THE BULLETIN OF SYMBOLIC LOGIC
Volume 2, Number 4, Dec. 1996

UNE PREUVE FORMELLE ET INTUITIONNISTE DU THÉORÈME DE COMPLÉTUDE DE LA LOGIQUE CLASSIQUE

JEAN-LOUIS KRIVINE

Introduction. Il est bien connu que la correspondance de Curry-Howard permet d'associer un programme, sous la forme d'un λ-terme, à toute preuve intuitionniste, formalisée dans le calcul des prédicats du second ordre (voir, par exemple [3]). Cette correspondance a été étendue, assez récemment, à la logique classique moyennant une extension convenable du λ-calcul (voir [1, 4, 5, 6]). Chaque théorème formalisé en logique du second ordre correspond donc à une spécification de programme.

Il se pose alors le problème, en général tout à fait non trivial, de trouver la spécification associée à un théorème donné ; autrement dit, de déterminer le comportement opérationnel commun aux λ-termes associés aux diverses démonstrations formelles du théorème considéré.

Cette question est résolue ici pour le théorème de complétude de la logique classique.

La première étape consiste à formaliser convenablement ce théorème en logique du second ordre. Ce travail est fait complètement dans la section 1. Il a comme sous-produit, peut-être inattendu, de montrer que ce théorème est prouvable en logique *intuitionniste* du second ordre (section 2). Ceci, toutefois, à condition d'introduire une légère variante de la notion de modèle, en admettant *un* modèle supplémentaire trivial, où toute formule est satisfaite.

On notera, à ce sujet, que des preuves intuitionnistes du théorème de complétude de la logique *intuitionniste*, utilisant des variantes de la notion de modèle de Kripke, ont été données par H. Friedman [7] et W. Veldman [8]. On remarquera également qu'un argument de G. Kreisel [2] montre que le théorème de complétude habituel de la logique intuitionniste n'a pas de preuve intuitionniste.

La seconde étape est abordée dans la section 3, où l'on se contente d'énoncer les résultats ; les preuves seront développées dans un article ultérieur. Cette étape consiste à analyser les λ-termes associés à *toutes* les preuves classiques du théorème de complétude.

Received January 20, 1996; revised July 17, 1996.

1079-8986/96/0204-0003/$2.70

On obtient ainsi la spécification associée au théorème de complétude, et elle apparaît quelque peu surprenante: il s'agit, en effet, d'un désassembleur interactif, muni du dispositif de protection des appels système qui est habituellement implémenté dans ce type d'utilitaires.

Ce résultat appelle naturellement des commentaires épistémologiques, que je me propose de faire dans l'article ultérieur annoncé plus haut.

Enfin, je tiens à remercier le rapporteur pour ses observations fort pertinentes, particulièrement à propos des preuves intuitionnistes du théorème de complétude intuitionniste; et également E. Saint-James, pour une conversation éclairante sur l'implémentation des débogueurs.

§1. Formalisation du théorème de complétude. On considère la logique du second ordre, dans le sens habituellement donné à cette expression ; il s'agit de la logique des prédicats usuelle avec plusieurs types de variables : des variables x, y, ... d'individu (ou variables du premier ordre), et des variables X, Y, ... de prédicat n-aire, pour chaque entier n (variables du second ordre). Les seuls axiomes sont le schéma de compréhension.

On se propose d'écrire le théorème de complétude du calcul des prédicats classique comme une formule valide TC de la logique du second ordre comportant les cinq symboles de fonction suivants : 0 (symbole de constante), s (pour "successeur", symbole unaire), $\hookrightarrow$, s (pour "substitution") et @ (trois symboles binaires). Aucun axiome ne sera postulé sur ces fonctions.

Pour orienter le lecteur, il n'est pas inutile d'expliciter tout de suite quel est le modèle "standard" $\mathcal{M}_0$ de la logique du second ordre pour le langage $\{0, s, \hookrightarrow, \mathrm{s}, @\}$, c'est-à-dire le modèle dans lequel la formule TC (qui est, bien sûr, vraie dans $\mathcal{M}_0$, comme dans tout autre modèle) s'interprète comme le théorème de complétude de la logique classique. En considérant d'autres modèles, on obtient toutes sortes de théorèmes de complétude, par exemple pour la logique du second ordre, ou pour la ω-logique, etc...

L'ensemble de base $|\mathcal{M}_0|$ de $\mathcal{M}_0$ est l'ensemble des formules closes du premier ordre d'un langage $\mathcal{L}$ dénombrable comportant une infinité de symboles de constante. Les seuls symboles logiques utilisés dans ces formules sont $\to$ et $\forall$. Le symbole $\hookrightarrow$ est interprété dans $\mathcal{M}_0$ de la façon suivante : $\hookrightarrow(F, G) = (F \to G)$.

On fixe une énumération $\{G_0, G_1, \dots\}$ de $|\mathcal{M}_0|$. Le symbole de constante 0 et le symbole de fonction s sont interprétés dans $\mathcal{M}_0$ respectivement par la formule G_0, et par la fonction $G_n \mapsto G_{n+1}$.

On fixe une bijection $G \mapsto t_G$ de $|\mathcal{M}_0|$ sur l'ensemble des termes clos du langage $\mathcal{L}$. L'interprétation du symbole s dans $\mathcal{M}_0$ est la suivante :

$$\mathrm{s}(A, B) = A \text{ si la formule } A \text{ n'est pas de la forme } \forall x\, A';$$
$$\mathrm{s}(\forall x\, A', B) = A'[t_B/x].$$

L'interprétation du symbole @ dans $\mathcal{M}_0$ est la suivante :

> $@(A, B)$ est une formule close C telle que t_C soit un symbole de constante de $\mathcal{L}$ qui n'apparaît pas dans les formules closes A, B.

