit $\vdash_{T^s} A \Rightarrow B$ (introduction), et donc par saturation (lemme 5.12) $(A \Rightarrow B) \in T^s$, contradiction.

Supposons $B \in T^s$. On a donc $\vdash_{T^s} B$ et donc $A \vdash_{T^s} B$ (affaiblissement). On en déduit $\vdash_{T^s} A \Rightarrow B$ (introduction), et donc par saturation $(A \Rightarrow B) \in T^s$, contradiction.

On a donc $A \in T^s$ et $B \notin T^s$, donc par hypothèse d'induction $v(A) = 1$ et $v(B) = 0$ et donc $v(A \Rightarrow B) = 0$.

On laisse le lecteur compléter la preuve pour les autres connecteurs, $\neg$, $\wedge$, $\vee$. Il faudra à chaque fois utiliser les règles d'introduction et d'élimination correspondantes. ■

Corollaire 5.14 *Il existe une valuation v telle que pour toute formule C de T, $v(C) = 1$.*

Démonstration. Comme $T^s \supset T$, c'est une conséquence immédiate du lemme précédent (5.13). ■

On a donc bien démontré le théorème de complétude pour le calcul propositionnel. ■

On peut remarquer que toutes les règles propositionnelles de la figure 5 page 18 ont été utilisées au cours de la preuve (entre la proposition 5.2 page 38, et les lemmes 5.11, 5.12 et 5.13), en dehors bien-sûr des règles d'introduction et d'élimination des connecteurs distincts de $\Rightarrow$ et $\bot$, puisque nous n'avons pas traité ces connecteurs dans la preuve du lemme 5.13. Nous avons utilisé la négation, mais nous pouvons considérer que $\neg F$ est une abréviation de $A \Rightarrow \bot$ (les règles utilisées sont alors celles de l'implication). Si vous complétez la preuve, pour chaque connecteur ses règles d'introduction et d'élimination doivent être utilisées.

On pourrait également se contenter de démonter le théorème de complétude pour le système de connecteurs $\{\Rightarrow, \bot\}$, qui est fonctionnellement complet. On pourrait utiliser cette propriété pour étendre le résultat aux autres connecteurs :

- Il faut vérifier syntaxiquement et sémantiquement les équivalences exprimant les connecteurs $\{\neg, \wedge, \vee\}$ à partir de $\{\Rightarrow, \bot\}$.
- Il faut utiliser la propriété de substitution 4.2 page 28 (dans le cas où l'on ne substitue des formules propositionnelles), et montrer et utiliser une propriété de substitution pour l'équivalence par déduction syntaxique.

6 Syntaxe des langages du premier ordre.

6.1 Introduction.

On va maintenant définir la *syntaxe* d'un langage du premier ordre, c'est à dire les constructions correctes pour les *termes* qui désignent des objets mathématiques, et les *formules* qui désignent des propriétés ou des assertions sur ces objets. On utilise à nouveau les définitions inductives. A cause des quantificateurs, et donc des liaisons de variables, les choses vont être un peu plus compliquée qu'en calcul propositionnel.

6.2 Signature.

On rappelle qu'une *signature* est la donnée d'une suite de symboles de constantes, d'une suite de symboles de fonctions chacun muni d'un entier appelé "arité" du symbole, et d'une suite de symboles de prédicats, chacun également muni d'une arité. Chacune de ces suites peut-être vide.

Sauf précision l'égalité fait toujours partie du langage (on précise parfois langage égalitaire du premier ordre), et le signe de l'égalité n'apparaît donc pas dans la signature.

On va définir dans la suite le langage égalitaire du premier ordre de signature $\mathcal{S}$, on dira plus rapidement le langage de signature $\mathcal{S}$, voire le langage $\mathcal{S}$.

6.3 Les termes.

On définit tout d'abord les *termes* d'un langage donné, qui désignent des objets. Voyons tout d'abord un exemple.

6.3.1 Termes de l'arithmétique.

Prenons pour exemple le langage de l'arithmétique de Peano, de signature $\mathcal{P} = (0, s, +, \cdot, \leqslant)$ avec les arités usuelles. On suppose donnée un ensemble dénombrable de variables $\mathcal{V}$. L'ensemble $\mathcal{T}_\mathcal{P}$ des termes de signature $\mathcal{P}$ est défini inductivement :

variable toute variable x de $\mathcal{V}$ est un terme.

constante 0 est un terme.

successeur si t est un terme $s\,t$ est un terme.

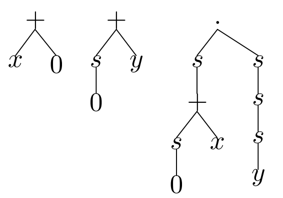
addition si t et t' sont des termes $(t + t')$ est un terme

multiplication si t et t' sont des termes $(t \cdot t')$ est un terme.

L'ensemble $\mathcal{T}$ est donc défini comme le plus petit ensemble de mots qui vérifie cette propriété. On admet la propriété dite de lecture unique, à savoir qu'un terme de $\mathcal{T}$ a un seul arbre de dérivation, ce qui autorise les définitions par induction sur les termes. En gros, il suffit d'avoir mis suffisamment de parenthèses et éventuellement de "séparateurs" (",", espacements etc.) pour que cette propriété soit vérifiée, et c'est bien le cas de la définition ci-dessus. On rappelle que la propriété de lecture unique énonce que la structure inductive est librement engendrée : le terme écrit comme mot (linéairement) est bien une représentation fidèle de son arbre de dérivation. L'arbre de dérivation serait la "vraie" notion de terme, mais évidemment les arbres sont peu commodes à écrire ! Voici des exemples de termes :

$$(x + 0), \quad (s\,0 + y), \quad (s(\,s\,0 + x) \cdot s\,s\,s\,y)$$

Leurs arbres de dérivations sont respectivement :



Bien entendu, le symbole de prédicat $\leqslant$ n'apparaît pas dans les termes.

6.3.2 Cas général.

On va définir les termes d'un langage de signature $\mathcal{L}$ en considérant que tous les symboles de fonction sont en notation préfixe (ce que nous n'avons pas fait dans le cas de $+$ et $\cdot$ ci-dessus).

L'ensemble $\mathcal{T}_{\mathcal{L}}$ des termes de signature $\mathcal{L}$ est défini inductivement par :

variable toute variable x de $\mathcal{V}$ est un terme.

constante toute constante est un terme.

fonction pour chaque symbole de fonction n-aire f de $\mathcal{L}$, si $t_1, \ldots, t_n$ sont des termes alors $f\,t_1\,\ldots\,t_n$ est un terme.

Le choix des notations (parenthèses, espaces ou "," pour séparer les arguments) n'a pas grande importance. Le principal est que les notations ci-dessus assurent la propriété de lecture unique : un terme de $\mathcal{T}_{\mathcal{L}}$ a un seul arbre de dérivation. On pourra donc utiliser des définitions par induction sur les termes[14].

Nous utiliserons des langages sans symboles de fonction ni de constante, comme le langage des ordre ($<$) ou celui de la théorie des ensembles ($\in$), qui ont donc pour seuls termes les variables.

6.3.3 Quelques définitions.

On peut maintenant définir par induction sur cette définition quelques notions, par un exemple, la notion de *terme clos*, un terme sans variables, se définit par :

variable Une variable x de $\mathcal{V}$ n'est pas un terme clos.

constante Toute constante est un terme clos.

fonction Pour chaque symbole de fonction n-aire f de $\mathcal{L}$, $f\,t_1\,\ldots\,t_n$ est un terme clos si $t_1, \ldots, t_n$ sont des termes clos.

Remarquons qu'un langage sans constantes n'a pas de termes clos.

On peut définir la substitution sur les termes. Étant donné un terme u et une variable x, on définit $t[u/x]$ (t dans lequel u remplace x) pour tout terme t de $\mathcal{L}$ par induction sur la définition des termes :

variable Soit y une variable dans $\mathcal{V}$, si $y \neq x$ $y[u/x] = y$, sinon $x[u/x] = u$.

constante Soit c une constantes de $\mathcal{L}$, $c[u/x] = c$.

fonction pour chaque symbole de fonction n-aire f de $\mathcal{L}$, pour $t_1, \ldots, t_n$ des termes, $f\,t_1\,\ldots\,t_n[u/x] = f\,t_1[u/x]\,\ldots\,t_n[u/x]$.

6.4 Formules atomiques.

6.4.1 L'arithmétique.

On reprend le langage de l'arithmétique, de signature $\mathcal{P}$. Les *formules atomiques* sont formées à partir de l'égalité et des symboles de prédicat du langage, ici $\leqslant$.

égalité Si t_1 et t_2 sont des termes de $\mathcal{P}$, $t_1 = t_2$ est une formule atomique de $\mathcal{P}$.

ordre Si t_1 et t_2 sont des termes de $\mathcal{P}$, $t_1 \leqslant t_2$ est une formule atomique de $\mathcal{P}$.

Par exemple $(s\,0 + x) = y$, $(s\,0 + x) \cdot (s\,0 + x) = 0$, $0 \leqslant 0$ sont des formules atomiques de l'arithmétique. Remarquez qu'intuitivement les formules atomiques sont essentiellement les égalités polynomiales et les inégalités polynomiales sur les entiers.

Une formule atomique close est définie comme une formule qui n'est construite qu'avec des termes clos, ainsi parmi les formules ci-dessus la seule formule close est $0 \leqslant 0$. La formule $x = x$ n'est pas une formule close, même si intuitivement elle ne dépend pas de x.

[14] Dans le cas général on s'est restreint à la notation préfixe. Les termes de l'arithmétique de l'exemple précédent utilisent la notation infixe. Les preuves de lecture unique utilisent donc des arguments différents.

6.4.2 Cas général.

On considère que le langage est égalitaire. Exceptionnellement il nous arrivera de considérer des langages non égalitaires, auquel cas on supprime bien sûr la première des clauses de la définition.

égalité Si t_1 et t_2 sont des termes de $\mathcal{L}$, $t_1 = t_2$ est une formule atomique de $\mathcal{L}$.

prédicat Pour chaque symbole de prédicat n-aire P de $\mathcal{L}$, si $t_1, \ldots, t_n$ sont des termes de $\mathcal{L}$, alors $P\,t_1 \ldots t_n$ est une formule atomique de $\mathcal{L}$.

On définit les formules atomiques closes (aucune variable n'apparaît dans la formule). La substitution sur les formules atomiques est la substitution sur chacun des termes : $P\,t_1 \ldots t_n[t/x] = P\,t_1[t/x] \ldots t_n[t/x]$.

Pour la plupart des langages étudiés, les symboles de prédicats du langage seront des symboles de prédicat binaire (l'ordre, l'appartenance) que l'on notera comme d'habitude de façon infixe (comme $\leqslant$ pour l'arithmétique ci-dessus).

6.5 Formules.

6.5.1 Définition des formules.

On construit de nouvelles formules à l'aide de *connecteurs* (les mêmes qu'en calcul propositionnel) et de *quantificateurs*. Les quantificateurs sont $\forall$ et $\exists$.

L'ensemble des *formules* du langage $\mathcal{L}$ est défini inductivement par les clauses suivantes

Formules atomiques. Les formules atomiques de $\mathcal{L}$ sont des formules.

absurde $\perp$ est une formule.

négation Si F est une formule $\neg F$ est une formule.

implication Si F et G sont des formules $(F \to G)$ est une formule.

conjonction Si F et G sont des formules $(F \wedge G)$ est une formule.

disjonction Si F et G sont des formules $(F \vee G)$ est une formule.

quantification universelle Si F est une formule et x une variable, alors $\forall x\ F$ est une formule.

quantification existentielle Si F est une formule et x une variable, alors $\exists x\ F$ est une formule.

Quand une formule n'utilise pas de quantificateurs on dit que c'est une *formule propositionnelle* (c'est une formule du calcul propositionnel construit sur toutes les formules atomiques du langage).

Là encore on admettra le théorème de lecture unique, et donc on utilisera les définitions par induction.

Voici deux formules du langage de l'arithmétique $\mathcal{P}$:

$$\forall x\ (x \leqslant s\,0 \to (x = 0 \vee x = s\,0), \quad \exists y(x = 2 \cdot y \vee x = 2 \cdot y + s\,0)$$

Rappelons que nous avons abordé informellement la notion de variable libre et liée. Ainsi la première de cette formule est close, la variable y n'a que des occurrences liées dans cette formule. Dans la seconde de ces formules, les occurrences de x sont libres et celles de y sont liées.

6.5.2 Occurrences libres ou liées d'une variable, substitution...

La notion de formule close se définit sans difficulté pour les formules propositionnelles. En présence de quantificateurs, c'est un peu plus compliqué, on peut maintenant le faire proprement grâce aux définitions par induction.

On ne va pas définir formellement la notion intuitive d'*occurrence* d'un symbole dans une formule : la place ou l'endroit où ce symbole apparaît, ce serait inutilement technique. On pourrait alors définir par induction les notions déjà abordées d'occurrences de variables libres et liées. Ceci ne nous empêchera d'ailleurs pas d'utiliser ces notions. En fait on peut le plus souvent se contenter de définitions où la notion d'occurrence reste implicite, comme dans ce qui suit. Tout d'abord, on peut définir par induction *x apparaît dans laformule F* (définition laissée au lecteur). On dira également *x a la occurrence dans une formule F*.

On définit ensuite *x apparaît libre dans une formule F* (ou x a une *occurrence libre* dans F) également par induction sur les formules :

Formules atomiques. Si x apparaît dans une formule atomique A, x apparaît libre dans A.

absurde x n'apparaît pas libre dans $\perp$.

négation x apparaît libre dans $\neg F$ ssi x apparaît libre dans F.

implication... x apparaît libre dans $(F \to G)$ ssi x apparaît libre dans F ou x apparaît libre dans G. De même pour la conjonction et la disjonction.

quantifications Si $y \neq x$, x apparaît libre dans $\forall y\ F$ ssi x apparaît libre dans F, sinon x n'apparaît pas libre dans $\forall x\ F$. De même pour la quantification existentielle.

On peut maintenant définir une *formule close* : c'est une formule F telle qu'aucune variable de $\mathcal{V}$ n'apparaît libre dans F. On peut également définir *x apparaît liée dans F*, ou x a une *occurence liée* dans F : cela signifie que x apparaît dans F et que x n'apparaît pas libre dans F.

On considérera que *deux formules sont égales* si elles sont identiques modulo un renommage cohérent des variables liées. Il s'agit d'une notion d'égalité purement syntaxique, que l'on va définir formellement par induction.

Voyons d'abord la substitution. Nous avons besoin de la substitution pour par exemple les règles de preuve de la déduction naturelle (quantificateurs, égalité).

La notion de substitution que nous allons définir et utiliser est une *substitution logique*, qui évite la *capture de variable* (voir paragraphe 2.2 page 16). Elle diffère de la *susbtitution simple* qui est la substitution sur les mots (on remplace toutes les occurrences d'un lettre par un mot). Quand on parlera de substitution sans préciser, c'est de la susbtitution logique qu'il s'agira. La substitution logique et la substitution simple ne diffèrent pas sur les termes. La substitution simple d'un terme à une variable se définit sur les formules par induction comme pour les termes.

