Interprétation fonctionnelle et élimination des coupures de l'arithmétique d'ordre supérieur

Jean-Yves Girard

Introduction

I. Points de vue extensionnel et calculatoire

1

Si nous considérons les deux fonctions suivantes, définies sur $\mathbb{N}^4$:

$$F_1 : x, y, z, n \mapsto \begin{cases} 0 & \text{si } xyzn = 0, \text{ ou } n = 1, \text{ ou } n = 2, \text{ ou } x^n + y^n \neq z^n \\ 1 & \text{sinon} \end{cases}$$

$$F_2 : x, y, z, n \mapsto 0$$

du point de vue des *graphes* associés, ces deux fonctions seront égales ou distinctes, suivant que le grand théorème de Fermat est vrai ou faux.

Par contre, les idées qui servent à définir ces deux fonctions (ou mieux : leurs *algorithmes* respectifs de calcul) sont très différentes, et ceci indépendemment de la vérité du théorème de Fermat.

En résumé, il faut distinguer entre deux points de vue dans l'étude des objets mathématiques, et spécialement des fonctions :

– le point de vue *extensionnel*, pour lequel les fonctions sont des graphes, *l'égalité extensionnelle* entre fonctions étant par définition l'identité des graphes. Ce point de vue se traduit, en théorie des ensembles, par l'axiome d'extensionalité.

– le point de vue intentionnel (ou mieux, *algorithmique*, *calculatoire*) pour lequel les fonctions seront d'avantage des algorithmes, des procédés de calcul. L'égalité intentionnelle (ou algorithmique, calculatoire) est alors définie comme l'égalité des algorithmes, modulo un certain nombre de transformations simples.

> En fait la situation est plus complexe qu'il ne semble, car si l'égalité extensionnel a une signification univoque, les propriétés de l'égalité algorithmique dépendent essentiellement du choix des "transformations simples" dont nous avons parlé. Par exemple, les deux fonctions calculatoires définies par les algorithmes
>
> $$F : x, y \mapsto x + y \quad \text{et} \quad G : x, y \mapsto y + x$$
>
> seront égales ou non, suivant que la commutation de la somme sera ou ne sera pas parmi nos "transformations simples". Il faut de plus remarquer que, si en général les tranformations en question se ramènent à des suites de transformations récursives primitives, ceci ne signifie pas que l'égalité algorithmique soit décidable, et à fortiori, récursive primitive. (Par exemple, dans les systèmes fonctionnels que nous étudions, l'identité des formes normales est une égalité algorithmique décidable, mais qui n'est pas récursive primitive.)

Le point de vue algorithmique permet la plupart du temps de définir une égalité décidable entre fonctions.

2

Dans les systèmes que nous utilisons par la suite, des fonctions aussi élémentairement semblables que F et G ne seront pas identifiées : F et G seront représentées par deux assemblages formels $\lambda x\ \lambda y\ x + y$ et $\lambda x\ \lambda y\ y + x$ que nous considèrerons comme irréductibles l'un à l'autre.

II. Systèmes fonctionnels (Étude syntaxique)

Les recherches en théorie de la démonstration ont mis en avant la notion de *système fonctionnel*.

Commençons par une mise au point :

– d'abord, les termes "fonctionnel", "fonctionnelle" évoquent des opérations extensionnelles (définies sur des graphes), alors que justement les systèmes et objets en question sont l'illustration-même du point de vue algorithmique.

Il serait bien plus correct de parler de "symboles fonctionnels", "système de symboles fonctionnels", et de réserver la terminologie "fonctionnelle", "système fonctionnel" pour l'interprétation de ces notions (voir III). Dans la suite de cet exposé cependant, nous emploierons systématiquement l'abus de langage qui consiste à appeler "fonctionnelles" les assemblages qui les dénotent.

– ensuite, il faut bien préciser que nous ne disposons pas à l'heure actuelle d'une définition génerale de la notion vague de "système fonctionnel". Ceci dit, nous décrivons dans la première partie de l'exposé un certain nombre de structures mathématiques qu'on a l'habitude d'appeler "systèmes fonctionnels". Nous allons décrire succinctement quelques uns de leurs points communs marquants. Pour la raison que nous avons donnée, cette description ne doit pas être prise pour une définition dogmatique.

1) Les objets des systèmes fonctionnels (*termes*, *fonctionnelles*, ou mieux, *symboles fonctionnels*) sont obtenus au moyen de constructions combinatoires.

Ainsi, étant donnés des termes a et b et une variable x, nous pouvons former (dans certaines conditions) les termes $\lambda x\ a$, $a(b)$. Le caractère combinatoire des termes (et des types) est lié à des questions de décidabilité. Dans le cours de l'exposé, nous envisageons même des termes qui ne sont pas obtenus par des méthodes combinatoires ($[\![$utilisation$]\!]$ de suites infinies), mais il ne s'agit là que de constructions temporaires, liées à certaines exigences démonstratives.

2) Les objets des systèmes sont classifiés à l'aide de *types*. Chaque fonctionnelle 3
possède un type unique. Les types sont eux aussi des objets combinatoires. Les types servent aussi à règlementer les constructions de fonctionnelles : les fonctionnelles $a_1, \ldots, a_n$, ne pourrent être utilisées dans une construction que si leurs types respectifs satisfont certaines conditions syntaxiques, dépendant de la construction envisagée.

Par exemple, si a est de type τ, x de type σ, $\lambda x\ a$ est de type $\sigma \to \tau$. Nous ne pourrons former $a(b)$ que si le type de a s'écrit $\sigma \to \tau$, le type de b étant alors σ.

Enfin, dans certains systèmes comme OF_2, les types peuvent aussi agir réellement sur les fonctionnelles. (Contrairement à ce qui se passe dans le système de Gödel OF_1.)

Par exemple, si a est de type $\wedge\alpha\ \sigma[\alpha]$, et τ est un type, $a\{\tau\}$ est un fonctionnelle de type $\sigma[\tau]$.

3) Pour chaque type σ, on définit un préordre $\preceq$ entre éléments de type σ. $\preceq$ s'appelle ***réduction***, et correspond à l'idée de calcul.

Par exemple, on a : $\lambda x\ a\ (b) \preceq a[b]$ et $\mathsf{DT}\alpha\ a\ \{\tau\} \preceq a[\tau]$, où $a[C]$ désigne la substitution combinatoire de l'objet (fonctionnelle ou type) C pour la variable (x ou α).

D'un point de vue élémentaire, la description de $\preceq$ en tant que préordre peut être avantageusement précisée comme suit : on dispose d'une relation binaire $\prec_s$, de réduction *stricte* (à ne pas confondre avec $\prec_1$ de la première partie ; $\prec_s$ est introduit en début de seconde partie). $a \prec_s b$ signifie que b est obtenu à partir de a en utilisant *exactement* une règle de calcul. $\preceq$ est alors la relation réflexive et transitive associée à $\prec_s$. $\prec_s$ doit vérifier la propriété : pour tout a, l'ensemble des b tels que $a \prec_s b$ est fini, c'est à dire qu'étant donné a, il n'y a qu'un nombre fini de premières étapes de calcul de a.