Enfin $\mathcal{M}_0$ est un modèle "plein" de la logique du second ordre, autrement dit les variables du second ordre d'arité n décrivent $\mathcal{P}(|\mathcal{M}_0|^n)$.

Bien entendu, ce modèle $\mathcal{M}_0$ n'est donné ici que pour la clarté de l'exposé, et n'est utilisé ni dans l'énoncé, ni dans la preuve du théorème de complétude. On vérifiera simplement que la formule TC a bien l'interprétation voulue dans $\mathcal{M}_0$.

NOTATIONS. Toutes les formules de la logique du second ordre seront écrites avec les seuls symboles logiques $\to$ et $\forall$. On écrira $A_1, \ldots, A_k \vdash A$ pour exprimer que A est conséquence de $A_1, \ldots, A_k$ en logique classique du second ordre. La formule

$$A_1 \to (A_2 \to (\ldots \to (A_k \to A) \ldots))$$

sera souvent écrite $A_1, A_2, \ldots, A_k \to A$. $\perp$ est, par définition, la formule $\forall X\, X$, où X est une variable propositionnelle (variable de prédicat 0-aire). La formule $\neg A$ est, par définition, $A \to \perp$, c'est-à-dire $A \to \forall X\, X$. La formule $A \vee B$ est, par définition,

$$\forall X[(A \to X), (B \to X) \to X]$$

où X est une variable propositionnelle qui n'apparaît pas dans A, B. On utilisera la notation $\exists x\{F_1, \ldots, F_k\}$ pour désigner la formule

$$\forall X\{\forall x[F_1, \ldots, F_k \to X] \to X\}$$

où X est une variable propositionnelle qui n'apparaît pas dans $F_1, \ldots, F_k$.

L'égalité est définie de la façon usuelle en logique du second ordre : $x = y$ est la formule $\forall X(Xx \to Xy)$, où X est une variable de prédicat unaire.

Le terme $\hookrightarrow(x, y)$ sera noté $x \hookrightarrow y$; le terme

$$x_1 \hookrightarrow (x_2 \hookrightarrow (\ldots (x_k \hookrightarrow y) \ldots))$$

sera noté aussi $x_1, x_2, \ldots, x_k \hookrightarrow y$. Enfin, le terme $(x \hookrightarrow y) \hookrightarrow y$ sera noté aussi $y \sqcup x$.

Soient M, J deux variables de prédicat unaire. On désigne par $\mathrm{Mod}(M)$ (lire "M est un modèle") l'ensemble des quatre formules suivantes :

$$\forall xy[M(x \hookrightarrow y) \to (Mx \to My)];$$
$$\forall xy[(Mx \to My) \to M(x \hookrightarrow y)];$$
$$\forall xy[Mx \to M\mathrm{s}(x, y)];$$
$$\forall x[\forall y\, M\mathrm{s}(x, y) \to Mx];$$

ou, ce qui revient au même, les deux formules suivantes :

$$\forall xy[(Mx \to My) \leftrightarrow M(x \hookrightarrow y)];$$
$$\forall x[Mx \leftrightarrow \forall y\, M\mathrm{s}(x,y)].$$

REMARQUE. Il est clair que, dans le modèle standard $\mathcal{M}_0$, une partie M qui satisfait $\mathrm{Mod}(M)$ est :

- ou bien l'ensemble de toutes les formules closes de $\mathcal{L}$;
- ou bien l'ensemble des formules closes de $\mathcal{L}$ satisfaites dans un modèle dont l'ensemble de base est l'ensemble des termes clos de $\mathcal{L}$, où les symboles de fonction de $\mathcal{L}$ ont leur interprétation naturelle.

On désigne par $\mathrm{Ded}(J)$ l'ensemble des formules suivantes (lire "J est clos par déduction") :

$$\forall xy\, J(y, x \hookrightarrow y);\ \forall xyz\, J(x \hookrightarrow y, (x, y \hookrightarrow z), x \hookrightarrow z);$$
$$\forall xy\, J(x \sqcup (x \hookrightarrow y)) \quad \text{(loi de Peirce)};$$
$$\forall xy[J(x \hookrightarrow y), Jx \to Jy] \quad \text{(modus ponens)};$$
$$\forall xy\, J(x \hookrightarrow \mathrm{s}(x,y));$$
$$\forall xy[J(y \sqcup \mathrm{s}(x, @(x,y)) \to J(y \sqcup x)].$$

REMARQUE. Dans le modèle standard $\mathcal{M}_0$, une partie J qui satisfait ces formules est un ensemble de formules closes, qui est clos par les règles de déduction suivantes :

- les règles du système de Hilbert du calcul propositionnel;
- les axiomes $\forall x\, F \to F[t/x]$ pour t terme clos, et $\forall x\, F$ formule close de $\mathcal{L}$;
- la règle d'introduction du quantificateur universel : de $(F[c/x] \to G) \to G$, déduire $(\forall x\, F \to G) \to G$ où c est un symbole de constante qui n'apparaît pas dans F, G.

Le théorème de complétude (du calcul des prédicats) est la formule suivante TC_0 de la logique du second ordre, dans le langage $\{\hookrightarrow, \mathrm{s}, @\}$:

$$\forall x\{\forall M[\mathrm{Mod}(M) \to Mx] \to \forall J[\mathrm{Ded}(J) \to Jx]\}$$

qui exprime que, si une formule x est valide (vraie dans tout modèle), alors il en existe une démonstration (x est dans tout ensemble clos par déduction).