On donne maintenant une définition par induction de la substitution logique. On la notera $F[t/x]$ que l'on lit "F dans laquelle t remplace x". Intuitivement la formule substituée est définie à renommage cohérent des variables liées près. Plutôt que de travailler dans le quotient des formules par renommage des variables liées, on va définir un représentant de ce quotient par induction, représentant qui n'a malheureusement rien de canonique. On va considérer que l'ensemble $\mathcal{V}$ des variables du langage est énuméré soit $\mathcal{V} = \{x_i \ / \ i \in \mathbb{N}\}$, l'énumération fournit un bon ordre sur les variables.

Étant donné un terme t, une variable x, on définit pour toute formule F la formule $F[t/x]$ par induction :

Formules atomiques. Si A est une formule atomique $A[t/x]$ est déjà défini.

absurde $\perp[t/x] := \perp$.

négation $\neg F[t/x] := \neg F[t/x]$.

implication... $(F \to G)[t/x] := (F[t/x] \to G[t/x])$ De même pour la conjonction et la disjonction.

quantifications Si $y = x$, $\{\forall x F\}[t/x] := \forall x F$.

Si $y \neq x$, et y n'apparaît pas dans t $\{\forall y F\}[t/x] := \forall y \{F[t/x]\}$.

si $y \neq x$ et y apparaît dans t, $\{\forall y F\}[t/x] := \forall z F[z/y][t/x]$, où z est une variable qui n'apparaît pas dans t ni dans F, et, pour être déterministe, on choisit pour z la "plus petite" variable (la première dans l'énumération) vérifiant ceci. Exceptionnellement la notation $F[z/y]$ indique la substitution ordinaire, ce qui est possible car z n'apparaît pas dans F (libre ou liée).

De même pour la quantification existentielle.

Remarquons qu'à la dernière clause nous n'avons pas tout à fait respecté le schéma de définition par induction tel que nous l'avons énoncé : on a $F[z/y]$ à la place de F. C'est possible, car ces deux formules ont la même complexité. On peut considérer que l'on définit en une étape la substitution pour toutes les formules de même complexité.

On pourrait étendre facilement cette définition à la *substitution simultanée* des termes $u_1, \ldots, u_n$ aux variables $x_1, \ldots, x_n$, notée $F[u_1/x_1, \ldots, u_n/x_n]$. Remarquez que ce n'est en général pas la même chose que la composée des substitutions $F[u_1/x_1] \ldots [u_n/x_n]$ (par exemple si x_2 apparaît dans u_1).

On peut définir également l'*égalité des formules à renommage cohérent des variables liées près*, que l'on va noter temporairement $\approx$, afin de réserver le signe $=$ pour l'identité des formules en tant que mots. Il ne faut pas confondre ce signe "=" qui est un signe du meta-langage, avec le signe "=" qui peut apparaître dans F et qui lui est un signe du langage étudié. Pratiquement cela ne pose pas réellement de problèmes, le contexte permettant de trancher. On définit par induction sur F, $F \approx E$ pour une formule E quelconque.

Formules atomiques, absurde. Si A est une formule atomique ou l'absurde $A \approx E$ ssi $A = E$.

négation $\neg F \approx E$ ssi E s'écrit $\neg E'$ et $F \approx E'$.

implication... $(F \to G) \approx E$ ssi E s'écrit $(E' \to E'')$ et $F \approx E'$ et $G \approx E''$.

De même pour la conjonction et la disjonction.

quantifications $\forall x\ F \approx E$ ssi E s'écrit $\forall y\ E'$ et $F \approx E'[x/y]$.

De même pour la quantification existentielle.

La présence du paramètre E peut rendre la définition un peu moins évidente que les précédentes. Formellement on peut considérer que l'on a définit l'ensemble des formules E telles que $F \approx E$ par induction sur F.

Dorénavant $F \approx G$ se notera $F = G$, et l'on considérera comme égales des formules égales à renommage des variables libres près.

Le fait de noter $F = G$ pour $F \approx G$ n'est pas innocent, on sous-entend que l'on peut travailler dans un quotient et que donc $\approx$ a les propriétés de l'égalité, symétrie, transitivité, stabilité par constructions (connecteurs, quantificateurs), par substitution etc. Ces propriétés se démontrent par induction sans grande difficulté autre que d'écriture. On les admettra.

Nous allons nous arrêter là pour la syntaxe. Même si nous n'avons pas terminé la formalisation, j'espère que le lecteur a été convaincu que l'on peut définir formellement les notions de variable libre, de formule close, de substitution etc. Elles sont suffisamment intuitives pour être maniées sans faire appel à ces définitions, et le plus souvent nous admettrons les propriétés "évidentes" dont nous avons besoin.

On retrouvera ces définitions formelles quand il s'agira de démontrer les théorèmes d'incomplétude de Gödel. En effet dans ce cas il s'agit justement de formaliser le langage d'une théorie dans cette même théorie mathématique (arithmétique ou ensembliste). Il s'agira donc de coder dans la théorie la définition inductive des formules, quelques définitions par induction sur les formules, essentiellement la substitution vue ci-dessus, et la définition inductive de la prouvabilité, que l'on peut maintenant énoncer formellement.

En effet la relation $\Gamma \vdash_T C$ entre un multi-ensemble de formules propositionnelles Γ et une formule C est la relation dont la définition inductive est donnée par les règles de la figure 5 page 18 et de la figure 6 page 19 (ces règles utilisent les notions d'occurrence libre et de substitution). On reprend une remarque déjà faite en calcul propositionnel (voir 4.5 page 30) : une relation de déduction peut manifestement avoir plusieurs dérivations, et donc il n'est donc pas question de définir une fonction par induction sur la définition d'une relation de déduction.

7 Sémantique des langages du premier ordre.

7.1 Introduction.

On va maintenant définir formellement *la sémantique* d'un langage du premier ordre. Il s'agit d'interpréter dans une structure donnée les termes et formules du langage que nous avons définis.

On va tout simplement suivre la définition intuitive d'interprétation que vous avez déjà pratiqué. La sémantique du calcul des prédicats généralise la sémantique du calcul propositionnel, qui peut apparaître comme un cas particulier de celle-ci. Bien que l'on ne parle plus de valuation, il est toujours essentiel qu'il n'y ait que deux valeurs de vérité. C'est à dire que dans une structure, un énoncé clos sera vrai ou faux, peu importe que nous ne sachions pas quelle alternative est la bonne.

7.2 Signature et structure.

Étant donné une signature $\mathcal{S}$, on rappelle qu'une *$\mathcal{S}$-structure* $\mathcal{M}$ est la donnée :
- d'un ensemble *non vide*, soit M, appelé *ensemble de base* de la structure $\mathcal{M}$;
- d'un élément $\overline{c}^{\mathcal{M}}$ de M pour chaque symbole de constante c de $\mathcal{S}$;
- d'une fonction[15] $\overline{f}^{\mathcal{M}}$ de M^n dans M pour chaque symbole de fonction f d'arité n dans $\mathcal{S}$;
- d'un sous-ensemble $\overline{R}^{\mathcal{M}}$ de M^n pour chaque symbole de prédicat R d'arité n de $\mathcal{S}$.

Ainsi, si nous prenons le langage de l'arithmétique de signature $\mathcal{P} = (0, s, +, \cdot, \leqslant)$ défini au paragraphe 6.3.1, on peut définir $\mathcal{N} = (\mathbb{N}, \overline{0}^{\mathcal{N}}, \overline{s}^{\mathcal{N}}, \overline{+}^{\mathcal{N}}, \overline{\cdot}^{\mathcal{N}}, \overline{\leqslant}^{\mathcal{N}})$ muni des constantes, opérations et relations, usuelles pour interpréter chacun des symboles de $\mathcal{P}$. Ainsi s est interprété par la fonction de $\mathbb{N} \to \mathbb{N}$ qui ajoute 1 à un entier, $+$ par la fonction addition usuelle (à deux arguments) etc.

Des notations comme $\overline{s}^{\mathbb{N}}$, $\overline{+}^{\mathbb{N}}$ sont assez lourdes. S'il n'y a pas d'ambiguïté sur la structure concernée, on oubliera l'indication de celle-ci, par exemple on notera $\overline{s}$, $\overline{+}$. Il nous arrivera même de noter de la même façon symbole et interprétation, s'il est clair d'après le contexte que l'on parle de l'interprétation.

Une autre $\mathcal{P}$-structure est $\mathbb{Z}$ muni des opérations et prédicats usuels, $\mathbb{Z}/2\mathbb{Z}$ en interprétant 0 par le 0 de $\mathbb{Z}/2\mathbb{Z}$, les opérations avec leur interprétation usuelle, et l'ordre par exemple par $\{(0,0),(0,1),(1,1)\}$ (ce qui signifie que dans cette structure l'on a $0 \leqslant 0$, $0 \leqslant 1$ et $1 \leqslant 1$, mais pas $1 \leqslant 0$)[16]. On pourrait tout aussi bien construire une structure $\mathcal{N}'$ d'ensemble de base N, où 0 est interprété par 1, s par la fonction constante nulle etc.

Il est important que l'ensemble de base d'une structure soit non vide. Outre que le cas où l'ensemble de base est vide a assez peu d'intérêt, cela compliquerait les systèmes de preuves si l'on voulait que celles-ci restent correctes dans de telles structures.

Par contre nous n'avons pas éliminé les structures dont l'ensemble de base a un seul élément, un cas évidemment dégénéré puisque l'interprétation des symboles de fonctions et de constantes est imposée, et que les symboles de prédicats n'ont que deux interprétations possibles ($\emptyset$ ou M^n).

7.3 Environnements.

Nous allons définir successivement l'interprétation des termes et des formules. On pourrait choisir de n'interpréter que les termes et les formules clos, mais la notion de formule close ne peut se définir directement par induction à cause des quantificateurs : si la formule $\forall x\ F$ est close, on s'attend en général à ce que la formule F dépende de x donc ne soit pas close. On va donc interpréter un terme ou une formule avec des variables libres, ce qui est possible si l'on a choisit d'attribuer une certaine valeur aux variables libres en question : c'est ce que l'on appelle un *environnement*.

Plus formellement, on définira un environnement dans une structure $\mathcal{M}$ comme la donnée d'une application d'un ensemble fini de variables dans l'ensemble de base M de $\mathcal{M}$. Si $x_1, \ldots, x_p$ sont

[15]Nous considérons que les fonctions sont *partout définies.*

[16]Cet ordre n'est pas compatible avec l'addition

les seules variables affectées par l'environnement, et ont pour images $m_1, \ldots, m_p$ (des éléments de M) on note celui ci :

$$[x_1 := m_1, \ldots, x_p := m_p]$$

et parfois on note en abrégé : $[\overrightarrow{x} := \overrightarrow{m}]$.

Remarquez que pour interpréter les formules propositionnelles (sans quantificateurs) et closes (donc sans variables) on peut se passer de la notion d'environnement pour définir l'interprétation.

7.4 Interprétation.

Soit $\mathcal{M}$ une $\mathcal{S}$-structure d'ensemble de base M. On va définir successivement l'interprétation des termes (par induction), des formules atomiques, et des formules (par induction).

7.4.1 Interprétation des termes.

L'interprétation d'un terme t du langage de signature S dans $\mathcal{M}$ pour l'environnement $[x_1 := m_1, \ldots, x_p := m_p]$ est notée $\overline{t}^{\mathcal{M}}[x_1 := m_1, \ldots, x_p := m_p]$. C'est un élément de M qui est défini par induction sur t, pour tout environnement qui attribue une valeur à toutes les variables libres de t.

variable $\overline{x_i}^{\mathcal{M}}[x_1 := m_1, \ldots, x_p := m_p] = m_i$, si $x = x_i$ est une variable affectée par l'environnement (sinon l'interprétation n'est pas définie) ;

constante $\overline{c}^{\mathcal{M}}[x_1 := m_1, \ldots, x_p := m_p] = \overline{c}^{\mathcal{M}}$ pour c une constante ;

fonction $\overline{f\, t_1 \ldots t_n}^{\mathcal{M}}[x_1 := m_1, \ldots, x_p := m_p] =$
$\overline{f}^{\mathcal{M}}(\overline{t_1}^{\mathcal{M}}[x_1 := m_1, \ldots, x_p := m_p], \ldots, \overline{t_n}^{\mathcal{M}}[x_1 := m_1, \ldots, x_p := m_p])$
pour f un symbole de fonction de $\mathcal{S}$ d'arité n.

7.4.2 Interprétation des formules : notations

Une formule est interprétée par "vrai" ou "faux". Pour dire qu'une formule F est *vraie dans la structure* $\mathcal{M}$ relativement à l'environnement $[x_1 := m_1, \ldots, x_p := m_p]$, on note :

$$\mathcal{M} \models F[x_1 := m_1, \ldots, x_p := m_p]$$

et on dira également que la formule F est *satisfaite* dans la structure $\mathcal{M}$, pour l'environnement $[x_1 := m_1, \ldots, x_p := m_p]$. Dans le cas contraire on notera

$$\mathcal{M} \not\models F[x_1 := m_1, \ldots, x_p := m_p] \ .$$

7.4.3 Interprétation des formules atomiques.

prédicat $\mathcal{M} \models R\, t_1 \ldots, t_n[x_1 := m_1, \ldots, x_p := m_p]$ ssi
$(\overline{t_1}^{\mathcal{M}}[x_1 := m_1, \ldots, x_p := m_p], \ldots, \overline{t_n}^{\mathcal{M}}[x_1 := m_1, \ldots, x_p := m_p]) \in \overline{R}^{\mathcal{M}}$, pour R un prédicat de S d'arité n, toutes les variables libres de $R\, t_1 \ldots t_n$ étant parmi $x_1, \ldots, x_p$.

égalité $\mathcal{M} \models t = t'[x_1 := m_1, \ldots, x_p := m_p]$ ssi
$\overline{t}^{\mathcal{M}}[x_1 := m_1, \ldots, x_p := m_p] = \overline{t'}^{\mathcal{M}}[x_1 := m_1, \ldots, x_p := m_p])$, toutes les variables libres de t et t' étant parmi $x_1, \ldots, x_p$.

Remarquez que dans la dernière assertion, le signe "=" utilisé dans deux sens différents : comme symbole du langage (première occurrence), et dans son sens usuel, celui de l'identité du meta-langage pour la deuxième occurrence.

La différence entre l'égalité et les autres symboles de prédicat est que pour l'égalité, l'interprétation est imposée. La notation choisie n'a pas grande importance, on aurait tout aussi bien pu écrire la clause pour l'égalité ainsi (notons $[\overrightarrow{x} := \overrightarrow{m}]$ pour $[x_1 := m_1, \ldots, x_p := m_p]$) :

$$\mathcal{M} \models t = t'[\overrightarrow{x} := \overrightarrow{m}] \ \text{ ssi } (\overline{t}^{\mathcal{M}}[\overrightarrow{x} := \overrightarrow{m}], \overline{t'}^{\mathcal{M}}[\overrightarrow{x} := \overrightarrow{m}]) \in \{(x, x) \ / \ x \in M\} \ .$$

7.4.4 Interprétation des formules.

On peut maintenant définir l'interprétation des formules par induction.

Notez bien que pour les connecteurs binaires, définis sur F et G, il n'y a que 4 cas possibles suivant que $\mathcal{M} \models F[\overrightarrow{x} := \overrightarrow{m}]$ ou $\mathcal{M} \not\models F[\overrightarrow{x} := \overrightarrow{m}]$ et $\mathcal{M} \models G[\overrightarrow{x} := \overrightarrow{m}]$ ou $\mathcal{M} \models G[\overrightarrow{x} := \overrightarrow{m}]$.