On définit d'autre part un certain nombre d'éléments *maximaux* pour $\preceq$, que l'on appelle éléments *normaux*. Un élément normal est donc un des points d'aboutissement désirables pour un calcul.

En fait, les éléments normaux vérifient la condition suivante (plus forte que la maximalité) : si a est normal, alors il n'existe pas de b tel que $a \prec_s b$. Par exemple, dans le lambda-calcul (qui n'est pas un système fonctionnel, dans le sens où nous l'entendons), le terme non normal $(\lambda x\, x(x))\ (\lambda x\, x(x))$ est maximal pour $\preceq$, mais se réduit strictement en lui-même. Un terme sans rédion strict n'apparaît pas nécessairement (d'un point de vue élémentaire) comme normal (c'est le cas dans les systèmes contenant GD et RED) ; c'est par contre vrai si le terme *possède tous ses rédions stricts* (voir partie II).

La relation $\preceq$ est supposée vérifier un certain nombre de propriétés, par exemple 4
que deux calculs effectués suivant des "chemins" différents, à partir d'un même point de départ, donnent le même résultat, ce qui se traduit par la *propriété de Church-Rosser* : si $a \preceq b$ et $a \preceq c$, alors il existe d, $b \preceq d$ et $c \preceq d$.

Par exemple, soit dans le système OF_1 une définition du produit et de la puissance n-ième, avec la règle $a \cdot 0 \preceq 0$. Si nous devons calculer l'expression $103^{207} \cdot 0$, divers chemins de réduction peuvent être utilisés, et ils donnent tous le même resultat. Mais un de ces chemins est nettement plus court que les autres, et nous sommes autorisés par Church-Rosser à considérer que ce chemin.

Toutes les propriétés dont nous venons de parler sont vérifiables au moyen de raisonnements élémentaires. Il n'en va pas de même pour la propriété essentielle que nous allons énoncer maintenant :

> Le préordre inverse de $\preceq$ est bien fondé, et ses éléments minimaux (les éléments maximaux de $\preceq$) sont exactement les termes normaux.

En particulier, $\preceq$ est un ordre.

Traduit en termes de $\prec_s$, ce qui précède s'énonce précisément : pour chaque terme a, toutes les suites de réductions strictes partant de a sont finies, et les éléments terminaux des calculs sont normaux. Ceci s'énonce "*tout terme est absolument normalisable*". Comme, pour un terme donné a, l'ensemble des *rédions stricts* de a (les b tels que $a \prec_s b$) est fini, ceci peut s'énoncer plus simplement par le théorème de Brouwer-König sous

forme arithmétique. Remarquons que, si a est absolument normalisable, alors a ne se réduit pas strictement en lui-même. Pour prouver que tous les termes d'un système donné sont absolument normalisables, il est nécessaire de faire intervenir le concept de ***terme héréditairement absolument normalisable***, ou ***réductible***. L'énoncé qui formalise la réductibilité ne se met pas sous forme élémentaire en général. (Pour OF_1, la réductibilité n'est pas arithmétique, pour OF_2, elle n'est pas analytique). En résumé, le résultat qui nous intéresse (l'absolue normalisabilité de tous les termes) est arithmétique (Π^0_2), mais sa démonstration nécessite un détour par une notion (la reductibilité) qui ne l'est pas.

Dans les systèmes associés aux théories non prédicatives, comme OF_2, la définition *correcte* de la réductibilité est très délicate.

Un terme de type primitif clos, comme le type o des entiers sera réductible ssi il est absolument normalisable (AN). De même, un terme a de type $\sigma \to \tau$ sera réductible ssi pour tout b réductible de type σ, $a(b)$ est réductible de type τ. Un terme a de type $\wedge\alpha\ \sigma$ sera réductible ssi pour tout type τ, et toute définition "arbitraire" de la réductibilité de type τ, $a\{\tau\}$ est réductible de type $\sigma[\tau]$, si on définit cette notion en prenant, pour chaque apparition de τ qui remplace une apparition de α dans σ, cette définition "arbitraire" comme définition primitive de la réductibilité de type σ. Nous allons revenir $\|$ sur la notion de définition "arbitraire". Remarquons tout de suite que 5
la réductibilité pour le terme a $[\![$de type $\wedge\alpha\ \sigma]\!]$ n'est pas définie en fonction du premier symbole de a. (Contrairement à ce que font, par exemple Martin-Löf et Prawitz.) Ce point est important pour la formalisation des résultats ; il necessite par ailleurs de compliquer les démonstrations, en introduisant la notion essentielle de ***simplicité***. D'autre part, l'emploi de définitions arbitraires (dans un certain sens) de la réductibilité permet de résoudre le problème posé par la circularité inhérente aux types universels.

Dans le cas de OF_2, la réductibilité se définit donc par quantification sur un ensemble de définitions "arbitraires" de la réductibilité, les ***candidats de réductibilité***, ou CR. Le problème de la définition correcte de la réductibilité se ramène donc au choix de la définition des CR.

D'abord, en vue de ce que l'on veut démontrer, l'ensemble des termes AN de type σ doit être un CR de type σ, et ce fait doit être une évidence élémentaire. D'autre part, l'ensemble des CR doit être clos par rapport aux constructions effectuées lors de la définition inductive de la réductibilité. On definit donc un CR de type σ comme un ensemble de termes de type σ satisfaisant aux conditions cruciales :

(CR 2) Tout élément de A est AN.

(CR 5) Si $a \in A$, et $a \preceq b$, alors $b \in A$.

Il nous faut enfin une condition (la plus importante) nous permettant de dire qu'un CR possède "suffisamment" d'éléments : pour celà on définit les termes ***simples***, dont la propriété essentielle est la suivante : si a est simple, tous les rédions stricts de $a(b)$ (resp. $a\{\tau\}$) sont de la forme $a'(b')$ (resp. $a'\{\tau\}$) avec $a \preceq a'$, $b \preceq b'$. Dans le cas des systèmes fonctionnels isomorphes à des systèmes de déduction naturelle, les termes simples correspondent aux déductions qui ne se terminent pas par une introduction (si les seules ***coupures*** considérées sont du type introd./élim.). On énonce alors

(CR 4) Si a est AN et simple, et si tous les rédions stricts de a sont dans A, alors $a \in A$. En particulier :

(CR 3) Si a est simple et normal, alors $a \in A$.