Soit P une constante de prédicat unaire. La formule $TC_0[P]$

$$\forall x\, \{\forall M\, [\mathrm{Mod}(M), P \subset M \to Mx] \to \forall J\, [\mathrm{Ded}(J), P \subset J \to Jx]\}$$

(où $P \subset M$ est, bien sûr, la formule $\forall z\, [Pz \to Mz]$) exprime, plus généralement, que si une formule close x est vraie dans tous les modèles d'un ensemble P de formules, alors il en existe une démonstration avec P comme axiomes

(x est dans le plus petit ensemble de formules contenant P, qui est clos par déduction). Noter que, dans le modèle standard $\mathcal{M}_0$, on devra supposer alors que $@(A, B) = C$, où C est une formule telle que t_C soit un symbole de constante qui n'apparaît pas dans A, B, *ni dans aucune formule de* P.
La formule $\forall P\, TC_0[P]$ est logiquement (et trivialement) équivalente à la formule suivante, notée TC_1 :

$$\forall x\, \forall J\, \{\forall M\, [\mathrm{Mod}(M), J \subset M \to Mx], \mathrm{Ded}(J) \to Jx\}$$

TC_1 et même TC_0 ne sont pas des formules valides de la logique du second ordre. Toutefois, on va montrer (théorème 1) que TC_1 est une conséquence en logique du second ordre, dans le langage $\{0, s, \hookrightarrow, \mathrm{s}, @\}$, de l'axiome $\forall x\ \mathrm{Ent}(x)$ ($\mathrm{Ent}(x)$, qui se lit "x est un entier", désigne la formule

$$\forall X\, [\forall y\, (Xy \to X(sy)), X0 \to Xx];$$

$\forall x\ \mathrm{Ent}(x)$ est donc l'axiome de récurrence). Notons que le modèle "standard" $\mathcal{M}_0$ satisfait évidemment $\forall x\ \mathrm{Ent}(x)$. Notons aussi qu'aucun axiome n'est postulé sur $0, s$ (ni, d'ailleurs, sur les autres symboles de fonction).

Pour énoncer un résultat plus précis, désignons par $\mathrm{Mod}'(M)$ l'ensemble des quatre formules suivantes :

(i) $\forall xy\, [M(x \hookrightarrow y) \to (Mx \to My)]$
(ii) $\forall xy\, [\mathrm{Ent}(x), (Mx \to My) \to M(x \hookrightarrow y)]$
(iii) $\forall xy\, [Mx \to M\mathrm{s}(x, y)]$
(iv) $\forall x\, [\mathrm{Ent}(x), \forall y\, M\mathrm{s}(x, y) \to Mx]$

THÉORÈME 1. *L'énoncé suivant TC du théorème de complétude du calcul des prédicats* :

$$\forall x\, \forall J\, \{\forall M\, [\mathrm{Mod}'(M), J \subset M \to Mx], \mathrm{Ded}(J) \to Jx\}$$

est une formule valide de la logique du second ordre dans le langage $\{0, s, \hookrightarrow, \mathrm{s}, @\}$.

Nous ferons la preuve formelle tout d'abord en logique *classique* du second ordre. Pour faciliter la lecture de cette preuve, donnons-en tout d'abord le schéma intuitif :

Soit a une formule close quelconque non démontrable, et soit $G_0, \ldots, G_p, \ldots$ une énumération des formules closes (G_p est formellement représentée par l'entier p). On définit par récurrence une suite $\phi_0, \ldots, \phi_n, \ldots$ de formules closes, en posant :

$\phi_0 = a$; si $\phi_n \sqcup G_n$ est démontrable, alors $\phi_{n+1} = \phi_n$; sinon : si G_n ne commence pas par $\forall$, alors $\phi_{n+1} = \phi_n \sqcup G_n$; si $G_n = \forall x\, G'_n$, alors $\phi_{n+1} = \phi_n \sqcup G'_n[c/x]$, c étant un symbole de constante qui n'est pas dans ϕ_n, G_n.

On définit alors $\tilde{M}$ comme l'ensemble des formules closes ϕ telles que $\phi_n \sqcup \phi$ soit démontrable pour un entier n. On montre que $\tilde{M}$ est un modèle, au sens défini plus haut, qui ne contient pas la formule a, ce qui termine la démonstration.

Passons maintenant à la preuve formelle du théorème 1. Soit a un symbole de constante. On a à montrer $\forall M\,[\mathrm{Mod}'(M), J \subset M \to Ma], \mathrm{Ded}(J) \vdash Ja$.

On définit la formule à quatre variables libres

$$\begin{aligned}(n,x) \leq (n',x') \equiv \forall Z\, \{&\forall py\, [Z(p,y), J(y \sqcup p) \to Z(sp,y)],\\ &\forall py\, [Z(p,y), \neg J(y \sqcup p) \to Z(sp, \mathrm{s}'(p,y))],\\ &Z(n,x) \to Z(n',x')\}\end{aligned}$$

où Z est une variable de prédicat binaire; $\mathrm{s}'(p,y)$ est, par définition, le terme $y \sqcup \mathrm{s}(p, @(p,y))$.