Formules atomiques Voir le paragraphe précédent.

absurde $\mathcal{M} \not\models \bot[x_1 := m_1, \dots, x_p := m_p]$.

négation $\mathcal{M} \models \neg F[x_1 := m_1, \dots, x_p := m_p]$ ssi $\mathcal{M} \not\models F[x_1 := m_1, \dots, x_p := m_p]$.

implication $\mathcal{M} \models (F \to G)[x_1 := m_1, \dots, x_p := m_p]$ ssi

$\mathcal{M} \models F[x_1 := m_1, \dots, x_p := m_p] \Rightarrow \mathcal{M} \models G[x_1 := m_1, \dots, x_p := m_p]$.

Le signe $\Rightarrow$ désigne l'implication dans le meta-langage. Cette clause peut paraître plus claire énoncée sous forme contraposée :

$\mathcal{M} \not\models (F \to G)[x_1 := m_1, \dots, x_p := m_p]$ ssi

$\mathcal{M} \models F[x_1 := m_1, \dots, x_p := m_p]$ et $\mathcal{M} \not\models G[x_1 := m_1, \dots, x_p := m_p]$.

conjonction $\mathcal{M} \models (F \wedge G)[x_1 := m_1, \dots, x_p := m_p]$ ssi

$\mathcal{M} \models F[x_1 := m_1, \dots, x_p := m_p]$ et $\mathcal{M} \models G[x_1 := m_1, \dots, x_p := m_p]x$.

disjonction $\mathcal{M} \models (F \vee G)[x_1 := m_1, \dots, x_p := m_p]$ ssi

$\mathcal{M} \models F[x_1 := m_1, \dots, x_p := m_p]$ ou $\mathcal{M} \models G[x_1 := m_1, \dots, x_p := m_p]$.

Le "ou" est le ou inclusif du meta-langage, usuel en mathématique. Cette clause peut s'exprimer peut-être plus clairement par contraposée :

$\mathcal{M} \not\models (F \vee G)[x_1 := m_1, \dots, x_p := m_p]$ ssi $\mathcal{M} \not\models F[x_1 := m_1, \dots, x_p := m_p]$ et $\mathcal{M} \not\models G[x_1 := m_1, \dots, x_p := m_p]$.

quantification universelle $\mathcal{M} \models \forall x\ F[x_1 := m_1, \dots, x_p := m_p]$ ssi

$\mathcal{M} \models F[x_1 := m_1, \dots, x_p := m_p, x := m]$ pour tout élément m de M, ceci si $x \notin \{x_1, \dots, x_p\}$.

Si $x \in \{x_1, \dots, x_n\}$ (x est déjà affectée par l'environnement) on choisit y la "première" (dans l'ordre d'énumération des variables) non affectée dans l'environnement et on donne la définition ci-dessus pour $\forall y\ F[y/x]$.

quantification existentielle $\mathcal{M} \models \exists x\ F[x_1 := m_1, \dots, x_p := m_p]$ ssi

$\mathcal{M} \models F[x_1 := m_1, \dots, x_p := m_p, x := m]$ pour au moins un élément m de $\mathcal{M}$, ceci si $x \notin \{x_1, \dots, x_p\}$.

Si $x \in \{x_1, \dots, x_n\}$, on procède comme pour $\forall$.

Dans le cas d'une formule close F on dira qu'elle est vraie dans $\mathcal{M}$, ou satisfaite par $\mathcal{M}$, ou que $\mathcal{M}$ *est modèle* de F quand $\mathcal{M}$ est satisfaite par $\mathcal{M}$ pour l'environnement vide. On note alors

$$\mathcal{M} \models F\ .$$

Remarquez que dès qu'une formule close contient des quantificateurs, on a effectivement besoin de la notion d'environnement (non vide !) pour définir l'interprétation.

Une formule close de $\mathcal{L}$ est dite *universellement valide* quand elle est satisfaite dans toutes les $\mathcal{L}$-structures.

Un exemple de formule universellement valide pour n'importe quelle signature $\mathcal{L}$ est $\forall x\ x = x$. On vérifie en effet que pour toute structure $\mathcal{M}$, pour tout élément m de l'ensemble de base de $\mathcal{M}$, $\mathcal{M} \models x = x[x := m]$ (reprendre la définition de la satisfaction des formules atomiques), et pour toute structure $\mathcal{M}$, $\mathcal{M} \models \forall x\ x = x$ (environnement vide).

Voyons d'autres exemples, prenons une signature $\mathcal{L}$ quelconque, et soit F une formule de $\mathcal{L}$ ayant x comme seule variable libre. alors $\forall x\ F \to \exists x\, F$ est universellement valide. En effet soit $\mathcal{M}$ une $\mathcal{L}$-structure. Supposons que $\mathcal{M} \models \forall x\ F$. Cela signifie par définition que pour tout élément m de l'ensemble de base de $\mathcal{M}$ $\mathcal{M} \models F[x := m]$. Comme cet ensemble de base est non vide, il contient au moins un élément m_0 et comme $\mathcal{M} \models F[x := m_0]$, $\mathcal{M} \models \exists x F$. On a bien montré que $\mathcal{M} \models \forall x\ F \to \exists x\, F$.

Dans l'exemple ci-dessus, n'importe quelle formule F avec x pour seule variable libre convient. On parle alors de *schéma de formules*.

Les preuves telles que ci-dessus s'écrivent rarement. Il s'agissait ici de vérifier sur des exemples que la définition de satisfaction fonctionnait correctement. Ces preuves très formelles (dans le meta-langage) suivent de près les règles de la déduction de la logique du premier ordre, et donc autant faire les preuves directement dans le langage du premier ordre.

On arrêtera là pour le moment au sujet des formules universellement valides. On verra dans les chapitres suivant des schémas de formules universellement valides purement propositionnels, et d'autres utilisant les quantificateurs comme le précédent.

On passe sur quelques lemmes utiles comme la stabilité de l'interprétation par renommage cohérent des variables liées, la stabilité de l'interprétation par extension de l'environnement etc.

7.4.5 Remarques.

Cette formalisation de la sémantique des formules du premier ordre ci-dessus est due à Tarski. Il faut bien comprendre que plus que de définir l'interprétation d'une formule, il s'agit de formaliser celle-ci. En effet vous avez peu de chance de comprendre cette définition sans avoir déjà pratiqué les mathématiques, en particulier la négation, la conjonction et les quantificateurs ! Ainsi les connecteurs et les quantificateurs sont tout simplement définis ... en utilisant ces mêmes connecteurs dans le meta-langage. Ceci peut paraître tautologique, mais ne l'est pas car une définition formelle nous permet de faire proprement des démonstrations de résultats concernant la sémantique.

7.5 Satisfaisabilité, axiomatisation.

7.5.1 Formules satisfaisables.

Une formule close F du langage de signature $\mathcal{S}$ est dite *satisfaisable* si elle possède au moins un modèle, c'est à dire s'il existe une $\mathcal{S}$-structure $\mathcal{M}$ telle que $\mathcal{M} \models F$.

La notion de formule satisfaisable est en quelque sorte duale de la notion de formule universellement valide :

Proposition 7.1 *Une formule close F est satisfaisable ssi $\neg F$ n'est pas universellement valide.*

Démonstration. Si F est satisfaisable ssi existe un modèle $\mathcal{M}$ de F. Par définition de l'interprétation $\mathcal{M} \models F$ ssi $\mathcal{M} \not\models \neg F$ et donc F est satisfaisable ssi $\neg F$ n'est pas universellement valide. ■

Une formule satisfaisable définit donc une classe de structures : celles qui la satisfont. Par exemple la formule du langage de l'égalité (signature vide) :

$$\exists x \exists y\ \neg x = y$$

est satisfaite dans toutes les structures égalitaires – disons plus simplement les ensembles – ayant au moins deux éléments. On dit qu'elle *axiomatise* cette notion.

De même la formule

$$\exists x \exists y \forall z (z = x \vee z = y)$$

est satisfaite dans les ensembles au plus deux éléments, et la conjonction des deux formules précédentes :

$$\exists x \exists y\ \neg x = y \wedge \exists x \exists y \forall z (z = x \vee z = y)$$

dans les ensembles ayant deux éléments.

7.5.2 Quelques notations.

Tout d'abord on peut s'autoriser des abréviations usuelles. Par exemple on écrira $x \neq y$ pour $\neg x = y$, $\neq$ n'est pas un symbole du langage, $x \neq y$ désigne la formule $\neg x = y$, et il est clair qu'à une formule de signature $\mathcal{S} \cup \{\neq\}$ apparaît, on fait correspondre par une substitution "simple" une formule du langage $\mathcal{S}$.

En généralisant les exemples précédents, on s'aperçoit facilement que l'on peut axiomatiser les ensembles à au moins (resp. au plus, resp. exactement) trois éléments, puis généraliser à un entier n quelconque.

Pour axiomatiser la notion d'ensemble à au moins n éléments on écrira :

$$\exists x_1 \ldots \exists x_n \bigwedge_{0 \leqslant i < j \leqslant n} x_i \neq x_j. \tag{3}$$

Pour pouvoir exprimer ceci on a introduit des notations *qui ne font pas partie du langage étudié* mais qui là encore renvoient *pour chaque entier* n à une formule du premier ordre. En ce qui concerne la suite de quantificateurs $\exists x_1 \ldots \exists x_n \; F$, la signification pour chaque entier n est claire. La conjonction $\bigwedge_{0 \leqslant i < j \leqslant n} x_i \neq x_j$, même si elle pourrait se définir syntaxiquement, utilise implicitement les propriétés dite d'associativité et de commutativité de la conjonction qui sont sémantiques (l'interprétation d'une suite de conjonctions ne dépend ni de l'ordre ni du parenthésage). Tout ce qui nous intéresse est de savoir qu'il existe pour chaque entier n une formule du langage notée $\bigwedge_{0 \leqslant i < j \leqslant n} x_i \neq x_j$ vérifiant pour toute structure $\mathcal{M}$:

$$\mathcal{M} \models \bigwedge_{0 \leqslant i < j \leqslant n} x_i \neq x_j \;\text{ ssi pour tout couple }\; i, j \text{ tel que } 0 \leqslant i < j \leqslant n \; \mathcal{M} \models x_i \neq x_j \;.$$

On utilisera souvent ce genre de notation. Il faut bien prendre garde à ce que, pour prendre un exemple, on ne peut parler que de *conjonctions finies*, les conjonctions infinies, qui intuitivement ont un sens n'existent pas en langage du premier ordre. Dans le même ordre d'idée, l'entier n dans la formule (3) *ne fait pas partie du langage*. En particulier on ne peut quantifier sur n dans le langage. Par exemple on dirait volontiers qu'un ensemble est infini à la condition :

$$\text{Pour tout entier } n \;\; \exists x_1 \ldots \exists x_n \bigwedge_{0 \leqslant i < j \leqslant n} x_i \neq x_j.$$

Mais ceci n'est pas une formule du langage du premier ordre égalitaire (ce que nous avons souligné en n'utilisant pas le signe $\forall$ pour la quantification universelle sur n). Nous verrons qu'il n'est pas possible d'axiomatiser la notion d'ensemble infini par une seule formule du premier ordre.

7.5.3 Théories.

Un moyen de remédier partiellement au manque d'expressivité de la logique du premier ordre est d'utiliser des ensembles infinis de formule. Évidemment il faudra bien trouver un moyen "fini" de décrire cette infinité de formules.

Une *théorie du premier ordre* dans un langage de signature $\mathcal{S}$ est un ensemble de formules *closes* du langage (comme on se situe en logique du premier ordre on parlera simplement de théorie).

Une $\mathcal{S}$-structure est $\mathcal{M}$ *modèle d'une théorie* T quand elle est modèle de chacune des formules de la théorie. On note comme pour les formules $\mathcal{M} \models T$.

Une théorie est dite *satisfaisable* si elle a un modèle, c'est à dire s'il existe une structure $\mathcal{M}$ qui est modèle de la théorie. Une théorie est dite *incohérente* ou inconsistente dans le cas contraire, c'est à dire si elle n'a pas de modèle. On dira aussi plus tard théorie contradictoire, mais pour le moment nous réservons ce mot pour la notion syntaxique de déduction.

Une propriété des $\mathcal{S}$-structures est dite *axiomatisée* par une théorie T quand les structures ont la propriété en question ssi elles sont modèles de T.

Ainsi la notion d'ensemble infini est axiomatisée par la théorie (infinie) :

$$\{\exists x_1 \dots \exists x_n \bigwedge_{0 \leqslant i < j \leqslant n} x_i \neq x_j \; / \; n \in \mathbb{N}\}. \tag{4}$$

Quand une propriété est axiomatisée par une théorie finie on dit qu'elle est *finiment axiomatisable.* Il est facile de voir qu'alors la propriété est axiomatisée par une seule formule : la conjonction des axiomes de la théorie.

Tout ne peut pas se résoudre en considérant une infinité de formules. Ainsi on peut exprimer dans le langage de l'égalité pour chaque entier n qu'un ensemble possède au plus n éléments, comme devrait vous en convaincre l'écriture suivante :

$$\exists x_1 \dots \exists x_n \; \forall z (z = x_1 \vee \dots \vee z = x_n)$$

que l'on peut aussi écrire :

$$\exists x_1 \dots \exists x_n \; \forall z \bigvee_{i=1}^{n} z = x_i \; .$$

Un ensemble est fini s'il vérifie :

$$\text{Il existe un entier } n \text{ tel que } \exists x_1 \dots \exists x_n \; \forall z \bigvee_{i=1}^{n} z = x_i \; .$$

mais il n'y a aucun moyen apparent d'exprimer ceci dans le langage égalitaire du premier ordre, même avec une infinité de formules. On démontrera qu'en fait la notion d'ensemble fini ne s'axiomatise pas au premier ordre.

Évidement on pourra exprimer qu'un ensemble est fini ou infini par une formule de la théorie des ensembles, qui est elle même une théorie du premier ordre, ce qui n'a rien de contradictoire avec ce qui précède, puisque ces notions seront relatives à un modèle "attendu" de la théorie des ensembles.

7.6 Morphismes.

On généralise les notions de morphisme déjà abordées en algèbre à une signature quelconque.

Étant donné une signature $\mathcal{S}$ on appelle *$\mathcal{L}$-homomorphisme* de $\mathcal{S}$-structures, une application ϕ d'une $\mathcal{S}$-structure $\mathcal{M}$ dans une $\mathcal{S}$-structure $\mathcal{N}$ qui vérifie :

constante $\phi\left(\overline{c}^{\mathcal{M}}\right) = \overline{c}^{\mathcal{N}}$ pour tout symbole de constante c de $\mathcal{L}$.

fonction $\phi\left(\overline{f}^{\mathcal{M}}(m_1, \dots, m_n)\right) = \overline{f}^{\mathcal{N}}(\phi(m_1), \dots, \phi(m_n))$ pour tout symbole de fonction f d'arité n, pour tous $m_1, \dots, m_n \in \mathcal{M}$.

prédicat Si $(m_1, \dots, m_n) \in \overline{R}^{\mathcal{M}}$ alors $(\phi(m_1), \dots, \phi(m_n)) \in \overline{R}^{\mathcal{N}}$, pour tout symbole de prédicat R d'arité n, pour tous $m_1, \dots, m_n \in \mathcal{M}$.