Ces définitions sont parfaitement adaptées pour la démonstration du théorème de réductibilité dans la plupart des systèmes fonctionnels. De plus, quelques retouches mineures des définitions suffisent à assurer la *formalisabilité locale* de la démonstration, pour OF_2, par exemple, dans HA_2. Par contre, ces définitions perdent leur caractère opératoire quand se pose le problème des réductions commutatives, ou *ultra-réductions*, dans les systèmes possédant des types disjonctifs et/ou existentiels.

La raison de cette difficulté s'explique ainsi : si a n'est pas simple, on peut toujours trouver, dans les systèmes sans ultra-réductions, un entier n (dépendant du premier symbole de a) tel que tous les *dégénérés* n-ièmes de a (les termes obtenus à partir de a quand on a effectué n fois l'opération $()$ ou $\{\}$) soient simples, et on peut alors appliquer (CR 4) à ces dégénérés.

Dans le cas des réductions commutatives, il n'y a pratiquement plus de termes simples, du moins si on s'en tient à la définition primitive. On est amené ainsi à modifier la définition de la simplicité, de manière à préserver les résultats obtenus précédemment, et à trouver un nouveau principe, analogue à (CR 4) qui puisse nous permettre de raisonner dans le cas où ce critère est devenu inopérant. Le résultat est le critère (CR″ 4), que nous ne décrirons pas ici. La démonstration de l'adéquation de ce critère est fort longue. ((CR″ 4) est une généralisation des principes employés par Prawitz dans sa démonstration de la réductibilité pour les systèmes du premier ordre avec ultra-réduction.) La démonstration que nous donnons se prête encore à une formalisation locale presque immédiate. 6

III. Interprétation des fonctionnelles

D'après ce qui précède, toute fonctionnelle est majorée, pour $\preceq$, par une fonctionnelle normale unique, *sa forme normale*. En ce sens, on peut dire que les règles de réduction donnent une signification aux fonctionnelles par l'intermédiaire de leurs formes normales.

Par exemple, si on a défini les règles de réduction standard pour la somme, $(\lambda x\ \lambda y\ x + y)\ (\overline{0}, \overline{1}) \preceq \overline{0} + \overline{1} \preceq \overline{1}$.

Encore faudrait il :

1) Que les formes normales possèdent elles-même une interprétation mathématique simple.

Par exemple, les fonctionnelles normales et closes de type 0 s'identifient canoniquement aux entiers naturels. Il n'en va pas de même pour les types supérieurs. Comme nous l'avons dit, le choix des règles de calcul (et donc des formes normales) ne correspond pas forcément à une nécessité mathématique bien comprise. On s'aperçoit ainsi qu'il est presque toujours possible d'adjoindre certaines règles, changeant par là-même la notion de forme normale. (Ceci n'est bien sûr pas possible au type 0, car nous ne voulons pas identifier des entiers distincts.) Par exemple, on aurait pu considérer les

η-réductions, dont un exemple est la règle $\lambda x\ a(x) \preceq a$ (x non libre dans a). Cette règle n'est pas dépourvue totalement d'intérêt : en ajoutant une nouvelle constante de chaque type, sans règle spécifique pour elle, on voit que l'égalité définie entre termes clos comme l'identité des formes normales permet d'obtenir à peu de frais un "modèle" extensionnel du système (et celà même pour les systèmes contenant GD et RED). Malheureusement, ce "modèle", si il vérifie une condition délicate (l'extensionalité), ne jouit pas par contre des propriétés très simples qu'on est en droit de demander par ailleurs. (Les η-reductions ne sont pas étudiées dans cet exposé ; la construction du modèle est complètement évidente.)

2) Que la signification (extra-combinatoire) des types soit évidente.

Pour le type o, il n'y a pas de problème. Ce type peut être identifié en toute tranquillité à $\mathbb{N}$. Si nous avons pu décrire les types σ et τ en tant qu'ensembles d'indices de fonctions récursives, le type $\sigma \to \tau$ sera l'ensemble des indices f de fonction récursives partielles $\|$ appliquant tout élément e de type σ, sur l'élément $\{f\}e$ de type τ. 7

La signification du type universel $\wedge\alpha\ \sigma$ est plus delicate : remarquons en effet que si e est de ce type, $e\{\tau\}$ doit être de type $\sigma[\tau]$, et ce, pour tout τ. C'est à dire que la compréhension d'***un*** type universel présuppose (en première analyse) la compréhension de tous les types, y compris lui-même. Il y a là un cercle, qui peut être brisé en introduisant un nouveau type pour chaque ensemble d'entiers. Ces nouveaux types sont appelés ***types primaires***, et le type que nous voulons définir fait déjà partie des types primaires. Les éléments de $\wedge\alpha\ \sigma$ sont par définition les e qui sont dans tous les $\sigma[P]$, P primaire. Nous venons de définir, à quelque chose près la structure HRO_2 de la seconde partie. (Remarquer la similitude entre types primaires et candidats de réductibilité. Ces constructions sont aussi apparentées à la "***réalisabilité***" de Kreisel-Troelstra ; la réalisabilité n'est pas étudiée dans cet exposé.)

Signalons au passage que seule une petite partie de HRO_2 est l'interprétation du système correspondant OF_2. Cependant, cette interprétation par HRO_2, tout en résolvant un problème, en ouvre un autre : a, de type universel $\wedge\alpha\ \sigma$, sera interprété par un indice e, qui est dans tous les $\sigma[P]$, en particulier, $e\{P\} = e$ pour tout P. Les objets de type universel sont donc des fonctions définies "uniformément" sur les types. Dans le cas précis des fonctions récursives, nous avons une caractérisation de l'uniformité, que nous venons de donner. Par contre le problème est ouvert de trouver une interprétation de la notion vague d'uniformite dans un contexte qui ne soit pas aussi restrictif que celui des fonctions récursives.

3) Que l'on possède (dans certains cas) des *modèles extensionnels standard*.

Un ***modèle*** M du système fonctionnel F est obtenu par l'adjonction de nouvelles constantes (pour les types et les fonctionnelles). Les règles de réduction s'étendent canoniquement à cette structure (en n'introduisant aucune règle spécifique pour ces nouvelles constantes), et on se restreint aux réductions qui ne font intervenir que des termes clos (réductions $*$). On suppose de plus que M est muni, sur chaque type σ, d'une équivalence $\overset{*}{=}$ entre éléments de type σ. La relation $\overset{*}{=}$ doit vérifier les propriétés :

$$a \overset{*}{=} b \Rightarrow T(a) \overset{*}{=} T(b), \qquad a \text{ et } b \text{ de type } \sigma,\ T \text{ de type } \sigma \to \tau$$
$$a \overset{*}{\preceq} b \Rightarrow a \overset{*}{=} b$$

Ces conditions ne sont pas suffisantes dans les applications pratiques. On est amené à définir un type $()$, à deux éléments V et F (voir IV), et on exige que $\overset{*}{=}$ détermine exactement deux classes sur ce type, celles de V et de F. Si le système possède de plus des types disjonctifs et/ou existentiels, on demande que $\overset{*}{=}$ vérifie certaines conditions supplémentaires (voir partie IV).