La formule $(0,a) \leq (n,x)$ est notée $\Phi(n,x)$. Intuitivement, cette formule définit le graphe d'une fonction $x = \phi(n)$, définie sur les entiers par récurrence, par les conditions $\phi(0) = a$; $\phi(sn) = \phi(n)$ si $J(\phi(n) \sqcup n)$; sinon, $\phi(sn) = \mathrm{s}'(n, \phi(n))$.

La formule $\Phi(n,x)$ s'écrit donc

$$\begin{aligned}\forall Z\, \{&\forall py\, [Z(p,y), J(y \sqcup p) \to Z(sp,y)],\\ &\forall py\, [Z(p,y), \neg J(y \sqcup p) \to Z(sp, \mathrm{s}'(p,y))],\\ &Z(0,a) \to Z(n,x)\}.\end{aligned}$$

On définit la formule $\tilde{M}y \equiv \exists nx\{\Phi(n,x), J(x \sqcup y)\}$. Il suffit évidemment de montrer $\mathrm{Ded}(J), \neg Ja \vdash \forall y\,(Jy \to \tilde{M}y), \neg\tilde{M}a, \mathrm{Mod}'(\tilde{M})$. On fait donc, dans toute la suite, les hypothèses $\mathrm{Ded}(J), \neg Ja$.

Preuve de $Jy \to \tilde{M}y$: on a évidemment $\vdash \Phi(0,a)$, et $\mathrm{Ded}(J), Jy \vdash J(a \sqcup y)$. D'où le résultat.

On a $\Phi(n,x) \vdash J(a \hookrightarrow x)$: il suffit, en effet, de prendre $Z(p,y) \equiv J(a \hookrightarrow y)$ dans $\Phi(n,x)$ ($Z(p,y)$ est donc indépendant de p). Les trois hypothèses de $\Phi(n,x)$ sont vérifiées, car on a $J(a \hookrightarrow a)$, et

$$J(a \hookrightarrow y) \to J(a \hookrightarrow \mathrm{s}'(p,y))$$

d'après $\mathrm{Ded}(J)$, puisque $a \hookrightarrow \mathrm{s}'(p,y)$ s'écrit $a, u \hookrightarrow y$, avec

$$u = \mathrm{s}(p, @(p,y)) \hookrightarrow y.$$

On a $\Phi(n,x) \vdash \neg Jx$: il suffit de prendre $Z(p,y) \equiv \neg Jy$ dans $\Phi(n,x)$ ($Z(p,y)$ est donc indépendant de p). La première hypothèse de $\Phi(n,x)$ est trivialement vérifiée. La deuxième se déduit de $\neg J(y \sqcup p) \to \neg J\mathrm{s}'(p,y)$, qui est une conséquence évidente de la dernière formule de $\mathrm{Ded}(J)$. La troisième est $\neg Ja$, qui est vraie par hypothèse.

Preuve de $\neg\tilde{M}a$: de $\tilde{M}a$, on tire $\Phi(n,x)$, $J(x \sqcup a)$. De $\Phi(n,x)$, on tire $J(a \hookrightarrow x)$. De $\mathrm{Ded}(J)$, $J(a \hookrightarrow x)$, $J(x \sqcup a)$, on tire Jx. Enfin, de $\Phi(n,x)$, Jx, on déduit $\perp$.

Il reste à montrer $\mathrm{Mod}'(\tilde{M})$, c'est-à-dire que $\tilde{M}$ satisfait (i), (ii), (iii) et (iv).

$\tilde{M}$ satisfait (iii) : De $\tilde{M}x$, on tire $\Phi(n,z)$ et $J(z \sqcup x)$. D'après l'avant-dernière formule de $\mathrm{Ded}(J)$, on a $J(x \hookrightarrow \mathrm{s}(x,y))$. On en déduit $J(z \sqcup \mathrm{s}(x,y))$ d'après $\mathrm{Ded}(J)$, d'où $\tilde{M}\mathrm{s}(x,y)$. On a ainsi montré

$$\forall xy\,[\tilde{M}x \to \tilde{M}\mathrm{s}(x,y)].$$

$\tilde{M}$ satisfait (ii) :

(1) On a $\tilde{M}y \to \tilde{M}(x \hookrightarrow y)$: de $\tilde{M}y$, on tire $\Phi(n,z)$ et $J(z \sqcup y)$. De $\mathrm{Ded}(J)$, on déduit alors $J(z \sqcup (x \hookrightarrow y))$, d'où $\tilde{M}(x \hookrightarrow y)$.

(2) Preuve de $\mathrm{Ent}(m) \to \tilde{M}m \vee \tilde{M}(m \hookrightarrow h)$.

On a $\Phi(0,a)$ et

$$\forall y\,[\Phi(p,y) \to \Phi(sp,y) \vee \Phi(sp,\mathrm{s}'(p,y))]$$

(on utilise le tiers exclu $J(y \sqcup p) \vee \neg J(y \sqcup p)$ pour obtenir cette dernière formule). On en déduit

$$\forall x\,[\mathrm{Ent}(x) \to \exists y\,\Phi(x,y)],$$

et donc $\exists y\,\Phi(m,y)$. On introduit donc une constante b telle que $\Phi(m,b)$. Par le tiers exclu, on a $J(b \sqcup m) \vee \neg J(b \sqcup m)$.

De $J(b \sqcup m)$, on déduit $\tilde{M}m$, d'après $\Phi(m,b)$.