Cette définition ne fait intervenir que la signature, et aucune notion d'axiomatique. Par exemple l'identité est un morphisme $(\mathbb{N}, 0, +)$ dans $(\mathbb{Z}, 0, +)$ (signature $(0, +)$), qui est un morphisme de monoïde mais pas de groupe. Si l'on veut parler de morphisme de groupe, on ajoutera au langage un symbole de fonction unaire "$-$" pour l'opposé.

Une conséquence immédiate de cette définition est la suivante :

Lemme 7.2 *Soit $\mathcal{S}$ une signature, deux $\mathcal{S}$-structures $\mathcal{M}$ et $\mathcal{N}$, ϕ un homomorphisme de $\mathcal{M}$ dans $\mathcal{N}$. Soit t un terme du langage de $\mathcal{S}$ dont les variables libres sont parmi $x_1, \dots, x_p$. Soit $m_1, \dots, m_p$ des éléments de $\mathcal{M}$. On a :*

$$\phi\left(\overline{t}^{\mathcal{M}}[x_1 := m_1, \dots, x_p := m_p]\right) = \overline{t}^{\mathcal{N}}[x_1 := \phi(m_1), \dots, x_p := \phi(m_p)]$$

Démonstration. Le résultat est intuitivement évident, et la preuve immédiate par induction sur la définition des termes. Comme c'est l'une des premières nous la rédigeons. Nous notons l'environnement $[\overrightarrow{x} := \overrightarrow{m}]$.

variables Si $t = x_i$, $\overline{x_i}^{\mathcal{M}}[\overrightarrow{x} := \overrightarrow{m}] = m_i$, $\overline{x_i}^{\mathcal{N}}[\overrightarrow{x} := \overrightarrow{\phi(m)}] = \phi(m_i)$ (définition de l'interpétation).

constantes Si $t = c$ est une constante de $\mathcal{S}$,

$\overline{c}^{\mathcal{M}}[\overrightarrow{x} := \overrightarrow{m}] = \overline{c}^{\mathcal{M}}$,

$\overline{c}^{\mathcal{N}}[\overrightarrow{x} := \overrightarrow{\phi(m)}] = \overline{c}^{\mathcal{N}}$,

$\phi(\overline{c}^{\mathcal{M}}) = \overline{c}^{\mathcal{N}}$ par définition d'homomorphisme.

fonctions Si $t = fu_1 \dots u_p$, où f est un symbole de fontion de $\mathcal{S}$,

$\overline{fu_1 \dots u_p}^{\mathcal{M}}[\overrightarrow{x} := \overrightarrow{m}] = \overline{f}^{\mathcal{M}}(\overline{u_1}^{\mathcal{M}}[\overrightarrow{x} := \overrightarrow{m}], \dots, \overline{u_p}^{\mathcal{M}}[\overrightarrow{x} := \overrightarrow{m}])$,

$\overline{fu_1 \dots u_p}^{\mathcal{N}}[\overrightarrow{x} := \overrightarrow{\phi(m)}] = \overline{f}^{\mathcal{N}}(\overline{u_1}^{\mathcal{N}}[\overrightarrow{x} := \overrightarrow{\phi(m)}], \dots, \overline{u_p}^{\mathcal{N}}[\overrightarrow{x} := \overrightarrow{\phi(m)}])$.

On a donc par définition de morphisme,

$\phi(\overline{fu_1 \dots u_p}^{\mathcal{M}}[\overrightarrow{x} := \overrightarrow{m}]) = \overline{f}^{\mathcal{N}}\left(\phi(\overline{u_1}^{\mathcal{N}}[\overrightarrow{x} := \overrightarrow{\phi(m)}]), \dots, \phi(\overline{u_p}^{\mathcal{N}}[\overrightarrow{x} := \overrightarrow{\phi(m)}])\right)$

et on en déduit le résultat par hypothèse d'induction sur $u_1, \dots, u_p$. ∎

Un *monomorphisme* ou *plongement* de $\mathcal{S}$-structures de $\mathcal{M}$ dans $\mathcal{N}$ est un homomorphisme injectif ϕ de $\mathcal{M}$ dans $\mathcal{N}$ qui vérifie de plus que

prédicat $(m_1, \dots, m_n) \in \overline{R}^{\mathcal{M}}$ **ssi** $(\phi(m_1), \dots, \phi(m_n)) \in \overline{R}^{\mathcal{N}}$, pour tout symbole de prédicat R d'arité n, pour tous $m_1, \dots, m_n \in \mathcal{M}$.

Un *isomorphisme* de $\mathcal{S}$-structures de $\mathcal{M}$ dans $\mathcal{N}$ est un homomorphisme bijectif ϕ de $\mathcal{M}$ dans $\mathcal{N}$ dont la réciproque est un homomorphisme de $\mathcal{S}$-structure, ou encore un plongement bijectif.

Intuitivement il devrait être clair que deux structures isomorphes vérifient les mêmes formules closes. Ce résultat se démontre par induction. Comme la définition de l'interprétation des formules closes passe par celle des formules en général, on est amené à démontrer ce lemme :

Lemme 7.3 *Soit $\mathcal{S}$ une signature, soit $\mathcal{M}$ et $\mathcal{N}$ et deux $\mathcal{S}$ structures isomorphes par ϕ. Pour toute formule F du langage du premier ordre sur $\mathcal{S}$ dont les variables libres sont parmi $x_1, \dots, x_p$, pour tout $m_1, \dots, m_p \in \mathcal{M}$, on a :*

$$\mathcal{M} \models F[x_1 := m_1, \dots, x_p := m_p] \quad \textit{ssi} \quad \mathcal{N} \models F[x_1 := \phi(m_1), \dots, x_p := \phi(m_p)]\ .$$

On en déduit, en prenant l'environnement vide, le résultat attendu :

Proposition 7.4 *Soit $\mathcal{S}$ une signature, soit $\mathcal{M}$ et $\mathcal{N}$ et deux $\mathcal{S}$ structures isomorphes par ϕ. Pour toute formule close F du langage du premier ordre sur $\mathcal{S}$*

$$\mathcal{M} \models F \quad \textit{ssi} \quad \mathcal{N} \models F\ .$$

Démonstration (lemme). On montre par induction sur la structure des formules le résultat *pour tout environnement* qui affecte toutes les variables libres de la formule. Aucune clause ne présente de réelle difficulté. À cause des clauses concernant les quantificateurs on doit démontrer par induction le résultat pour tout environnement. Passons quelques clauses en revue.

formules atomiques On utilise le lemme 7.2. Si la formule atomique est une égalité c'est une conséquence immédiate de l'injectivité de ϕ. Si la formule atomique est un prédicat, c'est une conséquence de la définition de morphisme pour ϕ et ϕ^{-1} (c'est à dire que ϕ est un plongement).

négation C'est une conséquence immédiate de l'hypothèse d'induction. On utilise la définition de la satisfaction, et le fait que l'on démontre bien une équivalence par induction.

conjonction On utilise directement la définition de la satisfaction, et l'hypothèse d'induction.

quantification universelle C'est une conséquence de l'hypothèse d'induction et de la surjectivité de ϕ. On a $F = \forall x\ F'$. Soit $[x_1 := m_1, \dots, x_p := m_p]$ un environnement qui affecte toutes les variables libres de F. Par définition de la satisfaction :

$\mathcal{M} \models \forall x\ F'[x_1 := m_1, \dots, x_p := m_p]$ ssi

$$\text{pour tout } m \text{ de } \mathcal{M},\ \mathcal{M} \models F'[x_1 := m_1, \dots, x_p := m_p, x := m] \tag{1}$$

$\mathcal{N} \models \forall x\ F'[x_1 := \phi(m_1), \ldots, x_p := \phi(m_p)]$ ssi

$$\text{pour tout } n \text{ de } \mathcal{N},\ \mathcal{N} \models F'[x_1 := \phi(m_1), \ldots, x_p := \phi(m_p), x := n] \tag{2}$$

On doit démontrer que (1) $\Leftrightarrow$ (2). L'hypothèse d'induction pour l'environnement $[x_1 := m_1, \ldots, x_p := m_p, x := m]$, donne :

$$\begin{array}{c} \mathcal{M} \models F'[x_1 := m_1, \ldots, x_p := m_p, x := m] \\ \text{ssi} \\ \mathcal{N} \models F'[x_1 := \phi(m_1), \ldots, x_p := \phi(m_p), x := \phi(m)] \end{array}$$

(1) $\Rightarrow$ (2) Soit n un élément quelconque de $\mathcal{N}$, comme ϕ est surjective, il existe m dans $\mathcal{M}$ tel que $\phi(m) = n$. On utilise l'hypothèse d'induction pour l'environnement $[x_1 := m_1, \ldots, x_p := m_p, x := m]$.

(2) $\Rightarrow$ (1) Soit m un élément quelconque de $\mathcal{M}$, on utilise l'hypothèse d'induction pour l'environnement $[x_1 := m_1, \ldots, x_p := m_p, x := m]$.

Les autre cas se démontrent essentiellement de la même façon, par exemple la clause pour l'existentielle utilise également la surjectivité de ϕ. ■

7.7 Conséquence sémantique.

Soit $\mathcal{S}$ une signature, T une théorie (c'est à dire un ensemble de formules closes) du langage de signature $\mathcal{S}$, soit Γ un ensemble *fini* de formules et F du même langage. Nous dirons que Γ *a pour conséquence sémantique F dans la théorie T* quand pour tout modèle $\mathcal{M}$ de T, tout environnement $[\overrightarrow{x} := \overrightarrow{m}]$ dans $\mathcal{M}$ qui valide chacune des formules de Γ, $F[\overrightarrow{x} := \overrightarrow{m}]$ est valide :

$$\text{pour tout } \mathcal{M} \ \text{ si } \mathcal{M} \models T \text{ et } \mathcal{M} \models \Gamma[\overrightarrow{x} := \overrightarrow{m}] \text{ alors } \mathcal{M} \models F[\overrightarrow{x} := \overrightarrow{m}]$$

On notera ultérieurement $\Gamma \vdash_T F$. Cependant, tant que nous n'avons pas démontré le théorème de complétude, nous noterons cette relation $\Gamma \mid\!\sim_T F$. Quand la théorie T est vide, on dit simplement que Γ a pour conséquence sémantique F, et l'on note $\Gamma \mid\!\sim F$. Quand l'ensemble Γ est vide on dit simplement que F est une conséquence sémantique de la théorie T et l'on note $\mid\!\sim_T F$. Remarquons qu'une formule close F est conséquence sémantique d'une théorie T quand tout modèle de T est modèle de F.

Une conséquence immédiate de la définition de la satisfaction pour $\perp$ est que :

$$\text{Une théorie } T \text{ est incohérente ssi } \mid\!\sim_T \perp .$$

Nous avons déjà vu la définition de formule universellement valide, remarquons qu'un formule close est universellement valide ssi $\mid\!\sim F$.

Une conséquence immédiate de la définition est que si Γ est un ensemble de formules closes :

$$\Gamma \mid\!\sim_T F \ \text{ ssi } \ \mid\!\sim_{T \cup \Gamma} F .$$

On dira que deux formules F et G *sont sémantiquement équivalentes* dans la théorie T quand $F \mid\!\sim_T G$ et $G \mid\!\sim_T F$, et l'on notera $F \equiv_T G$.

On utilise le signe $\Leftrightarrow$ pour l'équivalence en tant que connecteur, qui est définie de façon usuelle par :

$$F \Leftrightarrow G = (F \Rightarrow G) \wedge (G \Rightarrow F) .$$

Une conséquence immédiate des définitions qui précèdent et de la définition de la satisfaction est que pour toute théorie T, pour toute formules F et G :

$$F \equiv_T G \ \text{ ssi } \ \mid\!\sim_T F \Leftrightarrow G .$$

On généralise ces définitions aux théories : une théorie T' est conséquence d'une théorie T si toute formule de T' est conséquence de T, deux théories T et T' sont équivalentes si T' est conséquence de T et T conséquence de T'.

8 Calcul des prédicats.

8.1 Introduction.

Le but de ce court chapitre est essentiellement de donner quelques schémas de formules universelles qui concernent les quantificateurs.

Bien que nous n'ayons pas encore démontré le théorème de complétude, nous notons "$\vdash$" la déduction, qu'elle soit syntaxique ou sémantique.

8.2 Quelques équivalences logiques et formules universellement valides.

La signature du langage est quelconque. On désigne par F et G des formules quelconques.

Quantifications "inutiles" :

Si x n'apparait pas libre dans F :

$$\forall x\ F \equiv F \quad \exists x\ F \equiv F \tag{1}$$

Commutation des quantificateurs :

$$\forall x \forall y\ F \equiv \forall y \forall x\ F \quad \exists x \exists y\ F \equiv \exists y \exists x\ F$$

$$\exists x \forall y\ F \vdash \forall y \exists x\ F$$

mais en général (cela dépend de F) la réciproque est fausse :

$$\forall y \exists x\ F \nvdash \exists x \forall y\ F$$

Négation des quantificateurs :

$$\neg \forall x\ F \equiv \exists x \neg F \quad \neg \exists x\ F \equiv \forall x \neg F \tag{2}$$

Conjonction et disjonction :

$$\forall x (F \wedge G) \equiv \forall x\ F \wedge \forall x\ G \quad \exists x (F \vee G) \equiv \exists x\ F \vee \exists x\ G \tag{3}$$

$$\exists x (F \wedge G) \vdash \exists x\ F \wedge \exists x\ G \quad \forall x\ F \vee \forall x\ G \vdash \forall x (F \vee G)$$

Mais "en général" (cela dépend des formules F et G) la réciproque est fausse :

$$\exists x\ F \wedge \exists x\ G \nvdash \exists x (F \wedge G) \quad \forall x (F \vee G) \nvdash \forall x\ F \vee \forall x\ G$$

par contre si x n'apparaît pas dans G :

$$\exists x (F \wedge G) \equiv \exists x\ F \wedge G \quad \forall x (F \vee G) \equiv \forall x\ F \vee G \tag{4}$$

Remarquez que l'on pouvait déjà déduire des équivalences précédentes que si x n'apparaît pas dans G :

$$\exists x (F \vee G) \equiv \exists x\ F \vee G \quad \forall x (F \wedge G) \equiv \forall x\ F \wedge G$$

Implication :

$$\exists x (F \to G) \equiv \forall x\ F \to \exists x\ G$$

$$\exists x\ F \to \forall x\ G \vdash \forall x (F \to G)$$

mais en général (cela dépend de F et G) la réciproque est fausse :

$$\forall x (F \to G) \nvdash \exists x\ F \to \forall x\ G$$

par contre si x n'apparait pas libre dans F :

$$\forall x(F \to G) \equiv F \to \forall x\ G \tag{5}$$

et si x n'apparait pas libre dans G :

$$\forall x(F \to G) \equiv \exists x\ F \to G \tag{6}$$

On montre ces équivalences, soit en utilisant la déduction, soit en reprenant la définition sémantique.

À l'aide des équivalences (2) qui précèdent, et des résultats vus en calcul propositionnel, on montre facilement que toute formule de la logique du premier ordre est équivalente à une formule qui n'utilise que $\exists, \wedge, \neg$, ou encore $\forall, \to, \bot$. On peut donc se restreindre à l'un des ces ensembles de quantificateurs et connecteurs pour démontrer des propriétés sémantiques des formules. On ne peut pas vraiment parler de systèmes complets pour la logique du premier ordre. Par exemple des quantifications comme "Il existe une infinité de x tels que F" ne s'exprime pas dans le langage du premier ordre.