Un modèle est *standard* si pour tout a de type o, il existe n tel que $a \overset{*}{=} \overline{n}$. (Ce n est unique, à cause de la condition sur $()$.)

Un modèle est *extensionnel* si il vérifie de plus : $a \overset{*}{=} b$ ssi $a(c) \overset{*}{=} b(c)$ pour tout c (a et b de type $\sigma \to \tau$, c de type σ), et $a \overset{*}{=} b$ ssi $a\{\tau\} \overset{*}{=} b\{\tau\}$ pour tout τ (a et b de type universel).

Remarquons en passant que la condition sur le type $()$ élimine les "modèles" obtenus à l'aide des η-réductions.

Il est relativement facile de construire des modèles extensionnels standard pour les systèmes $\mathrm{OF}_1, \mathrm{OF}_2, \ldots$, au moyen des structures $\mathrm{HEO}_1, \mathrm{HEO}_2, \ldots$ (voir seconde partie).

(Bien entendu, la notion de modèle que nous utilisons ne diffère pas sensiblement de la notion habituelle de modèle d'une théorie formelle ; les modèles que nous considérons ici ne sont autres que des cas particuliers de modèles (au sens habituel) de théories formelles associées aux systèmes fonctionnels, dans us sens qui sera précisé en IV.B)

Bien que HEO_1 soit défini (localement) par des méthodes arithemétiques, on peut avoir l'impression vague suivante : les modèles extensionnels "intéressants" de OF_1 sont en rapport avec les modèles de la théorie des types. Cette idée assez floue est précisée au moyen du résultat élémentaire : *8*

D'un modèle extensionnel de OF_1 contenant la fonction caractéristique de l'égalité $\overset{*}{=}$, on déduit un modèle de la théorie des types (et réciproquement). Ce qu'on a dans la tête en parlant de modèle intéressant (et qui peut comporter des exigences variées, par exemple possibilité de passer à la limite, etc. . .) n'est donc rien d'autre que de pouvoir considérer l'égalité comme un élément du modèle.

Pour OF_2, on a par contre le résultat négatif suivant : il n'y a pas de modèle extensionnel de OF_2 contenant la fonction caractéristique universelle de l'égalité (c'est a dire un E de type $\wedge\alpha\ (\alpha, \alpha \to ())$, tel que pour tout τ, $E\{\tau\}$ soit la fonction caractéristique de l'égalite sur τ). On montre en effet qu'on en déduirait un modèle d'une théorie formelle U, que l'on sait par ailleurs être incohérente. (Voir III.Annexe)

En particulier, le résultat précédent nous donne une idée précise des limitations que nous pouvons rencontrer en essayant de préciser la notion vague d'uniformité.

IV. Équations dans les systèmes fonctionnels

Soient a et b deux termes de type o. Si a et b sont clos, l'*équation* $a = b$ a une signification évidente (l'égalité des formes normales). Pour donner une signification aux équations dans le cas général, on remarque d'abord que les seules équations qui nous intéressent vraîment sont celles où a et b ne peuvent prendre que les valeurs 0 et 1, c'est à dire où a et b dont isomorphes à des fonctions caractéristiques de

prédicates. Aussi, introduit-on un type spécial, noté $()$, à deux éléments V et F, qui représentent les deux alternatives de la logique classique.

1) Vérité d'une équation dans un modèle

On dira que l'équation $a = b$ est vraie dans $(M, \overset{*}{=})$ ssi pour toute suite C d'éléments de M telle que $a[C]$ et $b[C]$ sont clos, $a[C] \overset{*}{=} b[C]$. ‖ Dans la pratique, on 9
peut considérer le modèle particulier $(M_0, =_0)$ des termes clos muni de l'égalité des formes normales. Dans le cas des systèmes avec arithmétisation ou Bar-récursion, les équations associées aux interprétations fonctionnelles seront vérifiées uniquement dans $(M_0, =_0)$. Pour les autres systèmes, il est plus intéressant de demander que les équations soient vérifiées dans certaines classes de modéles assez généraux.

2) Validité d'une équation

On dira que l'équation $a = b$ est *valide* (pour la validité intentionnelle faible, VIF) si $a = b$ est vraie dans tous les modèles du système. On montre que la validité VIF d'une équation est décidable. Pour celà on introduit des nouvelles règles de réduction (pour les termes contenant des variable libres) réduisant certains termes de type $()$ (par exemple les variables de ce type) en V ou F. La donnée d'un ensemble cohérent de telles règles s'appelle *valuation*. Pour chaque valuation $\mathbb{L}$, on a donc une notion de $\mathbb{L}$-réduction, $\mathbb{L}$-forme normale. On prouve que $a = b$ est VIF ssi a et b ont même forme normale pour tout $\mathbb{L}$, et la décidabilité est alors un corollaire évident.

Mais en général, pour les besoins de l'interprétation fonctionnelle, on a besoin d'une définition de la validité d'une équation plus restrictive que VIF. (Par exemple VIF ne vérifie pas les axiomes de l'égalité pour le type o, et on ne sait pas si leur adjonction préserve la décidabilité ; à fortiori, VIF n'est pas clos par rapport à l'induction sur les équations.)

On est amené à définir une extension VI de VIF, qui est close par rapport à ce schéma. Il en résulte que VI est indécidable. VIF et VI sont aussi définissables à l'aide d'une axiomatique (voir tableau).

V. Démonstrations de cohérence pour les systèmes formels

Une démonstration formelle de cohérence pour une théorie T (dont nous sommes convaincus par ailleurs de la cohérence) peut avoir de nombreux à côtés intéressants.

Par exemple, soit T une théorie ($T = \mathrm{HA}_1, \mathrm{HA}_2, \ldots$), $T^{(i)}$ un sous-système *fini* (voir plus loin) de T. Il est possible de formaliser la démonstration la plus bête de la cohérence de $T^{(i)}$ dans T en construisant (localement) un modèle de $T^{(i)}$ dans T. Ceci est possible grâce au résultat bien connu de Gentzen (la propriété de la sous-formule). Pour chaque énoncé A, nous obtenons une démonstration formelle du résultat "si A est prouvable dans $T^{(i)}$, alors A". Ce résultat, et les formulations diverses qu'on peut en donner, est connu sous le nom de *schéma de réflexion*.

Ainsi, une des applications les plus simples des preuves formelles de cohérence est le schéma de réflexion pour les sous-systèmes finis. Il permet de donner des démonstrations locales suffisemment générales de résultats qui ne sont pas globalement formalisables.