De $\neg J(b \sqcup m)$, à l'aide de $\Phi(m,b)$, on déduit $\Phi(sm,q)$, où

$$q = \mathrm{s}'(m,b) = b \sqcup \mathrm{s}(m,r),$$

avec $r = @(m,b)$. D'après $\mathrm{Ded}(J)$, on a donc

$$(*) \qquad J(\mathrm{s}(m,r) \hookrightarrow q).$$

Or, d'après l'avant-dernière formule de $\mathrm{Ded}(J)$, on a $J(m \hookrightarrow \mathrm{s}(m,r))$, et donc $J(m \hookrightarrow q)$. Il en résulte $J(q \sqcup (m \hookrightarrow h))$, d'où $\tilde{M}(m \hookrightarrow h)$, d'après $\Phi(sm,q)$.

On a donc montré

$$\mathrm{Ent}(m) \to [\tilde{M}m \vee \tilde{M}(m \hookrightarrow h)].$$

De (1) et (2), on déduit alors

$$\forall xy\,[\mathrm{Ent}(x), (\tilde{M}x \to \tilde{M}y) \to \tilde{M}(x \hookrightarrow y)],$$

ce qui est le résultat voulu.

Par ailleurs, de $(*)$ et de $\mathrm{Ded}(J)$, on déduit $J(q \sqcup (\mathrm{s}(m,r) \hookrightarrow h))$, et donc $\tilde{M}(\mathrm{s}(m,r) \hookrightarrow h)$, d'après $\Phi(sm,q)$. Comme $r = @(m,b)$, on a ainsi montré le

LEMME 2. $\mathrm{Ded}(J) \vdash \mathrm{Ent}(m) \to \tilde{M}m \vee \exists b\, \forall h\, \tilde{M}(\mathrm{s}(m, @(m,b)) \hookrightarrow h)$.

Il reste à montrer que $\tilde{M}$ satisfait (i) et (iv). Le lemme essentiel est :

LEMME 3. $\mathrm{Ded}(J), \Phi(n,x), \Phi(n',x') \vdash J(x \hookrightarrow x') \vee J(x' \hookrightarrow x)$.

On en déduit (i), c'est-à-dire $\tilde{M}(x \hookrightarrow y), \tilde{M}x \to \tilde{M}y$. En effet :
De $\tilde{M}(x \hookrightarrow y)$, on tire $\Phi(n,z)$, $J(z \sqcup (x \hookrightarrow y))$. De $\tilde{M}x$, on tire $\Phi(n',z')$, $J(z' \sqcup x)$. Or

$$\mathrm{Ded}(J), J(z \sqcup (x \hookrightarrow y)), J(z' \sqcup x), J(z' \hookrightarrow z) \vdash J(z \sqcup y),$$

d'où $\tilde{M}y$ d'après $\Phi(n,z)$. Et

$$\mathrm{Ded}(J), J(z \sqcup (x \hookrightarrow y)), J(z' \sqcup x), J(z \hookrightarrow z') \vdash J(z' \sqcup y),$$

d'où, encore, $\tilde{M}y$ d'après $\Phi(n',z')$. Comme on a $J(z \hookrightarrow z') \vee J(z' \hookrightarrow z)$ d'après le lemme 3, on en déduit $\tilde{M}y$.

On en déduit aussi que $\tilde{M}$ satisfait (iv), et, en fait, la formule plus forte suivante :

(iv′) $\forall x\, [\mathrm{Ent}(x), \forall z\, M\mathrm{s}(x, @(x,z)) \to Mx]$.

Pour cela, on suppose $\mathrm{Ent}(m)$ et $\neg\tilde{M}m$; on en déduit

$$\exists b\, \forall h\, \tilde{M}(\mathrm{s}(m, @(m,b)) \hookrightarrow h),$$

d'après le lemme 2, et donc

$$\exists b\, \tilde{M}(\mathrm{s}(m, @(m,b)) \hookrightarrow a).$$

Comme $\tilde{M}$ satisfait (i), ainsi qu'on vient de le voir, on en tire

$$\exists b\, [\tilde{M}\mathrm{s}(m, @(m,b)) \to \tilde{M}a].$$

Or, on a montré plus haut $\neg\tilde{M}a$, d'où, finalement,

$$\exists b\, \neg\tilde{M}\mathrm{s}(m, @(m,b)).$$

Il ne reste donc plus qu'à montrer le lemme 3. On remarque d'abord que la formule $(n,x) \leq (n',x')$ définit un préordre, c'est-à-dire qu'on a

$$(n,x) = (n',x') \vdash (n,x) \leq (n',x');$$
$$(n,x) \leq (n',x'),\ (n',x') \leq (n'',x'') \vdash (n,x) \leq (n'',x'').$$

La preuve est triviale, d'après la définition de $(n,x) \leq (n',x')$.

LEMME 4. $(n,x) \leq (n',x') \vdash \{(n,x) = (n',x')\} \vee \{J(x \sqcup n) \wedge (sn,x) \leq (n',x')\} \vee \{\neg J(x \sqcup n) \wedge (sn, \mathrm{s}'(n,x)) \leq (n',x')\}$.

On rappelle que la formule $x = x'$ s'écrit $\forall X(Xx \to Xx')$, où X est une variable de prédicat unaire. La formule $(n, x) = (n', x')$ est

$$\forall X[X(n, x) \to X(n', x')],$$

où X est une variable de prédicat binaire; elle équivaut à $n = n' \wedge x = x'$.

On fixe n et x, et, dans la formule $(n, x) \leq (n', x')$, on prend

$$Z(p, y) \equiv \{(n, x) = (p, y)\} \vee \{J(x \sqcup n) \wedge (sn, x) \leq (p, y)\} \\ \vee \{\neg J(x \sqcup n) \wedge (sn, \mathrm{s}'(n, x)) \leq (p, y)\}.$$

On vérifie ci-dessous que les trois hypothèses de la formule $(n, x) \leq (n', x')$ sont satisfaites. Sa conclusion $Z(n', x')$ est le résultat cherché.