Par contre il est bien connu que l'on peu exprimer les quantificateurs "il existe au plus un x tel que F" et "il existe un unique x tel que F" avec l'égalité :

$$!x\ F \equiv_d \forall y \forall y'(F[y/x] \to F[y'/x] \to y = y')\ .$$

$$\exists !x\ F \equiv_d \exists x\ F \wedge !x\ F\ .$$

On montre facilement que :

$$\begin{aligned} \exists !x\ F &\equiv \exists x(F \wedge \forall y(F[y/x] \to y = x))\ . \\ \neg !x\ F &\equiv \exists y \exists y'(F[y/x] \wedge F[y'/x] \wedge y \neq y')\ . \\ \neg \exists !x\ F &\equiv \forall x \neg F \vee \exists y \exists y'(F[y/x] \wedge F[y'/x] \wedge y \neq y')\ . \end{aligned}$$

Il est possible également d'exprimer "il existe au plus un couple (x, y) tel que F", "il existe un unique couple (x, y) tel que F", de généraliser aux n-uplets (exercice).

8.3 Formes prénexes, formes de Skolem

Une formule F est dite *sous forme prénexe* quand les quantificateurs n'apparaissent dans F qu'en tête de la formule, c'est à dire que F s'écrit :

$$Q_1 x_1 \ldots Q_n x_n\ F_0$$

où F_0 est purement propositionnelle et les Q_i sont des quantificateurs $\forall$ ou $\exists$.

En utilisant les équivalences du paragraphe 8.2, plus précisément celles qui sont numérotées, on montre que :

Proposition 8.1 *Toute formule F d'un langage $\mathcal{L}$ est équivalente à une formule du même langage sous forme prénexe.*

Démonstration. On prouve le résultat par induction sur la structure des formules.

Formules atomiques, $\bot$. Les formules atomiques de $\mathcal{L}$ sont sous forme prénexe, de même $\bot$.

négation Si F est une formule équivalente à la forme prénexe :

$$F \equiv Q_1 x_1 \ldots Q_n x_n\ F_0$$

alors en utilisant les équivalences (2) du paragraphe 8.2

$$\neg F \equiv Q_1^* x_1 \ldots Q_n^* x_n\ \neg F_0$$

où Q_i^* est le connecteur dual de Q_i (Q_i^* est $\forall$, si Q_i est $\exists$, Q_i^* est $\exists$, si Q_i est $\forall$).

conjonction Si F et G sont des formules chacune équivalente à une forme prénexe :

$$F \equiv Q_1x_1 \dots Q_nx_n\ F_0 \qquad G \equiv Q'_1x'_1 \dots Q'_mx'_m\ G_0$$

on suppose de plus, modulo renommage des variables liées, que les ensembles de variables $x_1, \dots, x_n$ et $x'_1, \dots, x'_m$ sont disjoints. On obtient alors, en utilisant les équivalences (1) et parmi (3) et (4) celles qui concernent la conjonction :

$$F \wedge G \equiv Q_1x_1 \dots Q_nx_nQ'_1x'_1 \dots Q'_mx'_m(F_0 \wedge G_0)\ .$$

On remarque d'ailleurs que l'ordre entre les quantificateurs Q_i et Q'_i n'a pas d'importance : on peut tout aussi bien placer les Q'_i en premier, les alterner, le tout et de conserver l'ordre entre les Q_i et l'ordre entre les Q'_i. Il pourrait y avoir par ailleurs des mises sous forme prénexe plus économique, par exemple utilisant (3) directement.

disjonction analogue au cas précédent.

implication analogue aux deux cas précédents, il faut utiliser les équivalences (5) et (6).

quantifications Si F est une formule équivalente à une forme prénexe et x une variable, alors $\forall x\ F$ et $\exists x\ F$ sont évidemment équivalentes à des formules sous forme prénexe. ■

Il est clair qu'il n'y a pas unicité de la forme prénexe, ne serait-ce qu'à cause des équivalences (1). On peut essayer de minimiser le nombre de quantificateurs dans la mise sous forme normale, et également de minimiser le nombre d'alternances de groupe de quantificateurs de même nature (nous dirons alternances de quantificateurs) ce qui n'assure pas plus l'unicité. Intuitivement, le langage étant fixé, plus une formule prénexe a d'alternances de quantificateurs, plus elle est "compliquée" à comprendre et à démontrer. Par exemple l'énoncé de l'injectivité de f définie de A dans B, $\forall x \in A \forall y \in A (f(x) = f(y) \Rightarrow x = y)$ n'a pas d'alternances de quantificateurs, et semble plus "simple" que l'énoncé de la surjectivité de f, $\forall y \in B \exists x \in A (f(x) = y)$ qui contient une alternance de quantificateur. Les énoncés exprimant l'existence d'une limite en un point ont plus d'alternances et semblent plus "compliqués". Tout ceci n'est pas à prendre au pied de la lettre. Les alternances de quantificateurs fournissent un cadre par exemple pour analyser la complexité des sous-ensembles de $\mathbb{N}$ qui sont définissables par une formule de l'arithmétique, la complexité de certains problèmes algorithmiques etc.

En composant avec les résultats de mise sous forme normale en calcul propositionnel, on obtient que toute formule est équivalente à une formule sous forme prénexe dont la partie propositionnelle est sous forme normale disjonctive, ou conjonctive.

On ne peut faire disparaître par équivalence les alternances de quantificateurs. Il est par contre possible d'obtenir de telles formules par *équisatisfaisabilité*, en étendant le langage (et donc cela ne contredit pas ce qui précède). Deux formules F et G sont *équisatisfaisables* quand F a un modèle ssi G a un modèle.

Étant donné une formule F du langage $\mathcal{L}$ il est possible de trouver une formule équisatisfaisable F' d'un langage $\mathcal{L}'$ qui étend $\mathcal{L}$, qui est sous forme prénexe et ne contient que des quantificateurs universels. On construit F' à partir d'une forme prénexe de F en ajoutant un symbole de fonction f_i pour chaque quantificateur existentiel $\exists x_i$ qui a autant d'arguments que de quantificateurs universels antérieurs dans la forme prénexe, soient $x_{j_1}, \dots, x_{j_k}$, et en remplaçant dans la partie propositionnelle de la forme prénexe de F chaque occurrence de variable x_i par le terme $f_ix_{j_1} \dots x_{j_k}$. On appelle une formule ainsi construite *forme de Skolem de F*.

Par exemple, prenons pour langage (f), une forme de Skolem de $\forall y \exists x\ fx = y$ est $\forall y\ fgy = y$ formule du langage (f, g). Intuitivement g est une fonction réciproque à droite de f. Un modèle de cette dernière formule fournit évidemment un modèle de la première formule (en "oubliant" l'interprétation de g). Un modèle $\mathcal{M} = (M, \overline{f})$ de la première formule fournit un modèle de sa forme de Skolem : on construit l'interprétation de g en choisissant pour chaque élément m de M un élément m' de M qui vérifie $\mathcal{M} \models fx = y[x := m, y := m']$ (on utilise l'axiome du choix si M est infini).

On a exposé sur un cas particulier très simple la démonstration de la proposition suivante :

Proposition 8.2 *Si G est une forme de Skolem de F, F et G sont équisatisfaisables.*

8.4 Un exemple d'élimination des quantificateurs.

Donnons un exemple d'utilisation de la mise sous-forme normale en calcul propositionnel vue au chapitre précédent, et de la mise sous forme prénexe.

Dans certaines théories, il est possible de montrer que toute formule est équivalente à une formule sans quantificateurs, on dit alors qu'une telle théorie admet *l'élimination des quantificateurs*. L'intérêt d'un tel résultat est assez évident, et cela a des conséquences importantes que nous verrons ensuite.

Un des exemples les plus simples d'un telle théorie est la théorie des *ordres totaux denses sans extrémités*. On exprime cette théorie dans le langage de signature $(<)$ (ordre strict). Les axiomes sont :

anti-réflexivité $\forall x \neg x < x$;

transitivité $\forall x \forall y \forall z (x < y \to y < z \to x < z)$;

totalité $\forall x \forall y (x < y \vee x = y \vee y < x)$;

densité $\forall x \forall y (x < y \to \exists a (x < a \wedge a < y)$;

sans majorant $\forall x \exists y\ x < y$;

sans minorant $\forall x \exists y\ y < x$.

Clairement $(\mathbb{Q}, <)$ et $\mathbb{R}, <)$ sont des modèles de cette théorie.

Appelons T cette théorie. En plus de $\perp$, on va considèrer également $\top$ comme un élément primitif du langage (juste pour ce paragraphe).

Il est commode dans un premier temps d'éliminer les négations, ce que permet le lemme suivant :

Lemme 8.3 *On vérifie pour toutes variables x et y que :*

$$\neg x < y \equiv_T y < x \vee y = x$$
$$\neg x = y \equiv_T x < y \vee y < x$$

Démonstration. C'est une conséquence immédiate de la totalité. ■

On en déduit :

Lemme 8.4 *Toute formule propositionnelle du langage $(<)$ est équivalente dans la théorie des ordres stricts totaux à une formule qui utilise les mêmes variables libres et qui est, ou bien sous forme normale disjonctive utilisant pour seuls connecteurs $\wedge, \vee$, ou bien $\perp$, ou bien $\top$.*

Démonstration. Soit F une formule. Tout d'abord F est équivalente à une formule F_0 sous forme normale disjonctive. Cette forme normale disjonctive utilise les mêmes atomes que F, donc les mêmes variables libres. On a considéré $\perp$ et $\top$ comme des cas dégénérés de forme normale disjonctive.

On suppose que F_0 n'est ni $\perp$ ni $\top$. Dans F_0 les négations apparaissent devant une formule atomique, soit sous une des deux formes $\neg x < y$ ou $\neg x = y$. En utilisant le lemme précédent, on obtient par substitution une formule F_1 équivalente à F_0 et qui n'utilise que les connecteurs $\wedge$ et $\vee$ (sans négation). On montre ensuite que F_1 est équivalente à une forme normale disjonctive sans négation par induction sur la structure de F_1, en utilisant la distributivité du $\wedge$ sur le $\vee$. ■

Une formule du langage des ordres est donc équivalente dans la théorie T à une forme prénexe dont la partie propositionnelle est comme au lemme précédent. Le lemme clef pour l'élimination des quantificateurs est le suivant :

Lemme 8.5 (lemme clef) *Soit F une formule propositionnelle du langage de signature $(<)$ utilisant pour seul connecteur $\wedge$, et dont les variables sont parmi $x_0, x_1, \ldots, x_n$. Alors il existe une formule propositionnelle F' du même langage, utilisant pour seul connecteur $\wedge$ dont les variables libres sont parmi $x_1, \ldots, x_n$ et telle que :*

$$\exists x_0\ F \equiv_T F' \ .$$

Démonstration. L'idée de la preuve est simple. Une fois les conditions ne concernant pas x_0 réalisées, il est toujours possible de trouver un tel x_0, en vertu des axiomes de densité, d'absence de majorant et de minorant.

Détaillons tout d'abord la construction de la formule F'.

– Il y a un nombre fini de formules $x_i < x_j$, $x_i = x_j$, on peut donc définir la formule F_0 obtenue, tout d'abord en ajoutant à F toutes les formules $x_i < x_j$ et $x_i = x_j$ qui sont conséquence de F dans la théorie T, sauf les égalité $x_i = x_i$:

$$F_0 = \left(\bigwedge_{\substack{i,j=0 \\ F \vdash_T x_i < x_j}}^{n} x_i < x_j \right) \wedge \left(\bigwedge_{\substack{0 \leqslant i < j \leqslant n \\ F \vdash_T x_i = x_j}} x_i = x_j \right)$$

clairement F_0 contient en particulier chaque formule atomique qui apparaît dans F et donc $F \equiv F_0$. Si F_0 contient une formule $x_i < x_i$, les relations induites par les égalités et inégalités de la formule F sur $x_1, \ldots, x_n$ ne sont pas réalisables parce qu'il y a un cycle (anti-réflexivité). La formule F est équivalente à $\perp$, la formule $\exists x_0\ F$ également.

– Si la variable x_0 apparait dans une égalité de F_0 (et donc de F) : $F \vdash x_0 = x_i$, alors la formule $\exists x_0\ F$ est équivalente à F' obtenue à partir de F_0 en supprimant toutes les formules où x_0 apparait. On pourrait également remplaçer dans F toutes les occurrences de x_0 par x_i (c'est "déjà fait" dans F_0).

– Dans tous les autres cas la formule F est équivalente à la formule F' obtenue en supprimant dans F_0 toutes les formules atomiques où x_0 apparait. On prend $F' = \top$ si ceci supprime toutes les formules. On utilise pour cela soit l'absence de minorant, soit l'absence de majorant, soit la densité.

Donnons quelques exemples avant la démonstration.

$\exists y(x_1 < y \wedge y < x_2 \wedge x_1 = x_2) \equiv_T \perp$	cycle
$\exists y(x_1 < y \wedge x_3 < x_2 \wedge y = x_3) \equiv_T x_1 < x_3 \wedge x_3 < x_2$	égalité
$\exists y(x_1 < y \wedge y < x_2 \wedge x_1 = x_3) \equiv_T x_1 < x_2 \wedge x_3 < x_2 \wedge x_1 = x_3$	densité
$\exists y(x_1 < y \wedge x_2 < y) \equiv_T \top$	pas de majorant
$\exists y(y < x_1 \wedge x_3 < x_2 \wedge x_1 = x_3) \equiv_T x_1 < x_2 \wedge x_3 < x_2 \wedge x_1 = x_3$	pas de minorant

Précisons la preuve. On suppose que F est satisfaisable. La formule F' est obtenue comme indiqué ci-dessus : par construction $\exists x_0\ F \vdash F'$. Voyons la réciproque. On se donne dans un modèle quelconque $\mathcal{M}$ un environnement $[\overrightarrow{x_i} := \overrightarrow{m_i}]$ pour $x_1, \ldots, x_n$ les variables libres de $\exists x_0\ F$ (donc pour les variables libres de F'). Si $F'[\overrightarrow{x_i} := \overrightarrow{m_i}]$ est satisfaite, on montre que $F[\overrightarrow{x_i} := \overrightarrow{m_i}]$ est alors satisfaite. Si x_0 apparait dans au moins une égalité $x_0 = x_i$, on interprète x_0 par m_i (la construction de F' assure que toutes les égalités et inégalités où x_0 apparait sont alors satisfaites). Si x_0 n'apparait que dans des inégalités, à droite et à gauche du signe $<$, on obtient une interprétation pour x_0 par densité entre m_{i_0} le plus grand des m_i tel que $x_i < x_0$ apparait dans F', et m_{i_1} le plus petit des m_i tel que $x_0 < x_i$ apparait dans F'. On a bien que $m_{i_0} < m_{i_1}$. En effet comme les formules $x_{i_0} < x_0$ et $x_0 < x_{i_1}$, apparaissent dans F', par construction de F', la formule $x_{i_0} < x_{i_1}$ apparait dans F' (T est transitive). Or la conjonction F' est satisfaite dans l'environnement $[\overrightarrow{x_i} := \overrightarrow{m_i}]$. Si x_0 n'apparait que dans des inégalités, à droite du signe $<$, on utilise de façon analogue l'absence de majorant. Si x_0 n'apparait n'apparait que dans des inégalités, à gauche du signe $<$ on utilise de façon analogue l'absence de minorant. ■

Proposition 8.6 *Une formule du premier ordre du langage de* $(<)$ *est équivalente dans la théorie* T *des ordres denses sans extrémités à une formule sans quantificateurs dont les variables libres sont parmi les variables libres de* F.