On définit les *sous-systèmes finis* des systèmes fonctionnels, en restreignant l'application de certains schémas (récursion, extraction, etc. . .) : on demande que dans REC^{σ}, $a\{\sigma\}$, . . . , le type σ appartienne à un ensemble de types engendré par un nombre fini d'entre eux, au moyen des opérations de changement de nom des variables libres et de substitution réciproque. Prenons par exemple le cas de OF_2 : le théorème de réductibilité pour OF_2 n'est pas démontrable dans HA_2. Si par contre, on considère un sous-système fini de OF_2, ${\mathrm{OF}_2}^{(i)}$, on voit immédiatement que le théorème de réductibilité pour ce système se démontre, pour chaque fonctionnelle a, à partir d'un certain nombre de compréhensions et d'inductions, ne dépendant pas de a, et que cet ensemble d'axiomes constitue un sous-système fini de HA_2, soit ${\mathrm{HA}_2}^{(j)}$. En appliquant le schéma de réflexion de ${\mathrm{HA}_2}^{(j)}$ dans HA_2, on en deduit une démonstration dans HA_2 de "tout terme de ${\mathrm{OF}_2}^{(i)}$ est absolument normalisable". 10

D'autre part, en formulant les systèmes logiques en termes de *systèmes de déduction naturelle*, on aboutit au concept essentiel de *démonstration sans coupure* (à ne pas confondre avec la notion similaire pour les systèmes de *séquents*).

Les inférences logiques sont divisées en deux catégories : les *introductions* et les *éliminations*. Les *coupures* sont rencontrent en général quand une élimination suit immédiatement une introduction. Les démonstrations sans coupure ont des propriétés très intéressantes, spécialement pour les systèmes du premier ordre (propriété de la sous-formule). Leur intérêt, pour les systèmes d'ordre supérieur, quoique plus limité, est cependant indéniable.

On montre élémentairement qu'il existe un isomorphisme entre systèmes de déduction naturelle et certains systèmes fonctionnels. L'image isomorphique des déductions sans coupure n'est autre que l'ensemble des formes normales. La transportée par cet isomorphisme de la relation de réduction définit le processus de *normalisation*. Le théorème de réductibilité pour les systèmes fonctionnels trouve alors son analogue dans le théorème de *normalisation forte*.

Les *sous-systèmes finis* des systèmes de déduction sont obtenus au moyen de cet isomorphisme, à partir de la notion similaire déjà définie pour les systèmes fonctionnels.

Il y a plusieurs façons non équivalentes de dire qu'un système de déduction naturelle admet "suffisemment" de démonstrations sans coupures : considérons les énoncés

(H) (Hauptsatz) Si A est démontrable, il est démontrable sans coupure.

(N) (Normalisation faible) Si $\mathbb{D}$ est une démonstration de A, on peut normaliser $\mathbb{D}$ en une démonstration sans coupure.

(NF) (Normalisation forte) L'ensemble des suites de normalisations strictes de A sont fini.

En général, (NF) $\Rightarrow$ (N) $\Rightarrow$ (H) ; des exemples simples montrent que les implications réciproques ne sont pas toujours vraies. Par exemple, $\|$ soit T un système de déduction dans lequel il existe un A ne vérifiant pas (N). Soit T' le système obtenu en ajoutent l'axiome (règle d'introduction sans prémisse) A. Alors A vérifie (H) dans T', mais pas (N). 11

Le problème du choix des règles de réduction se traduit isomorphiquement par celui du choix de la procédure de normalisation. On débouche sur le problème délicat de l'*identité de preuves*. Pour plus d'informations sur ce sujet, voir Kreisel 1970 et Prawitz 1970.

Pour les sous-systèmes finis, le théorème de normalisation forte est démontrable dans le système global.

Soit T une théorie (par exemple $T = \text{HA}_2$), $T^{(i)}$ un sous-système finis de T. On prouve formellement dans T que, si $\exists x\ A$ est démontrable dans $T^{(i)}$ ($\exists x\ A$ clos), alors il existe un n tel que $A[\overline{n}]$ soit prouvable dans $T^{(i)}$. Ce résultat se généralise de diverses façons, à l'aide du schéma de réflexion.

VI. Interprétations fonctionnelles

Soit T une théorie ($T = \text{HA}_1, \text{HA}_2$), F un système fonctionnel ($F = \text{OF}_1, \text{OF}_2$). L'interprétation fonctionnelle de T dans F traitée dans cet exposé associe à tout énoncé A de T une expression A^* de la forme $\exists x^\sigma\ \forall y^\tau\ E_A(x, y)$, où E_A est une équation de F. La fonction $[\![A \mapsto E_A]\!]$ est récursive primitive. On dit $\exists x\ \forall y\ E_A(x, y)$ est *valide* ssi on peut trouver a (ne contenant pas y) tel que $E_A(a, y)$ soit vérifiée (voir IV).

L'interprétation fonctionnelle vérifie alors

– si A est prouvable dans T, A^* est valide

– $\bot^*$ n'est pas valide

En particulier, on peut caractériser les fonction récursives qu'on peut montrer être totales dans T (*fonctions récursives prouvables de* T) à l'aide des systémes fonctionnels. Ainsi, les *graphes* des fonctions récursives prouvables de HA_n sont exactement les graphes associés aux fonctions (fonctionnelles de type $0 \to 0$) de OF_n.

L'interprétation de T dans F assure aussitôt, modulo le théorème de réductibilité pour F, la cohérence de T.

Par exemple, soit T la théorie $\text{HA}_n + (\text{CT}_n)$, où (CT_n) est une forme forte de la thèse de Church. T s'interprète dans le système fonctionnel OFA_n. D'autre part le théorème de réductibilité pour OFA_n est prouvable localement (c'est à dire pour les sous-systèmes finis) dans HA_n. On en déduit immédiatement la cohérence relative de $\text{HA}_n + (\text{CT}_n)$ par rapport à HA_n. (CT_n) exprime formellement que toute fonction calculable est définie par une fonction de OFA_n. En particulier, la *négation* du théorème de réductibilité pour OFA_n est conséquence formelle de (CT_n).

Puisqu'il existe en général des énoncés non prouvables dans T, mais dont l'interprétation est valide, on en déduit non seulement leur cohérence relativé mais aussi leur caractère conservatif par rapport à certaines classes d'énoncés, et des résultats de clôture, par exemple : 12

Le Principe de Markov est interprétable dans OF_n, alors qu'il n'est pas dérivable dans HA_n. En conbinant ce fait avec l'élimination des coupures et le schéma de réflexion, on démontre la clôture de HA_n pour la *règle* de Markov.

Finalement, les interprétations fonctionnelles permettent dans certains cas une réduction du problème de la prouvabilité dans T à la solution d'une équation de F d'un type particulier, notemment pour les énoncés purement universels.

Je tiens à remercier le Professeur Kriesel pour ses suggestions (notemment les résultats de V.sec.5 et VI.sec.2 répondent à des problèmes qu'il m'a signalés) et ses critiques (par exemple les résultats de II.sec.5, III.Ann.B, répondent à ses critiques touchant à un manque de clarté dans une version préliminaire).