La troisième hypothèse est $Z(n, x)$, qui est trivialement vérifiée.

On vérifie $Z(p, y), J(y \sqcup p) \to Z(sp, y)$. On suppose donc $Z(p, y), J(y \sqcup p)$; on a alors $(p, y) \leq (sp, y)$. D'après $Z(p, y)$, il y a trois cas :

(1) $(n, x) = (p, y)$; on a alors $J(x \sqcup n)$ (puisqu'on a $J(y \sqcup p)$), et $(sn, x) \leq (sp, y)$ (puisque $(sn, x) = (sp, y)$). Donc, on a $Z(sp, y)$.

(2) $J(x \sqcup n) \wedge (sn, x) \leq (p, y)$; on a $(p, y) \leq (sp, y)$, donc $(sn, x) \leq (sp, y)$ (transitivité de $\leq$), d'où $Z(sp, y)$.

(3) $\neg J(x \sqcup n) \wedge (sn, \mathrm{s}'(n, x)) \leq (p, y)$; on a $(p, y) \leq (sp, y)$, donc $(sn, \mathrm{s}'(n, x)) \leq (sp, y)$ (transitivité de $\leq$), d'où $Z(sp, y)$.

On vérifie maintenant $Z(p, y), \neg J(y \sqcup p) \to Z(sp, \mathrm{s}'(p, y))$. On suppose donc $Z(p, y), \neg J(y \sqcup p)$; on a donc $(p, y) \leq (sp, \mathrm{s}'(p, y))$. D'après $Z(p, y)$, il y a trois cas :

(1) $(n, x) = (p, y)$; on a alors $\neg J(x \sqcup n)$ (puisqu'on a $\neg J(y \sqcup p)$), et $(sn, \mathrm{s}'(n, x)) \leq (sp, \mathrm{s}'(p, y))$ (puisque $(sn, \mathrm{s}'(n, x)) = (sp, \mathrm{s}'(p, y))$). Donc, on a $Z(sp, \mathrm{s}'(p, y))$.

(2) $J(x \sqcup n) \wedge (sn, x) \leq (p, y)$; on a $(p, y) \leq (sp, \mathrm{s}'(p, y))$, donc $(sn, x) \leq (sp, \mathrm{s}'(p, y))$ (transitivité de $\leq$), d'où $Z(sp, \mathrm{s}'(p, y))$.

(3) $\neg J(x \sqcup n) \wedge (sn, \mathrm{s}'(n, x)) \leq (p, y)$; on a $(p, y) \leq (sp, \mathrm{s}'(p, y))$, donc $(sn, \mathrm{s}'(n, x)) \leq (sp, \mathrm{s}'(p, y))$ (transitivité de $\leq$), d'où $Z(sp, \mathrm{s}'(p, y))$. $\dashv$

LEMME 5. $\Phi(n, x), \Phi(n', x') \vdash (n, x) \leq (n', x') \vee (n', x') \leq (n, x)$.

On rappelle que $\Phi(n, x)$ s'écrit $(0, a) \leq (n, x)$. On fixe n' et x', et, dans la formule $\Phi(n, x)$, on prend

$$Z(p, y) \equiv (p, y) \leq (n', x') \vee (n', x') \leq (p, y).$$

On vérifie que les trois hypothèses de $\Phi(n, x)$ sont vérifiées. La conclusion $Z(n, x)$ de la formule $\Phi(n, x)$ est alors le résultat cherché.

La troisième hypothèse est $Z(0, a)$, qui est vérifiée, puisqu'on a $(0, a) \leq (n', x')$ d'après l'hypothèse $\Phi(n', x')$.

On vérifie $Z(p, y), J(y \sqcup p) \to Z(sp, y)$. On a donc les hypothèses $Z(p, y), J(y \sqcup p)$; on a donc $(p, y) \leq (sp, y)$. D'après $Z(p, y)$, il y a deux cas :

(1) $(n', x') \leq (p, y)$; comme $(p, y) \leq (sp, y)$, on a $(n', x') \leq (sp, y)$, d'où $Z(sp, y)$.

(2) $(p, y) \leq (n', x')$; d'après le lemme précédent, et comme on a $J(y \sqcup p)$, il y a deux cas :

(a) $(p, y) = (n', x')$; comme $(p, y) \leq (sp, y)$, on a $(n', x') \leq (sp, y)$, d'où $Z(sp, y)$.

(b) $(sp, y) \leq (n', x')$, ce qui donne immédiatement $Z(sp, y)$.

On vérifie maintenant $Z(p, y), \neg J(y \sqcup p) \to Z(sp, \mathrm{s}'(p, y))$. On a donc les hypothèses $Z(p, y), \neg J(y \sqcup p)$; par suite $(p, y) \leq (sp, \mathrm{s}'(p, y))$. D'après $Z(p, y)$, il y a deux cas :

(1) $(n', x') \leq (p, y)$; comme $(p, y) \leq (sp, \mathrm{s}'(p, y))$, on a $(n', x') \leq (sp, \mathrm{s}'(p, y))$, d'où $Z(sp, \mathrm{s}'(p, y))$.

(2) $(p, y) \leq (n', x')$; d'après le lemme précédent, et comme on a $\neg J(y \sqcup p)$, il y a deux cas :

(a) $(p, y) = (n', x')$; comme $(p, y) \leq (sp, \mathrm{s}'(p, y))$, on a $(n', x') \leq (sp, \mathrm{s}'(p, y))$, d'où $Z(sp, \mathrm{s}'(p, y))$.