Démonstration. On le montre pour les formules sous forme prénexe par récurrence sur le nombre d'occurrences de quantificateurs, soit n.

$n = 0$**.** Cette forme normale est sans quantificateurs.

$n \to n+1$. Soit F une formule sous forme prénexe avec $n+1$ occurences de quantificateurs. Deux cas sont possibles.

$F = \exists x G$. Par hypothèse de récurrence, G est équivalente à une formule sans quantificateurs dont les variables libres sont parmi celles de F ou x. D'après le lemme 8.4, G est équivalente à une formule $\bigvee_{i=1}^{n} G_i$ où les G_i sont des conjonctions de formules atomiques qui ont leurs variables libres parmi celles de G. On a donc

$$\exists x F \equiv \bigvee_{i=1}^{n} \exists x\ G_i\ .$$

D'après le lemme 8.5, chacune des formules $\exists x\ G_i$ est équivalente à une formule sans quantificateurs dont les variables libres sont parmi celles de F, d'où le résultat.

$F = \forall x G$. Par hypothèse de récurrence, G est équivalente à une formule sans quantificateurs dont les variables libres sont parmi celles de F ou x. On a $F \equiv \neg\exists x \neg G$. On procède pour $\exists x \neg G$ comme au cas précédent, d'où le résultat pour F. ■

Corollaire 8.7 *Pour toute formule close du premier ordre F dans le langage $(<)$,*

$$\vdash_T F \text{ ou } \vdash_T \neg F\ .$$

Démonstration. En effet d'après le lemme précédent F est équivalente à une formule sans quantificateurs ni variables libres, qui ne peut donc être que $\top$ ou $\bot$, car le langage n'a pas de constantes. ■

Une telle théorie, c'est à dire une théorie T vérifiant pour toute formule F que $\vdash_T F$ ou $\vdash_T \neg F$, est dite *complète*.

Les modèles d'une théorie complète T vérifient les mêmes formules closes du premier ordre : celles qui sont démontrables dans la théorie T. Par exemple $(\mathbb{Q}, <)$ et $(\mathbb{R}, <)$ satisfont les mêmes formules closes (alors ces deux structures dont les ensembles de base n'ont pas même cardinalité, ne sont pas isomorphes).

Il existe des théories qui ne sont évidemment pas complètes, comme la théorie des ordres, la théorie des groupes etc.

Il existe des théories qui ont un modèle «attendu» et dont on pourrait souhaiter qu'elles soient complètes, mais qui ne le sont pas. C'est le cas par exemple de l'arithmétique de Peano, ou de la théorie des ensembles. Cela signifie par exemple que l'arithmétique de Peano a des modèles qui ne vérifient pas les mêmes formules que $\mathbb{N}$. C'est une conséquence du premier théorème d'incomplétude de Gödel, que l'on ne démontrera pas ici. En fait, d'après le premier théorème d'incomplétude de Gödel, il n'y a pas de théorie « que l'on puisse énoncer de façon raisonnable[17] », qui soit cohérente, qui permette de développer « suffisament » d'arithmétique (comme l'arithmétique de Peano ou la théorie des ensembles), et qui soit complète.

Il existe cependant des théories complètes intéressantes, comme la théorie des corps algébriquement clos, l'arithmétique de Pressburger (essentiellement l'arithmétique avec l'addition comme seule opération) ...

[17] informellement on doit pouvoir reconnaître « mécaniquement » si une formule est ou non un axiome, on sait formaliser ceci.

9 La relation de déduction du calcul des prédicats : adéquation et complétude.

Le but de ce chapitre est de montrer que les notions syntaxique et sémantique de déduction coïncident.

9.1 Quelques préliminaires sur la relation de déduction.

On rappelle que la relation de déduction $\Gamma \vdash_T C$ entre un multi-ensemble de formules propositionnelles Γ et une formule C est la relation dont la définition inductive est donnée par les règles de la figure 5 page 18 et de la figure 6 page 19.

On retrouve les propriétés de la relation de déduction énoncées dans le cadre du calcul propositionnel au paragrahe 5.1 page 38, qui se justifient de la même façon qu'en calcul propositionnel.

Proposition 9.1 *Pour toutes théories T et T', toute formule C,*

$$\textit{si} \;\; T \subset T', \;\; \textit{et} \;\; \Gamma \vdash_T C \;\; \textit{alors} \;\; \Gamma \vdash_{T'} C \ .$$

Proposition 9.2 *Pour toute théorie T, toute formule close A, toute formule C,*

$$\Gamma \vdash_{T \cup \{A\}} C, \quad \textit{ssi} \;\; \Gamma, A \vdash_T C \ .$$

La propriété de finitude 5.7 page 39 reste valide en calcul des prédicats, pour les mêms raisons qu'en calcul propositionnel :

Proposition 9.3 (finitude) *Si $\vdash_T C$, il existe une partie finie de T, soit T_0, telle que $\vdash_{T_0} C$.*

9.2 La propriété d'adéquation.

La notion de déduction sémantique définie pour une théorie T un ensemble de formules Γ et une formule F (voir paragraphe 7.7), s'étend évidemment à un multi-ensemble Γ : on dira que $\Gamma \vdash_T F$ pour un multi-ensemble Γ quand on a $\Gamma' \vdash_T F$ pour l'ensemble Γ' support de Γ (Γ dans lequel on a éliminé les répétitions).

Comme en calcul propositionnel (lemme 4.3), on va montrer par induction sur la définition de la relation de déduction, c'est à dire en vérifiant règle par règle le résultat, que la déduction "conserve" la satisfaction dans une structure, c'est à dire que la déduction syntaxique entraîne la déduction sémantique. C'est la *propriété d'adéquation* qu'énonce le lemme suivant.

Lemme 9.4 (adéquation) *Pour toute théorie T, tout ensemble fini de formules Γ, toute formule C dans le langage L,*

$$\textit{si}\; \Gamma \vdash_T C, \quad \textit{alors}\; \Gamma \mathrel{|\!\!\sim}_T C \ .$$

On obtient comme cas particulier :

Proposition 9.5 (adéquation) *Pour toute théorie T, toute formule close C dans le langage L, si $\vdash_T C$ alors $\mathrel{|\!\!\sim}_T C$. Autrement dit, pour toute L-structure $\mathcal{M}$:*

$$\textit{Si} \;\; \vdash_T C \;\; \textit{alors} \;\; \mathcal{M} \models T \Rightarrow \mathcal{M} \models C \ .$$

Démonstration (lemme). La preuve par induction du lemme est de pure routine. détaillons tout de même quelques cas pour comprendre ce genre de preuve. Soit $\mathcal{M}$ une $\mathcal{L}$-structure, telle que $\mathcal{M} \models T$. On écrit $[\overrightarrow{x} := \overrightarrow{m}]$ pour désigenr un environnement $x_1 := m_1, \ldots, x_n := m_n$.

On va montrer par induction sur la déduction de $\Gamma \vdash_T C$ que pour tout environnement $[\overrightarrow{x} := \overrightarrow{m}]$ dans $\mathcal{M}$ qui affecte toutes les variables libres des formules considérées (dans la suite nous appelons "convenable" un tel environnement), si $\mathcal{M} \models \Gamma[\overrightarrow{x} := \overrightarrow{m}]$ alors $\mathcal{M} \models C[\overrightarrow{x} := \overrightarrow{m}]$.

Axiomes de la déduction On a $A \vdash_T A$, on doit montrer pour un environnement convenable quelconque que si $\mathcal{M} \models A[\overrightarrow{x} := \overrightarrow{m}]$, alors $\mathcal{M} \models A[\overrightarrow{x} := \overrightarrow{m}]$, ce qui est tautologique !

Axiomes de la théorie $\mathcal{M} \models T$ signifie $\mathcal{M} \models F$ pour chaque formule close F de T.

Contraction, affaiblissement Pour la contraction on utilise directement l'hypothèse d'induction (c'est la définition de la satisfaction d'un multi-ensemble), pour l'affaiblissement, il est évident que si $\mathcal{M} \models \Gamma, A[\overrightarrow{x} := \overrightarrow{m}]$, alors $\mathcal{M} \models \Gamma[\overrightarrow{x} := \overrightarrow{m}]$ et on conclut par hypothèse d'induction.

Introduction de l'implication Cette règle étant la dernière appliquée, par hypothèse d'induction on a pour tout environnement convenable, si $\mathcal{M} \models \Gamma, A[\overrightarrow{x} := \overrightarrow{m}]$ alors $\mathcal{M} \models B[\overrightarrow{x} := \overrightarrow{m}]$. On doit montrer que si $\mathcal{M} \models \Gamma[\overrightarrow{x} := \overrightarrow{m}]$ alors $\mathcal{M} \models (A \to B)[\overrightarrow{x} := \overrightarrow{m}]$. Par définition de la satisfaction pour l'implication $\mathcal{M} \models (A \to B)[\overrightarrow{x} := \overrightarrow{m}]$ signifie que si $\mathcal{M} \models A[\overrightarrow{x} := \overrightarrow{m}]$ alors $\mathcal{M} \models B[\overrightarrow{x} := \overrightarrow{m}]$ d'où le résultat.

Élimination de l'implication Cette règle étant la dernière appliquée, par hypothèse d'induction on a pour tout environnement convenable, si $\mathcal{M} \models \Gamma[\overrightarrow{x} := \overrightarrow{m}]$ alors $\mathcal{M} \models (A \to B)[\overrightarrow{x} := \overrightarrow{m}]$, et que si $\mathcal{M} \models \Delta[\overrightarrow{x} := \overrightarrow{m}]$ alors $\mathcal{M} \models A[\overrightarrow{x} := \overrightarrow{m}]$. On en déduit que si $\mathcal{M} \models \Gamma, \Delta[\overrightarrow{x} := \overrightarrow{m}]$, alors $\mathcal{M} \models (A \to B)[\overrightarrow{x} := \overrightarrow{m}]$, et $\mathcal{M} \models A[\overrightarrow{x} := \overrightarrow{m}]$, et donc par définition de la satisfaction pour l'implication, $\mathcal{M} \models B[\overrightarrow{x} := \overrightarrow{m}]$, ce qui est le résultat cherché.

Élimination de l'absurde Par hypothèse d'induction $\mathcal{M} \models \bot$, mais ceci n'est pas possible par définition de la satisfaction. On en déduit donc n'importe quoi, par exemple que $\mathcal{M} \models A[\overrightarrow{x} := \overrightarrow{m}]$.

Raisonnement par l'absurde Par hypothèse d'induction on a pour tout environnement convenable, si $\mathcal{M} \models \Gamma, \neg A[\overrightarrow{x} := \overrightarrow{m}]$ alors $\mathcal{M} \models \bot[\overrightarrow{x} := \overrightarrow{m}]$. Mais $\mathcal{M} \models \bot$ est exclu par définition de la satisfaction, donc $\mathcal{M} \not\models \Gamma, \neg A[\overrightarrow{x} := \overrightarrow{m}]$. Cela a pour conséquence que si $\mathcal{M} \models \Gamma[\overrightarrow{x} := \overrightarrow{m}]$ alors $\mathcal{M} \not\models \neg A[\overrightarrow{x} := \overrightarrow{m}]$, c'est à dire par définition de la satisfaction de la négation, $\mathcal{M} \models A[\overrightarrow{x} := \overrightarrow{m}]$, ce qui est le résultat cherché.

Introduction du quantificateur universel On doit montrer que pour tout environnement convenable pour Γ et $\forall x\ A$, soit $[\overrightarrow{x} := \overrightarrow{m}]$, si $\mathcal{M} \models \Gamma[\overrightarrow{x} := \overrightarrow{m}]$ alors $\mathcal{M} \models \forall x\ A[\overrightarrow{x} := \overrightarrow{m}]$. Soit n un élément quelconque de $\mathcal{M}$. L'environnement $[\overrightarrow{x} := \overrightarrow{m}, y := n]$ convient pour Γ et $A[y/x]$. Du fait que y n'apparait pas libre dans Γ, $\mathcal{M} \models \Gamma[\overrightarrow{x} := \overrightarrow{m}]$ signifie $\mathcal{M} \models \Gamma[\overrightarrow{x} := \overrightarrow{m}, y := n]$. Par hypothèse d'induction on en déduit $\mathcal{M} \models A[y/x][\overrightarrow{x} := \overrightarrow{m}, y := n]$, ceci pour tout élément n de $\mathcal{M}$ ce qui par définition de la satisfaction pour la quantification universelle, signifie que $\mathcal{M} \models \forall x\ A[\overrightarrow{x} := \overrightarrow{m}]$ ce qui est le résultat cherché.

Élimination du quantificateur universel Étant donné un environnement convenable pour Γ et $A[t/y]$, soit $[\overrightarrow{x} := \overrightarrow{m}]$, on doit montrer que si $\mathcal{M} \models \Gamma[\overrightarrow{x} := \overrightarrow{m}]$ alors $\mathcal{M} \models A[t/y][\overrightarrow{x} := \overrightarrow{m}]$. Si $[\overrightarrow{x} := \overrightarrow{m}]$ est un environnement convenable pour $A[t/y]$, c'est un environnement convenable pour $\forall y\ A$. En effet, toute variable qui apparait libre dans A et qui est distincte de y apparait libre dans $A[t/y]$. On a donc par définition de la satisfaction pour le quantificateur universel, $\mathcal{M} \models A[\overrightarrow{x} := \overrightarrow{m}, y := n]$, ce pour n'importe quel élément de n de $\mathcal{M}$, en particulier $n = \overline{t}^{\mathcal{M}}[\overrightarrow{x} := \overrightarrow{m}]$. On en déduit par une induction évidente sur la définition des formules que $\mathcal{M} \models A[t/y][\overrightarrow{x} := \overrightarrow{m}]$.

Réflexivité de l'égalité Immédiat par définition de la satisfaction dans le cas de l'égalité.

Élimination de l'égalité Étant donné un environnement convenable pour Γ, Δ et $A[t'/y]$, on doit montrer que si $\mathcal{M} \models \Gamma, \Delta[\overrightarrow{x} := \overrightarrow{m}]$, alors $\mathcal{M} \models A[t'/y][\overrightarrow{x} := \overrightarrow{m}]$. Complétons $[\overrightarrow{x} := \overrightarrow{m}]$ en $[\overrightarrow{x} := \overrightarrow{m}, \overrightarrow{z} = \overrightarrow{n}]$, où les z_i sont d'éventuelles variables qui apparaissent dans t mais pas dans Γ, Δ et $A[t'/x]$. On sait par hypothèse d'induction que si $\mathcal{M} \models \Gamma$, donc en particulier si $\mathcal{M} \models \Gamma, \Delta$ alors $\mathcal{M} \models A[t/y][\overrightarrow{x} := \overrightarrow{m}, \overrightarrow{z} = \overrightarrow{n}]$ et $\mathcal{M} \models t = t'[\overrightarrow{x} := \overrightarrow{m}, \overrightarrow{z} = \overrightarrow{n}]$. On en déduit (formellement il faudrait le montrer par induction sur la définition de la formule A) en utilisant la définition de la satisfaction pour l'égalité que $\mathcal{M} \models A[t'/y][\overrightarrow{x} := \overrightarrow{m}, \overrightarrow{z} = \overrightarrow{n}]$, et donc comme les variables z_i n'apparaissent pas dans $A[t'/y]$, $\mathcal{M} \models A[t'/y][\overrightarrow{x} := \overrightarrow{m}]$

Les règles pour les connecteurs et quantificateur restant se traitent de façon analogue et sont laissées au lecteur. ■

On s'aperçoit que dans chacun des cas traités, on a essentiellement utilisé dans le meta-langage …la règle de déduction dont on montre la validité. Les seules difficultés de la preuve sont bien d'énonciation. Ce n'est pas le cas de la réciproque de la propriété d'adéquation – le théorème de complétude – que nous verrons ultérieurement.