Je remercie également M. Reznikoff pour ses encouragements et pour avoir vérifié un certain nombre de démonstrations.

Je remercie finalement le Professeur Choquet pour avoir eu la gentillesse de diriger mon sujet de thèse complémentaire.

Conseils généraux de lecture

Le rédaction de cet exposé étant assez uniforme, il est peut être bon de donner quelques points de repères. 1

Approximativement, on peut dire que les résultats que nous exposons sont centrés sur la dualité 1er ordre / second ordre, l'accent étant plutôt mis sur le second ordre. Celà signifie, qu'au niveau des systèmes fonctionnels, le lecteur aura avantage à se concentrer en première lecture sur $\textsc{of}_1$ et $\textsc{of}_2$, au niveau des systèmes formels, sur $\textsc{ha}_1$ et $\textsc{ha}_2$.

En seconde lecture, on pourra se concentrer sur $\textsc{ofa}_1$ et $\textsc{ofb}_1$.

Les autres systèmes $\textsc{of}_n$, $\textsc{ofa}_n$, $\textsc{ofb}_n$, ainsi que les $\textsc{ha}_n$, n'offrent alors plus de difficulté spéciale, quand les cas particuliers $n = 1$ et $n = 2$ ont été bien compris.

Nous avons adjoint un certain nombre de tableaux descriptifs et comparatifs pour les cas $n = 1$ et $n = 2$.

PREMIÈRE PARTIE

Description des systèmes fonctionnels

SECTION I

Ordres et opérateurs

1. La hiérarchie des ordres

1

Les *ordres* sont donnés par la définition inductive :

(1) 0 est un ordre
(2) 1 est un ordre
(3) $()$ est un ordre
(4) si $R_1, \ldots, R_n$ sont des ordres, $(R_1, \ldots, R_n)$ est un ordre.

Remarquons que (3) est un cas particulier de (4).

Il n'est pas nécessaire d'utiliser tous les ordres pour la construction des systèmes fonctionnels. En particulier, 0 et 1 jouent un rôle pratiquement insignifiant dans les deux premières parties.

Si nous n'utilisons qu'un sous-ensemble de la hiérarchie, il faudra bien entendu que ce sous-ensemble contienne $()$, et qu'il vérifie la propriété suivante : si $(\mathbf{R})$ est dans ce sous-ensemble, chaque élément de la suite $\mathbf{R}$ est aussi dans cet ensemble, ceci afin d'éviter des anomalies.

$()$ sera l'ordre des types ; similairement, $((), ())$ sera l'ordre des opérateurs binaires, que, telle la flèche $\to$, agissent sur les types ; similairement, tous les ordres construits sans les clauses (1) et (2) ont une interprétation du même genre.

2. Les opérateurs

Ce que suit donne, pour chaque ordre R, une définition inductive de la notion d'*opérateur d'ordre* R.

2.1 Variables et constantes

– Variables : les variables d'ordre R, ou *indéterminées d'ordre* R $\|$ $\alpha^R, \beta^R, \gamma^R, \ldots,$ sont des opérateurs d'ordre R. On abrègera indéterminée en *ind.* et ind. d'ordre R en R-*ind.* 2

– Constantes : le choix des constantes dépend, bien entendu du système fonctionnel considéré. En général, o sera une constante d'ordre $()$. Dans la seconde partie, nous introduirons une autre constante d'ordre $()$, que nous noterons aussi $()$.

En vue de la troisième partie, remarquons que $=$ est généralement parmi les constantes d'ordre $(0, 0)$, S parmi les constantes d'ordre 1, $\overline{0}$ parmi les constantes d'ordre 0, etc...

2.2 Opérateurs d'ordre 0 (en vue de la troisième partie)

– Si σ et τ sont des opérateurs d'ordres respectifs 1 et 0, $\sigma\,\tau$ est un opérateur d'ordre 0.

– Si σ et τ sont des opérateurs d'ordre 0, $\sigma + \tau$ et $\sigma \cdot \tau$ sont des opérateurs d'ordre 0.

(Ainsi, les opérateurs d'ordre 0 sont définis comme les termes de l'arithmétique, construits en utilisant aussi éventuellement des variables de fonctions.)

2.3 Opérateurs d'ordre $()$, ou *types*

– si σ et τ sont des types, $\sigma \to \tau$ est un type ;

– si σ est un type, si α est une R-ind, $\lambda\alpha\,\sigma$ est un type ;

– si $\sigma, \tau_1, \ldots, \tau_n$, sont des opérateurs d'ordres respectifs $(R_1, \ldots, R_n)$, $R_1, \ldots, R_n$, $\sigma\,\tau_1\,\ldots\,\tau_n$ est un type.

2.4 Opérateurs d'ordre supérieur

Soient $\alpha_1, \ldots, \alpha_n$ des ind. distinctes d'ordres respectifs $R_1, \ldots, R_n$, et σ un type, c'est à dire un opérateur d'ordre $()$, $\lambda\alpha_1 \ldots \alpha_n\,\sigma$ est un opérateur d'ordre $(R_1, \ldots, R_n)$.

3. Indéterminées libres et muettes

3

Dans $\wedge\alpha\,\sigma$ et $\lambda\alpha_1 \ldots \alpha_n\,\sigma$, $\alpha, \alpha_1, \ldots, \alpha_n$ sont muettes. Ceci est suffisant pour pouvoir définir généralement la notion d'indéterminée libre et d'indéterminée muette. Nous faisons grâce au lecteur de cette fastidieuse définition.

Dans ce que suit, nous supposons que tous les opérateurs utilisés sont libérés par un procédé quelconque des complications qui résultent d'un malencontreux usage des variables libres et muettes. Nous convenons d'identifier systématiquement, *au niveau de l'écriture* des objets qui ne diffèreraient que par les appellations différentes de leurs variables muettes. (Ceci est toujours possible, au moyen du "carré de Bourbaki".)

Nous ne définirons pas non plus la notion de substitution (combinatoire) d'un opérateur pour une ind. Avec les conventions précédentes, cette opération s'effectue sans problème.

En général, nous noterons $[\alpha_1, \ldots, \alpha_n/\tau_1, \ldots, \tau_n]\sigma$ le résultat de la substitution *simultanée* pour $\alpha_1, \ldots, \alpha_n$ des opérateurs $\tau_1, \ldots, \tau_n$, sous réserve que l'ordre de α_i soit égal à celui de τ_i $(i = 1 \ldots n)$.

Si la suite α_i est déterminée sans ambigüité par le contexte, nous pourrons adopter la notation $\sigma[\tau_1, \ldots, \tau_n]$.

4. Egalité entre opérateurs

La définition de l'égalité entre opérateurs est une conséquence nécessaire de l'interprétation des opérateurs :

4.1 Signification des opérateurs

– o est le type des entiers.

– $\sigma \to \tau$ est le type des fonctions qui, à tout objet de type σ, associent un objet de type τ.