(b) $(sp, \mathrm{s}'(p, y)) \leq (n', x')$, ce qui donne immédiatement $Z(sp, \mathrm{s}'(p, y))$.⊣

Pour terminer la preuve du lemme 3, il suffit de montrer

$$(n, x) \leq (n', x') \vdash J(x \hookrightarrow x').$$

Pour cela, on fixe n et x, et, dans la formule $(n, x) \leq (n', x')$, on prend $Z(p, y) \equiv J(x \hookrightarrow y)$ (qui ne dépend donc pas de p). On vérifie les trois hypothèses de la formule $(n, x) \leq (n', x')$; sa conclusion $Z(n', x')$ donne alors le résultat cherché $J(x \hookrightarrow x')$.

La première hypothèse est

$$Z(p, y), J(y \sqcup p) \to Z(sp, y),$$

c'est-à-dire $J(x \hookrightarrow y), J(y \sqcup p) \to J(x \hookrightarrow y)$, qui est trivialement valide.

La deuxième hypothèse est

$$Z(p, y), \neg J(y \sqcup p) \to Z(sp, \mathrm{s}'(p, y)),$$

soit $J(x \hookrightarrow y), \neg J(y \sqcup p) \to J(x \hookrightarrow \mathrm{s}'(p, y))$, qui est conséquence de $\mathrm{Ded}(J)$; en effet, $J(x \hookrightarrow y) \to J(x \hookrightarrow \mathrm{s}'(p, y))$ est conséquence de $\mathrm{Ded}(J)$, puisque $\mathrm{s}'(p, y)$ s'écrit $u \hookrightarrow y$.

La troisème hypothèse $Z(n, x)$ est vérifiée, puisque c'est $J(x \hookrightarrow x)$. ⊣

§2. Preuve intuitionniste du théorème de complétude. Le théorème de complétude peut s'écrire, ainsi qu'on l'a vu plus haut, comme la formule *TC* suivante :

$$\forall x\,\forall J\,\{\forall M\,[\mathrm{Mod}'(M), \forall y\,(Jy \to My) \to Mx], \mathrm{Ded}(J) \to Jx\}.$$

On a montré que TC est une formule valide de la logique classique du second ordre. On se propose de montrer que TC est, en fait, démontrable en logique *intuitionniste* du second ordre.

Nous utiliserons, dans ce paragraphe, la notation $\vdash$ pour la déduction en logique intuitionniste du second ordre (jusqu'ici, ce symbole désignait la déduction en logique classique du second ordre). Rappelons que les seuls symboles logiques utilisés sont $\to$ et $\forall$.

Considérons une nouvelle constante propositionnelle O (constante de prédicat d'arité 0). Pour chaque formule F, on pose $\neg_0 F \equiv (F \to O)$, et on désigne par F^* la traduction de Gödel de F, obtenue en remplaçant, dans F, chaque formule atomique A par $\neg_0 A$. Il est bien connu que, si F est démontrable en logique classique du second ordre, alors F^* l'est en logique intuitionniste (voir [3]). On a donc $\vdash TC^*$, c'est-à-dire :

$$\forall M\,[\mathrm{Mod}'^*(M), \forall y(\neg_0 Jy \to \neg_0 My) \to \neg_0 Ma], \mathrm{Ded}(\neg_0 J) \vdash \neg_0 Ja.$$

Désignons par $\mathrm{Mod}'_1(M)$, $\mathrm{Mod}'_2(M)$, $\mathrm{Mod}'_3(M)$, $\mathrm{Mod}'_4(M)$ les quatre formules (i), (ii), (iii), (iv) de $\mathrm{Mod}'(M)$. On a donc :

$$\mathrm{Mod}_1'^*(M) \equiv \forall xy\,[\neg_0 M(x \hookrightarrow y), \neg_0 Mx \to \neg_0 My]$$
$$\mathrm{Mod}_2'^*(M) \equiv \forall xy\,[\mathrm{Ent}^*(x), (\neg_0 Mx \to \neg_0 My) \to \neg_0 M(x \hookrightarrow y)]$$
$$\mathrm{Mod}_3'^*(M) \equiv \forall xy\,[\neg_0 Mx \to \neg_0 M\mathrm{s}(x,y)]$$
$$\mathrm{Mod}_4'^*(M) \equiv \forall xy\,[\mathrm{Ent}^*(x), \forall y\,\neg_0 M\mathrm{s}(x,y) \to \neg_0 Mx].$$

On a donc $\mathrm{Mod}_1'^*(M) \equiv \mathrm{Mod}'_1(\neg_0 M)$ et $\mathrm{Mod}_3'^*(M) \equiv \mathrm{Mod}'_3(\neg_0 M)$.