En considérant le cas particulier de la proposition d'adéquation pour une théorie vide on obtient :

Corollaire 9.6 *Soit F est une formule close ; Si $\vdash F$ alors F est universellement valide.*

On dira qu'une théorie est T *contradictoire* quand on peut dériver $\vdash_T \perp$. Une conséquence immédiate de la proposition d'adéquation est que :

Corollaire 9.7 *Si une théorie T est contradictoire, c.a.d. $\vdash_T \perp$, alors elle est incohérente, c.a.d. qu'elle n'a pas de modèle.*

9.3 Énoncés du théorème de complétude, compacité.

On a vu que la déduction sémantique contenait la déduction syntaxique (proposition d'adéquation 9.5 page 62). Il nous reste à démontrer la réciproque, que l'on appelle *théorème de complétude.* Donnons en un énoncé précis :

Théorème 9.8 (complétude) *Pour toute théorie T, toute formule close C dans le langage L,*

$$\textit{Si } \mid\!\sim_T C \quad \textit{alors} \quad \vdash_T C\ .$$

On rappelle que $\mid\!\sim_T C$ signifie que pour toute L-structure $\mathcal{M}$, si $\mathcal{M} \models T$, alors $\mathcal{M} \models C$. On énonce souvent ce résultat dans le cas particulier où C est l'absurde $\perp$. On rappelle qu'une théorie T telle que $T \vdash \perp$ est dite *contradictoire*, et qu'une théorie T qui possède un modèle est dite *satisfaisable.* Par contraposée le résultat de complétude donne :

Corollaire 9.9 (complétude) *Toute théorie non contradictoire est satisfaisable, c'est à dire que si T est une théorie dans le langage L :*

$$\textit{Si } \nvdash_T \perp \ \textit{ alors il existe une } L\textit{-structure } \mathcal{M} \textit{ telle que } \mathcal{M} \models T\ .$$

De ce cas particulier on dérive facilement le cas général. En effet :

Lemme 9.10 *Pour toute théorie T, toute formule close C dans le langage L,*

$$\vdash_T C \ \textit{ssi} \ \vdash_{T\cup\{\neg C\}} \perp$$

Démonstration. Supposons $\vdash_T C$, alors $\vdash_{T\cup\{\neg C\}} C$ (proposition 9.1 page 62) or $\vdash_{T\cup\{\neg C\}} \neg C$ (axiome) donc $\vdash_{T\cup\{\neg C\}} \perp$ (élimination).
Réciproquement, supposons $\vdash_{T\cup\{\neg C\}} \perp$, alors $\neg C \vdash_T \perp$ (proposition 9.2 page 62) et donc $\vdash_T C$ (raisonnement par l'absurde). ■

On peut maintenant déduire le théorème de complétude 9.8 du corollaire 9.9. Procédons par contraposée. Supposons que $\nvdash_T C$. Alors d'après le lemme 9.10, $\nvdash_{T\cup\{\neg C\}} \perp$. D'après le corollaire 9.9, il existe un modèle $\mathcal{M}$ de $T \cup \{\neg C\}$, c'est à dire un modèle de T qui n'est pas modèle de C, et donc $\not\mid\!\sim_T C$. ■

Une conséquence importante du théorème de complétude est le *théorème de compacité*, dont nous verrons des applications plus tard.

Théorème 9.11 (compacité) *Une théorie T est satisfaisable ssi toutes ses parties finies sont satisfaisables.*

Démonstration. Si T est une théorie satisfaisable, il est évident que toutes ses parties le sont par le même modèle, en particulier celles qui sont finies. Réciproquement supposons que T n'est pas satisfaisable. Alors d'après le théorème de complétude (c'est la contraposée du corollaire 9.9) $\vdash_T \bot$. De la proposition de finitude 9.3 page 62 on déduit qu'il existe une partie finie T_0 de T telle que $\vdash_{T_0} \bot$. D'après la proposition d'adéquation 9.5 page 62, T_0 n'est pas satisfaisable. ■

9.4 Principe de la démonstration.

Le principe de la démonstration est le même que dans le cas du calcul propositionnel, à la différence près qu'il faut maintenant construire un modèle et non simplemen t une valuation. On se restreindra aux langages de signature *dénombrable* : une restriction tout à fait raisonnable, en mathématiques usuelles où les langages sont le plus souvent de signature finie[18]. On montre alors que l'ensemble des formules du premier ordre sur un tel langage est dénombrable. en effet l'ensemble des formules peut être vu comme un sous-ensemble de l'ensemble des mots sur un alphabet contenant la signature du langage, les parenthèses, les symboles pour les connecteurs et les quantificateurs, les noms de variables et l'égalité. Les mots sur un alphabet sont des suites finies d'éléments de cet alphabet. Comme en calcul propositionnel on utilise la proposition suivante.

proposition. *L'ensemble des suites finies d'éléments d'un ensemble dénombrable est dénombrable.*

On va démontrer directement le corollaire 9.9, dont on a vu qu'il était équivalent au théorème de complétude 9.8. La démarche est la même qu'en calcul propositionnel :

1. étant donnée une théorie non contradictoire T l'*étendre*, en utilisant la déduction, en une théorie T^s :
$$T \subset T^s$$
qui doit être *complète*[19], c'est à dire telle que
$$\text{pour toute formule close } C\,, C \in T^s \text{ ou } \neg C \in T^s\ .$$
qui doit être *non contradictoire* c'est à dire telle que :
$$\nvdash_{T^s} \bot\ .$$
et qui doit être *saturée*, c'est à dire telle que :
$$\text{pour toute formule close } C\ (\vdash_{T^s} C \Rightarrow C \in T^s)\ .$$
2. Construire un modèle de la théorie T à partir de la théorie T^s.

On sait que les connecteurs et quantificateurs s'expriment avec $\{\to, \bot, \forall\}$, on montrera le résultat de complétude en restreignant les formules à l'usage de ces seuls connecteurs et quantificateurs. On utilise quand même la négation, comme abréviation $\neg A \equiv_{def} A \to \bot$. Remarquez que les règles de déduction de la négation sont les règles de l'implication dans ce cas particulier (voir figure 6 page 19).

9.5 Calcul des prédicats sans égalité.

Le calcul des prédicats non égalitaire est obtenu en supprimant l'égalité du langage, et les règles de l'égalité de la déduction. On va tout d'abord montrer la complétude dans ce cadre, qui est un peu plus simple que le calcul des prédicats égalitaire.

Il s'agit d'étendre la méthode vue pour le calcul propositionnel au calcul des prédicats. Il y a une difficulté supplémentaire : il faut construire non plus une simple valuation, mais une structure qui

[18]Les langages de signature infinie non dénombrable sont utiles en théorie des modèles, une branche de la logique mathématique.

[19]en un sens plus fort que la notion de théorie complète introduite au paragraphe 8.4, et que nous adoptons uniquement pour la preuve du théorème de complétude. Pour une théorie saturée les deux notions de théorie complète sont équivalentes.

soit modèle de la théorie non contradictoire donnée. En particulier il faut construire l'ensemble de base de cette structure. Pour cela on va utiliser les termes du langage eux-mêmes. On a évidemment besoin de faire correspondre un élément du modèle à chaque terme qui intervient dans la preuve, et que le modèle soit clos pour l'interprétation des symboles de fonction du langage. Tous les éléments d'un modèle ne sont pas forcément désignés par un terme clos (on peut d'ailleurs utiliser un langage sans aucun termes clos !), et dans une preuve on se sert naturellement de variables, donc de termes non clos. Il faut donc utiliser des termes avec variables. Deux des règles de la déduction demandent vraiment d'introduire de nouvelles variables : l'introduction du $\forall$, et l'élimination du $\exists$.

Bien que la démonstration ne soit pas faite pour le $\exists$, voyons d'abord cette règle. Dans une preuve mathématique courante elle correspond à la locution "on sait que $\exists x\ A$, posons x_0 tel que A". On désigne ainsi dans le modèle auquel on fait implicitement référence un élément interprétant le x en question – appelons-le c – que l'on appelle *témoin* de la vérité de $\exists x\ A$, c'est à dire que si $\exists x\ A$ est vraie dans le modèle, c désigne un élément tel que $A[c/x]$ est vraie. Remarquons que la règle impose que la variable x soit *nouvelle* : qu'elle n'apparaisse pas libre dans la relation de déduction en cours (ce que l'on a démontré, les hypothèses courantes, et ce que l'on cherche à démontrer), ce qui signifie que x désigne bien un "nouvel élément" du modèle, un élément auquel il n'a pas encore été fait référence.

Pour l'introduction du $\forall$, on est dans une situation analogue, même si l'intuition est un peu moins immédiate. En effet il s'agit d'une règle d'introduction – elle correspond à la locution "pour montrer $\forall x\ A$, montrons A pour x *quelconque*" – alors que la règle de $\exists$ traitée ci-dessus est une règle d'élimination. C'est cette fois le mot "quelconque" qui fait apparaître que x est "nouvelle". Comme $\forall x\ A$ est cette fois ci une formule que l'on cherche à démontrer (introduction), on va ajouter au modèle que l'on cherche à construire un élément interprétant x – appelons le c – qui sera un témoin de la fausseté de $\forall x\ A$, c'est à dire que si $\forall x\ A$ est fausse dans le modèle, c désigne un élément tel que $A[c/x]$ est fausse.

Avant d'entrer dans le détail de la construction du modèle, voyons un premier lemme syntaxique qui justifie ce genre de manipulation :

Lemme 9.12 *Soit T une théorie du langage L, et $\Gamma \vdash_T F$ une relation de déduction dans ce même langage. Soit c une constante qui n'apparaît pas dans L. Alors, pour toute variable x :*

$$\Gamma \vdash_T F \quad \textit{ssi} \quad \Gamma[c/x] \vdash_T F[c/x]$$

Démonstration. Supposons $\Gamma \vdash_T F$ dérivable, et montrons $\Gamma[c/x] \vdash_T F[c/x]$ par induction sur la dérivation de la relation de déduction. C'est évident pour les règles qui ne mettent pas de variable en jeu. Pour les règles qui mettent en jeu une variable : introduction du $\forall$ élimination du $\exists$, si cette variable n'est pas x c'est évident. Si c'est x, la condition d'utilisation de ces règles impose que x n'apparaisse pas dans la relation déduite. La substitution $[c/x]$ est donc l'identité d'où le résultat.

Supposons $\Gamma[c/x] \vdash_T F[c/x]$ dérivable, comme c n'apparait pas dans L, aucun axiome de T ne met c en jeu, et donc on peut substituer c dans la dérivation par n'importe quel terme, en particulier par n'importe quelle variable x, en renommant éventuellement les variables qui apparaissent en prémisse des règles d'introduction du $\forall$ et d'élimination du $\exists$.

Remarquons que la dérivation de $\Gamma[c/x] \vdash_T F[c/x]$ peut tout aussi bien passer pour une dérivation dans le langage L où on ajouté le nom de variable c. ■

9.5.1 Une théorie complète et saturée T^s.

On se donne une énumération des formules du langage L (y compris celles qui ne sont pas closes) soit $(F_i)_{i \in \mathbb{N}}$.

On suppose T non contradictoire : $\nvdash_T \bot$. On définit par récurrence sur n un multi-ensemble Γ_n défini de la façon suivante.

i. $\Gamma_0 = \emptyset$;

ii. Si $\Gamma_n \vdash_T F_n$ alors $\Gamma_{n+1} = \Gamma_n, F_n$.
Si $\Gamma_n \not\vdash_T F_n$ et si F_n n'est pas quantifiée universellement alors $\Gamma_{n+1} = \Gamma_n, \neg F_n$.
Si $\Gamma_n \not\vdash_T F_n$ et si F_n est de la forme $\forall x\, G$, alors $\Gamma_{n+1} = \Gamma_n, \neg G[x_k/x]$, où x_k est la variable d'indice le plus petit possible qui n'apparait pas dans Γ_n (c'est toujours possible car il y a une infinité de variables).

Quand on prend la réunion ensembliste des Γ_n on obtient un ensemble de formules qui contient nécessairement T, et dont on pourrait montrer qu'il est en un certain sens “complet” et saturé, mais il ne s'agit pas d'une théorie – une théorie est par définition un ensemble de formules closes, ce qui est bien naturel – et cela poserait des problèmes pour définir le modèle.

On va donc on va étendre le langage avec une infinité dénombrables de constantes $H = \{c_i \ / \ i \in \mathbb{N}\}$ *dites constantes de Henkin* ou encore *témoins de Henkin*, qui correspondent aux variables. Si F est une formule du langage L, on lui associe la formule close F^c du langage $L \cup H$ obtenue en remplaçant pour chaque entier i chaque occurrence de variable x_i par la constante c_i. De même si F est une formule close du langage $L \cup H$, on lui associe la formule F^x du langage L obtenue en remplaçant pour chaque entier i chaque occurrence de constante c_i par la variable x_i. Comme $(F_i)_{i\in\mathbb{N}}$ est une énumération des formules du langage L, $(F^c)_{i\in\mathbb{N}}$ est une énumération des formules closes du langage $L \cup H$. On étend la notation Γ^c aux multi-ensembles Γ.

On peut maintenant définir les théories suivantes, du langage $L \cup H$:

$$T_n = T \cup \{F^c \ / \ F \text{ apparaît dans } \Gamma_n\} \qquad T^s = \bigcup_{i\in\mathbb{N}} T_n$$

On a alors :

Lemme 9.13 *Pour F une formule du langage L, pour n n'importe quel entier :*

$$\vdash_{T_n} F^c \ \textit{ssi}\ \Gamma_n \vdash_T F$$

Démonstration. On commence par montrer par récurrence sur n que :

$$\vdash_{T_n} F^c \text{ ssi } \Gamma_n^c \vdash_T F^c$$

en utilisant la proposition 9.2 page 62.

On démontre ensuite le résultat par récurrence sur le nombre de variables apparaissant dans Γ_n et F en utilisant le lemme 9.12. ■

On montre maintenant que certaines des constantes de Henkin sont des “témoins” pour les formules universelles.

Lemme 9.14 *Soit $\forall xG$ une formule universelle du langage L. Si $(\forall xG)^c \notin T^s$, il existe une constante de Henkin $c \in H$ telle que $\neg G[c/x]^c \in T^s$.*

Démonstration. Soit n l'indice de cette formule : $\forall xG = F_n$. Comme $F_n^c \notin T^s$, cela signifie que $\Gamma_n \not\vdash_T \forall xG$, donc qu'il existe une variable x_k c telle que $\neg G[x_k/x] \in \Gamma_{n+1}$ (définition de Γ_{n+1}), donc $\neg G[x_k/x]^c \in T_{n+1}$, or $\neg G[x_k/x]^c = \neg G[c_k/x]^c$. ■

Lemme 9.15 *La théorie T^s (dans le langage $L \cup H$) est complète et contient T.*

Démonstration. La théorie T^s contient T car pour toute formule F de T, $\vdash_T F$, donc F apparaît dans un Γ_n, et comme F est close $F^c = F$.