– $\wedge\alpha\ \sigma$ est le type des objets qui sont définis "uniformément" sur les types $\sigma[\tau]$, pour tout τ.

– $\sigma\ \tau_1 \dots \tau_n$ est le type obtenu en appliquant les arguments $\tau_1, \dots, \tau_n$ à l'opérateur σ. 4

– $\lambda\alpha_1 \dots \alpha_n\ \sigma$ est l'opérateur, qui, appliqué aux arguments $\tau_1, \dots, \tau_n$, prend la valeur $\sigma[\tau_1, \dots, \tau_n]$.

Ainsi, ce que précède suppose l'identification de $(\lambda\alpha_1 \dots \alpha_n\ \sigma)\ \tau_1 \dots \tau_n$ et de $\sigma[\tau_1, \dots, \tau_n]$.

4.2 Opérateurs normaux

Un opérateur est *normal* ssi il ne renferme parmi ses sous-opérateurs (c'est à dire les opérateurs qui lui sont antérieurs dans sa construction) aucun type de la forme $(\lambda\alpha_1 \dots \alpha_n\ \sigma)\ \tau_1 \dots \tau_n$.

4.3 Réduction des opérateurs

La relation de préordre $\preceq$ définie entre opérateurs de même ordre par

(OR 1) $\sigma \preceq \sigma$

(OR 2) si $\sigma \preceq \tau$ et $\tau \preceq \rho$, alors $\sigma \preceq \rho$

(OR 3) si σ' est obtenu à partir de σ par remplacement d'une apparition correspondante d'un τ' tel que $\tau \preceq \tau'$, alors $\sigma \preceq \sigma'$

(OR 4) $(\lambda\alpha_1 \dots \alpha_n\ \sigma)\ \tau_1 \dots \tau_n \preceq \sigma[\tau_1, \dots, \tau_n]$.

est appelée relation de *réduction*. $\sigma \preceq \tau$ se lit "σ se réduit en τ". Si $\sigma \preceq \tau$, et si τ est normal, alors τ est *une forme normale* de σ.

THÉORÈME 1 *Tout opérateur admet une forme normale et une seule, que nous appelerons* **sa** *forme normale.*

DÉMONSTRATION Le théorème 1 peut être obtenu directement par des procédés élémentaires. Mais cependant, il apparaîtra que les opérateurs se plongent isomorphiquement dans une partie très restreinte des systèmes fonctionnels, et que donc, le théorème 1 est conséquence des résultats généraux des deux premières parties. Pour voir que cette justification ne tourne pas en rond, il nous suffit de remarquer que l'image isomorphique de l'ensemble $\|$ des opérateurs est un système fonctionnel qui n'utilise pas la notion générale d'opérateur. 5

4.4 Egalité de deux opérateurs

Nous dirons que deux opérateurs sont *égaux* ss'ils ont la même forme normale. Ceci constitue évidemment une relation d'équivalence, d'après le théorème 1.

Dans ce qui suit, nous convenons d'identifier toujours un opérateur avec sa forme normale. Ainsi, "α est libre dans σ" signifiera que α est libre dans la forme normale de σ. De même, $\sigma[\tau]$ représentera la forme normale de $\sigma[\tau]$, tel qu'il a été défini en 3.

Cependant nous n'avons pas défini les opérateurs non-normaux pour le plaisir de nous restreindre ensuite aux opérateurs normaux. En fait dans certains cas il s'avère malaisé de raisonner directement sur les opérateurs normaux. Le détour par les opérateurs non normaux sera alors simplificateur. Cette situation se présentera lorsqu'il s'agira de définir des propriétés par induction sur les opérateurs, comme la QR de la seconde partie. Il nous faudra bien entendu nous assurer de la clôture de nos définitions par rapport à l'égalité telle que nous venons de la définir.

Dans les cas où la notion d'égalité entre opérateurs utilisée ne ressort pas clairement du contexte, nous réfèrerons à l'égalité des écritures formelles (modulo les changements de variables muettes) de 3. sous le nom d'*égalité syntaxique*, alors que la notion que nous venons de définir sera appelée *égalité modulo les* (OR *i*).

SECTION 2

Les fonctionnelles 1

Ce qui suit est une définition inductive, pour chaque type σ, de la notion de *fonctionnelle* de type σ (ou σ-fonctionnelle, ou σ-*fcl*, ou fcl si le type n'importe pas ou ressort clairement du contexte). On pourra aussi employer l'expression "terme", qui aura le même sens. 1

Bien entendu, de même que l'ensemble des ordres que l'on considère peut être restreint, nous pouvons de même restreindre l'ensemble des opérateurs, et l'ensemble des fonctionnelles. Il y a même des schémas qui sont mutuellement incompatibles, tels B d'une part, GD et RED de l'autre. En section 5, nous donnerons une terminologie pour les systèmes fonctionnels courants.

Une coupure très nette s'établit entre deux types de schémas de construction de fonctionnelles. Certains d'entre eux, les *schémas logiques*, sont liés à la notion de démonstration intuitionniste (voir troisième partie), tout en ayant cependant une signification évidente dans les systèmes fonctionnels proprement dits. Les autres schémas, qui supposent généralement l'utilisation du type 0, n'ont pas d'interprétation complètement satisfaisante en termes de démonstrations ; nous les appellerons *schémas fonctionnels*.

Typiquement, la lambda-abstraction appartient aux schémas logiques, alors que la Bar-récursion est un schéma fonctionnel.

1. Schémas logiques

1.1 Variables

Pour chaque type σ, les *variables* de type σ, $x^\sigma, y^\sigma, z^\sigma, \ldots$ sont des σ-fcl.

Qui dit variables dit tôt ou tard distinction entre variables libres et variables muettes. Nous faisons des conventions similaires à celles de la section 1.3. Il nous suffira d'indiquer quels sont les schémas mutifiants, au cours de notre description.

1.2 Schémas liés à $\to$ 2

– Si a est une τ-fcl, x une σ-variable, $\lambda x\ a$ est une $(\sigma \to \tau)$-fcl.

– Si a est une $(\sigma \to \tau)$-fcl, si b est une σ-fcl, AP $a\,b$ est une τ-fcl, usuellement notée $a(b)$, ou encore $a\,b$.

Exemples et signification

$a(b)$ représente la valeur de la fonction a (voir sec.1.4.1) sur l'argument b. AP est donc une abréviation pour *application*. $\lambda x\ a$, dans lequel x est muet, est obtenu par λ-*abstraction*, et dénote la fonction qui, à tout b de type σ, associe la fcl de type τ $a[b]$.

Cette explication implique que nous aurons plus tard à identifier $\lambda x\ a(b)$ avec $a[b]$. Si σ est un type, d'après 1.1. et 1.2., $\lambda x^\sigma\ x^\sigma$ est une fonctionnelle de type $\sigma \to \sigma$. Les explications que nous avons données montrent que $\lambda x\ x(b)$ doit être entendu (pour le moment heuristiquement) comme b. C'est à dire que $\lambda x^\sigma\ x^\sigma$ représente l'application identique.

1.3 Schémas liés au $\wedge$

– Si a est une σ-fcl, et si α est une R-ind ***stratifiable*** dans a, c'est à dire que α n'est pas libre dans l'indice supérieur d'une variable libre de a, $\mathsf{DT}\alpha a$ est une $(\wedge\alpha\sigma)$-fcl.

– Si a est une $(\wedge\alpha\sigma)$-fcl (α d'ordre R), et si τ est un R-opérateur, $\mathsf{EXT}a\tau$ est une $(\sigma[\tau])$-fcl. On abrège $\mathsf{EXT}a\tau$ en $a\{\tau\}$, ou encore $a\tau$.

Exemples et signification

DT (la ***stratification universelle***) et EXT (***l'extraction***) offrent de grandes similitudes avec λ et AP. Remarquons d'abord que α est muet dans $\mathsf{DT}\alpha\ a$. Si a est une $\wedge\alpha\ \sigma$-fcl, et τ un opérateur, $a\{\tau\}$ désigne la composante de l'ensemble "uniforme" de fcl dont la composante pour τ est $a[\tau]$.

En reprenant l'exemple donné en 1.2., $\lambda x^\alpha\ x^\alpha$ est une $(\alpha \to \alpha)$-fcl ; il en résulte 3
que $\mathsf{DT}\alpha\ \lambda x^\alpha\ x^\alpha$ est une fcl de type $\wedge\alpha\ (\alpha \to \alpha)$, qui correspond à la donnée uniforme de toutes les applications identiques d'un type sur lui-même. Les indications heuristiques sur la signification de DT et EXT nous laissent pressentir l'identification de $(\mathsf{DT}\alpha\ a)\{\tau\}$ et de $a[\tau]$. Ainsi $(\mathsf{DT}\alpha\ \lambda x^\alpha\ x^\alpha)\{\tau\}$ sera $\lambda x^\tau\ x^\tau$. La condition "α stratifiable dans a" vient naturellement de l'isomorphisme entre DT et la règle de $\forall$-introduction (III, sec.1, 2.5.1) ; on peut aussi remarquer que, si par exemple, on avait le droit de former $\mathsf{DT}\alpha\ x^\alpha$, la variable libre x n'aurait plus de type, puisque dès que α est muet, α perd toute identité.

2. Schémas fonctionnels

Tous les schémas présentés ici supposent le type 0. Les schémas 2.2. et 2.3. sont exclusifs les uns des autres.

2.1 Récursion

– $\overline{0}$ est une fcl de type 0.

– S est une fcl de type $0 \to 0$.

– Pour chaque type σ, REC^σ est une fcl de type $(\sigma, (\sigma, \sigma \to \sigma), \sigma \to \sigma)$. (Nous avons utilisé l'abréviation bien connue $(\rho, \sigma \to \tau)$ pour $(\rho \to (\sigma \to \tau))$.)

Exemples et signification

Si on définit le terme $\overline{n}$ par $\overline{p+1} = \mathsf{S}\,\overline{p}$, on voit que le type 0 est le type des entiers, S représentant le successeur de l'arithmétique. De par sa définition, REC^σ peut prendre trois arguments a, b, et c. $\mathsf{REC}(a, b)$ est une fonctionnelle de type $0 \to \sigma$ définie par récursion à partir de a et b, a étant la base de la récursion (c'est à dire que $\mathsf{REC}(a, b, \overline{0})$ dénotera a), et b nous permettant de passer de $\overline{n}$ à $\overline{n+1}$, (c'est

à dire que si $\mathrm{REC}(a, b, \overline{n})$ a été reconnu comme étant c, $\mathrm{REC}(a, b, \mathrm{S}\,\overline{n})$ dénotera $b(c, \overline{n})$).

Si on considère les termes suivants

4

CINQUIÈME PARTIE

L'interprétation de Gödel

22

SECTION I

L'interprétation de Gödel et ses variantes

L'interprétation de Gödel associe à tout énoncé A de l'arithmétique, de l'analyse, 1
de la théorie des ordres, ..., une expression A^* de la forme $\exists x^\sigma\ \forall y^\tau\ A'[x, y, \mathbf{Z}]$, où A' est un énoncé élémentaire d'un système fonctionnel à préciser.

Une expression $\exists x^\sigma\ \forall y^\tau\ A'[x, y, \mathbf{Z}]$ est *valide* si, dans le système fonctionelle de référence, pour la notion de validité associée à ce système (voir IV sec. 6), on peut trouver un terme élémentaire a, ne dépendant que de la suite $\mathbf{Z}$ des variables et indéterminées de A' autres que x et y, tel que

$$\vdash A'[a, y, \mathbf{Z}]$$

Par extension, nous considèrerons encore comme forme possible d'une interprétation fonctionnelle des expressions $\exists \mathbf{x}\ \forall \mathbf{y}\ A'[\mathbf{x}, \mathbf{y}, \mathbf{Z}]$, où $\mathbf{x}$ et $\mathbf{y}$ sont des suites (peut être vides) de variables. Une telle expression est *valide* si on peut trouver une suite $\mathbf{a}$ de fcl élémentaires, ne dépendant que de $\mathbf{Z}$, et telle que

$$\vdash A'[\mathbf{a}, \mathbf{y}, \mathbf{Z}]$$

Si on dispose du produit de types (et nous supposerons que c'est le cas), on peut toujours ramener des expressions de la deuxième forme à des expressions de la première. Une suite vide de quantifications peut être remplacée par une quantification sur une variable qui n'apparaît pas dans A'. De même, l'expression $\exists x_1\ \exists x_2\ \exists \mathbf{x}\ \forall \mathbf{y}\ A'[x_1, x_2, \mathbf{x}, \mathbf{y}, \mathbf{Z}]$ peut être remplacée par $\exists x'\ \exists \mathbf{x}\ \forall \mathbf{y}\ A'[\pi^1 x', \pi^2 x', \mathbf{x}, \mathbf{y}, \mathbf{Z}]$. Si $A'[a_1, a_2, \mathbf{a}, \mathbf{y}, \mathbf{Z}]$ est valide, $A'[\pi^1 a', \pi^2 a', \mathbf{a}, \mathbf{y}, \mathbf{Z}]$ est valide par la substitution $a' = a_1 \otimes a_2$. La réciproque est obtenue au moyen de la substitution $a_i = \pi^i a'$.

Similairement, $A'[\mathbf{a}, y_1, y_2, \mathbf{y}, \mathbf{Z}]$ et $A'[\mathbf{a}, \pi^1 y', \pi^2 y', \mathbf{Z}]$ sont valides simultané- 2
ment.

aAfoo...bar