Montrons qu'elle est complète. Toutes les formules closes du langage $L \cup H$ s'écrivent F_n^c. Pour tout entier n tel que F_n n'est pas universelle, par construction, $F_n^c \in T^s$ ou $\neg F_n^c \in T^s$.

Pour tout entier n telle que F_n est universelle, soit $F_n = \forall xG$, on a $F_n^c \in T^s$ ou $\neg G[c_k/x]^c \in T^s$ (le fait que x_k n'apparait pas dans Γ_n ne sert pas pour ce lemme).

Supposons que $\neg G[c_k/x]^c \in T^s$, c'est à dire $\Gamma_n \not\vdash_T F_n$. On a par construction de Γ_{n+1}, $\Gamma_{n+1} \vdash_T \neg G[x_k/x]$ (axiome). Par ailleurs $\forall xG \vdash_T G[x_k/x]$ (élimination). De ces deux énoncés on déduit $\Gamma_{n+1}, \forall xG \vdash_T \bot$ (élimination), donc $\Gamma_{n+1} \vdash_T \neg\forall xG$ (introduction). Appelons p l'indice de $\neg F_n = \neg\forall xG = F_p$.

Si $p > n$, Γ_p contient Γ_{n+1}, donc $\Gamma_p \vdash_T F_p$ et donc $\neg F_n^c \in T^s$.

Si $p < n$, Montrons que l'on a également $\Gamma_p \vdash_T F_p$. Sinon, comme $n \geqslant p+1$, $\Gamma_n \vdash \neg F_p$, c'est à dire $\Gamma_n \vdash_T \neg\neg F_n$. On en déduit (axiome, puis élimination), $\Gamma_n, \neg F_n \vdash_T \perp$ et donc $\Gamma_n \vdash F_n$ (raisonnement par l'absurde). Ceci contredit l'hypothèse $\neg G[c_k/x]^c \in T^s$. On a donc bien $\Gamma_p \vdash_T F_p$ et donc à nouveau $\neg F_n^c \in T^s$.

On a donc bien montré que pour tout entier n $F_n^c \in T^s$ ou $\neg F_n^c \in T^s$. ■

Lemme 9.16 *La théorie T^s n'est pas contradictoire : $\not\vdash_{T^s} \perp$.*

Démonstration. En vertu de la finitude 9.3, si T^s était contradictoire, l'une de ses parties finies le serait. Toute partie finie de T^s est inclue dans T_n pour un entier n suffisamment grand. Il suffit donc de montrer que pour tout entier n, T_n n'est pas contradictoire, ce que l'on fait par récurrence sur n.

$n = 0$. $T_0 = T$ n'est pas contradictoire par hypothèse.

$n \to n+1$. On procède par contraposée. Supposons que T_{n+1} est contradictoire, c'est à dire $\vdash_{T_{n+1}} \perp$, et montrons qu'alors T_n l'est. De $\vdash_{T_{n+1}} \perp$ on déduit $\Gamma_{n+1} \vdash_T \perp$ (lemme 9.13). Par construction de Γ_{n+1}, trois cas se présentent :

$\Gamma_{n+1} = \Gamma_n, F_n$. Cela signifie que $\Gamma_n \vdash_T F_n$. Or T_{n+1} contradictoire signifie que $\vdash_{T_n \cup \{F_n^c\}} \perp$, donc $F_n^c \vdash_{T_n} \perp$ (proposition 9.2), donc $\Gamma_n, F_n \vdash_T \perp$ (lemme 9.13) donc $\Gamma_n \vdash_T \neg F_n$ (introduction). Or $\Gamma_n \vdash_T F_n$, donc $\Gamma_n \vdash_T \perp$ (élimination), c'est à dire T_n contradictoire (lemme 9.13).

$\Gamma_{n+1} = \Gamma_n, \neg F_n$. Cela signifie que $\Gamma_n \not\vdash_T F_n$. Or T_{n+1} contradictoire signifie que $\vdash_{T_n \cup \{\neg F_n^c\}} \perp$, donc $\Gamma_n, \neg F_n \vdash_T \perp$ (lemme 9.13), donc $\Gamma_n \vdash_T F_n$ (raisonnement par l'absurde), ce qui contredit l'hypothèse. Ce cas ne peut se produire.

$\Gamma_{n+1} = \Gamma_n, \neg G[x_k/x]$, avec $F_n = \forall x\, G$ et x_k n'apparait pas dans Γ_n. Cela signifie que $\Gamma_n \not\vdash_T F_n$. Or T_{n+1} contradictoire signifie que $\vdash_{T_n \cup \{\neg G[c_k/x]^c\}} \perp$, donc d'après le lemme 9.13 $\Gamma_n, \neg G[x_k/x] \vdash_T \perp$, donc $\Gamma_n, \vdash_T G[x_k/x]$ (raisonnement par l'absurde), donc $\Gamma_n \vdash_T \forall x G$ (introduction du $\forall$, on a bien x_k qui n'apparait pas dans Γ_n par construction), c'est à dire $\Gamma_n \vdash_T F_n$. ■

Lemme 9.17 *La théorie T^s est saturée.*

Démonstration. Supposons $\vdash_{T^s} C$. Comme T^s est complète (lemme 9.15) $C \in T^s$ ou $\neg C \in T^s$. Si $\neg C \in T^s$, $\vdash_{T^s} \neg C$, et donc $\vdash_{T^s} \perp$ (élimination), or T^s n'est pas contradictoire (lemme 9.16). On a donc bien $C \in T^s$. ■

9.5.2 Construction d'un modèle de T.

L'ensemble de base M de la structure $\mathcal{M}$ est l'ensemble des termes clos du langage $L \cup H$ (le langage de départ étendu avec les constantes de Henkin). Les symboles de constante du langage $L \cup H$ sont interprétées par eux-mêmes. Les éventuels symboles de fonction du langage L ont leur interprétation canonique (f n-aire est interprétée par la fonction qui à $t_1, \ldots, t_n$ associe $ft_1, \ldots, t_n$). Les formules atomiques closes α sont vraies si et seulement si elles sont dans T^s :

$$\mathcal{M} \models \alpha \text{ ssi } \alpha \in T^s.$$

On montre que ceci s'étend à toutes les formules closes de $L \cup H$.

Lemme 9.18 *Pour toute formule F du langage L :*

$$\mathcal{M} \models F^c \textit{ ssi } F^c \in T^s.$$

Démonstration. La preuve se fait pas une induction sur la complexité des formules un peu renforcée : pour une formule universelle $\forall x\ A$, on a besoin de l'hypothèse d'induction sur toutes les formules $A[t/x]$, t étant un terme quelconque. Ce principe d'induction est une simple conséquence du principe de récurrence sur la complexité des formules (hauteur de son arbre de dérivation). Dans le cas des formules atomiques et de l'implication, la preuve est essentiellement la même qu'en calcul propositionnel.

Atomes. C'est la définition du modèle.

Absurde. On a $\mathcal{M} \not\models \perp$ par définition de la satisfaction et $\perp \notin T^s$ d'après le lemme 9.16.

Implication. Supposons le résultat pour A et B, et montrons le pour $A \to B$. Distinguons deux cas.

Supposons $(A \to B)^c \in T^s$. Si $A^c \notin T^s$, on a $\mathcal{M} \not\models A^c$ par hypothèse d'induction et donc $\mathcal{M} \models (A \to B)^c$. Si $A^c \in T^s$, on a $\vdash_{T^s} A^c \to B^c$ et $\vdash_{T^s} A^c$. On en déduit (élimination) $\vdash_{T^s} B^c$. Par saturation $B^c \in T^s$, et donc $\mathcal{M} \models B^c$, donc $\mathcal{M} \models (A \to B)^c$.

Supposons $(A \to B)^c \notin T^s$. Les deux cas $A^c \notin T^s$ et $B \in T^s$ sont exclus car ils contredisent l'hypothèse $(A \to B) \notin T^s$, comme le montre ce qui suit.

Supposons $A^c \notin T^s$. Comme la théorie est complète on a que $\neg A^c \in T^s$. On en déduit (axiome de la théorie et élimination) $A^c \vdash_{T^s} \perp$. On a donc $A^c \vdash_{T^s} B^c$ (élimination de $\perp$). On en déduit $\vdash_{T^s} A^c \to B^c$ (introduction), et donc par saturation (lemme 5.12) $(A \to B)^c \in T^s$.

Supposons $B^c \in T^s$. On a donc $\vdash_{T^s} B^c$ et donc $A^c \vdash_{T^s} B^c$ (affaiblissement). On en déduit $\vdash_{T^s} A^c \to B^c$ (introduction), et donc par saturation $(A \to B)^c \in T^s$.

Comme $A^c \in T^s$ et $B^c \notin T^s$, on a par hypothèse d'induction $\mathcal{M} \models A$ et $\mathcal{M} \not\models B$ et donc $\mathcal{M} \not\models A \to B$.

Quantification universelle. Supposons le résultat pour toutes les formules $A[t/x]$ et montrons le pour $\forall x A$. Distinguons deux cas.

Supposons $(\forall x A)^c \in T^s$. Par élimination $\vdash_{T^s} A[t/x]^c$ pour tout terme t, donc par saturation, $A[t/x]^c \in T^s$ pour tout terme t, or chaque élément du modèle $\mathcal{M}$ est l'interprétation d'un terme clos de $L \cup H$, donc $\mathcal{M} \models \forall x A^c$.

Supposons $(\forall x A)^c \notin T^s$. On sait (lemme 9.14) qu'il existe un témoin de Henkin c telle que $\neg A[c/x]^c \in T^s$. Par hypothèse d'induction $\mathcal{M} \not\models A[c/x]^c$, par définition de la satisfaction $\mathcal{M} \not\models \forall x A^c$. ∎

Corollaire 9.19 *Il existe un modèle $\mathcal{M}$ tel que pour toute formule F de T $\mathcal{M} \models F$.*

Démonstration. Comme $T^s \supset T$, c'est une conséquence immédiate du lemme précédent : il suffit d'oublier dans la $L \cup H$-structure $\mathcal{M}$ les interprétations des constantes de Henkin pour obtenir une L-structure satisfaisant toutes les formules de T^s où les constantes de Henkin n'apparaissent pas, en particulier les formules de T. ∎

On a donc bien démontré le théorème de complétude sous la forme énoncée au corollaire 9.9.

9.6 Calcul des prédicats avec égalité.

Remarquons tout d'abord que l'on ne déduit pas directement le théorème de complétude du calcul des prédicats égalitaire de la complétude du calcul des prédicat sans égalité, en la particularisant à la théorie de l'égalité (et aux théories contenant celle-ci). En effet, même s'il est tout à fait possible de présenter de façon axiomatique la théorie de l'égalité, la sémantique elle est plus précise : dans un modèle égalitaire on doit interpréter l'égalité par l'identité, et non par une simple relation satisfaisant les axiomes de l'égalité. Le théorème de complétude en calcul des prédicats égalitaire pour une théorie T est donc en un sens "plus fort" que le théorème de complétude en calcul des prédicats pour la théorie T plus la théorie de l'égalité (puisque l'on a alors a priori d'autre modèles de la théorie de l'égalité que ceux où celle-ci est interprétée par l'identité).

Cependant, la preuve est essentiellement la même que pour le calcul des prédicats sans égalité. On va se contenter de montrer comment la modifier : il s'agit de construire un contre-modèle pour une théorie égalitaire non contradictoire en interprétant l'égalité par l'identité.

Tout d'abord, on étend de la même façon le langage par des constantes de Henkin, et l'on construit de la même façon une théorie T^s contenant T (voir paragraphe 9.5.1), avec pour seule différence que l'on utilise les règles de l'égalité pour la déduction (dans la définition des Γ_n).

On montre de la même façon que si T est cohérente, T^s est cohérente, non contradictoire et saturée (cette fois, y compris pour les règles de l'égalité). Il n'y a rien à modifier aux preuves des lemmes du paragraphe 9.5.1.

Seule la construction de l'ensemble de base du modèle $\mathcal{M}$ diffère (voir paragraphe 9.5.2). On considère toujours l'ensemble des termes clos du langage $L \cup H$, mais on le quotiente par la relation définie par les égalités syntaxiques entre les termes dans T^s :

$$t \sim t' \text{ ssi } t = t' \in T^s.$$

- Cette relation est une relation d'équivalence : elle est réflexive par définition, symétrique et transitive d'après le lemme 2.1 page 19.
- Elle est compatible avec les éventuels symboles de fonction du langage. Pour f un symbole de fonction n-aire :

$$\text{si } t_1 \sim t'_1, \ldots, t_n \sim t'_n \text{ alors } ft_1 \ldots t_n \sim ft'_1 \ldots t'_n \,.$$

  En effet il suffit d'appliquer n fois la règle de l'égalité à l'égalité $ft_1 \ldots t_n = ft_1 \ldots t_n$ obtenue par réflexivité.
- Cette relation est compatible les éventuels symboles de prédicat du langage. Pour P un symbole de prédicat n-aire

$$\text{si } t_1 \sim t'_1, \ldots, t_n \sim t'_n \text{ alors } (Pt_1 \ldots t_n \in T^s \text{ ssi } Pt'_1 \ldots t'_n \in T^s)$$

  En effet il suffit de montrer $Pt_1 \ldots t_n \in T^s \Rightarrow Pt'_1 \ldots t'_n \in T^s$ qui se démontre en appliquant n fois la règle de l'égalité au prédicat $Pt_1 \ldots t_n$.

On peut alors définir le modèle $\mathcal{M}$ comme précédemment. On démontre alors de la même façon le lemme 9.18.

9.7 Théorème de Lowenheim-Skolem.

On peut remarquer que la construction précédente fournit pour une théorie satisfaisable d'un langage de signature finie ou dénombrable, un modèle dont l'ensemble de base est fini ou dénombrable, puisqu'il est obtenu comme quotient sur un ensemble de termes clos d'un langage dénombrable (le langage d'origine, complété par une infinité dénombrable de constantes de Henkin). On en déduit une forme faible du théorème de Lowenheim-Skolem :

Théorème 9.20 *Toute théorie T satisfaisable d'un langage de signature finie ou dénombrable possède un modèle d'ensemble de base dénombrable.*

Démonstration. Reprendre la construction du modèle effectuée pour une théorie satisfaisable, dans la preuve du théorème de complétude. ∎

Cet énoncé est parfois connu sous le nom de *paradoxe de Skolem*, bien que ce ne soit pas réellement un paradoxe. En effet, il peut être surprenant de penser que la théorie des ensembles, dans laquelle on peut construire l'ensemble des réels $\mathbb{R}$, a un modèle dénombrable. Le paradoxe disparait si l'on analyse les choses de près.

Tout d'abord la théorie des ensembles considérée est une théorie du premier ordre de l'appartenance. Dans un modèle de la théorie des ensembles, un ensemble est donc représenté par un point du modèle. Les axiomes sur l'appartenance assurent que ce point représente un ensemble au sens intuitif (collection de points). Mais il n'est pas possible pour des questions de cardinalité que chaque collection de points soit représentée par un point du modèle.

La notion de dénombrable dans un modèle dénombrable (au sens intuitif) de la théorie des ensembles n'est pas celle du meta-langage. C'est à dire que dans le modèle étudié on ne peut trouver de bijection (un point du modèle) entre les ensembles $\mathbb{N}$ et $\mathbb{R}$.

Index

Table des matières