Par ailleurs, d'après la forme de la formule

$$\mathrm{Ent}[x] \equiv \forall X\,[\forall y\,(Xy \to Xsy), X0 \to Xx],$$

on a trivialement $\mathrm{Ent}[x] \vdash \mathrm{Ent}^*[x]$. Il en résulte que

$$\mathrm{Mod}_2'^*(M) \vdash \mathrm{Mod}'_2(\neg_0 M) \quad \text{et} \quad \mathrm{Mod}_4'^*(M) \vdash \mathrm{Mod}'_4(\neg_0 M).$$

On a donc $\mathrm{Mod}'^*(M) \vdash \mathrm{Mod}'(\neg_0 M)$. Il en résulte que :

$$\forall M\,[\mathrm{Mod}'(\neg_0 M), \forall y\,(\neg_0 Jy \to \neg_0 My) \to \neg_0 Ma], \mathrm{Ded}(\neg_0 J) \vdash \neg_0 Ja$$

et, par suite, puisque $\forall M\,F(M) \vdash \forall M\,F(\neg_0 M)$:

$$\forall M\,[\mathrm{Mod}'(M), \forall y\,(\neg_0 Jy \to My) \to Ma], \mathrm{Ded}(\neg_0 J) \vdash \neg_0 Ja.$$

En remplaçant J par $\neg_0 J$, on obtient :

$$\forall M\,[\mathrm{Mod}'(M), \forall y\,(\neg_0\neg_0 Jy \to My) \to Ma], \mathrm{Ded}(\neg_0\neg_0 J) \vdash \neg_0\neg_0 Ja.$$

On a $Jy \vdash \neg_0\neg_0 Jy$. Comme la première occurrence de $\neg_0\neg_0 Jy$ est en position négative dans l'expression ci-dessus, on a :

$$\forall M\,[\mathrm{Mod}'(M), \forall y\,(Jy \to My) \to Ma], \mathrm{Ded}(\neg_0\neg_0 J) \vdash \neg_0\neg_0 Ja.$$

On vérifie ci-dessous que $\mathrm{Ded}(J) \vdash \mathrm{Ded}(\neg_0\neg_0 J)$. Il en résulte que

$$\forall M\,[\mathrm{Mod}'(M), \forall y\,(Jy \to My) \to Ma], \mathrm{Ded}(J) \vdash (Ja \to O) \to O.$$

Il suffit alors de prendre $O \equiv Ja$ pour obtenir :

$$\forall M\,[\mathrm{Mod}'(M), \forall y\,(Jy \to My) \to Ma], \mathrm{Ded}(J) \vdash Ja$$

ce qui est le résultat cherché.

Il reste à vérifier que $\mathrm{Ded}(J) \vdash \mathrm{Ded}(\neg_0\neg_0 J)$. En fait, $\mathrm{Ded}(J)$ est un ensemble de six formules $\mathrm{Ded}_i(J)$, $1 \leq i \leq 6$, et on montre que $\mathrm{Ded}_i(J) \vdash \mathrm{Ded}_i(\neg_0\neg_0 J)$ pour chaque i.

Pour $i = 1, 2, 3$ ou 5, on a $\mathrm{Ded}_i(J) \equiv \forall xyz\, J[t(x, y, z)]$, où t est un terme du langage $\{\hookrightarrow, \mathrm{s}, @\}$. Le résultat est alors évident, puisque $J[t] \vdash \neg_0\neg_0 J[t]$.

On a $\mathrm{Ded}_4(J) \equiv \forall xy\,(J[t(x, y)], J[u(x, y)] \to J[v(x, y)])$, donc

$$\mathrm{Ded}_4(\neg_0\neg_0 J) \equiv \forall xy\,(\neg_0\neg_0 J[t(x, y)], \neg_0\neg_0 J[u(x, y)] \to \neg_0\neg_0 J[v(x, y)]).$$

Le résultat se déduit du fait que $A, B \to C \vdash \neg_0\neg_0 A, \neg_0\neg_0 B \to \neg_0\neg_0 C$.

Même démonstration pour $\mathrm{Ded}_6(J) \equiv \forall xy\,(J[t(x, y)] \to J[u(x, y)])$. $\dashv$

§3. Spécification associée au théorème de complétude. Il s'agit maintenant de déterminer le comportement opérationnel d'un programme extrait d'une preuve *quelconque*, en logique classique du second ordre, du théorème de complétude. Pour abréger notablement l'exposé, on ne considérera ici que le théorème de complétude du calcul propositionnel (les résultats sont essentiellement les mêmes pour le calcul des prédicats). De plus, on ne fera aucune démonstration. Un exposé plus détaillé sera publié ailleurs.

3.1. Généralités sur le λ-calcul pur et typé. Les termes du λ-calcul, ou λ-termes sont obtenus, à partir de variables et de constantes (appelées λ-variables et λ-constantes) par les règles de construction suivantes :

- une λ-variable ou une λ-constante est un λ-terme;
- (abstraction) si t est un λ-terme, et x une λ-variable, alors $\lambda x\, t$ est un λ-terme;
- (application) si t, u sont des λ-termes, $(t)u$ est un λ-terme.

On utilisera la notation $(t)u_1u_2\ldots u_m$ pour $(\ldots((t)u_1)u_2\ldots)u_m$.

Pour extraire un λ-terme d'une preuve en logique classique du second ordre, on utilise un système de λ-calcul typé dont les types sont les formules du second ordre (notées $A_1, \ldots, A_m, A, B, \ldots$) d'un langage $\mathcal{L}$. Les preuves dans ce système, que nous appellerons $LC^2[\mathcal{L}]$, sont décrites par les règles suivantes (cf. [3]) :

(1) $\Gamma, x : A \vdash x : A$;

(2) $\Gamma, x : A \vdash t : B \Rightarrow \Gamma \vdash \lambda x\, t : A \to B$;

(3) $\Gamma \vdash t : A, \Gamma \vdash u : A \to B \Rightarrow \Gamma \vdash ut : B$;

(4) $\Gamma, k : A \to B \vdash t : A \Rightarrow \Gamma \vdash (\mathrm{c})\lambda k\, t : A$;

(5) $\Gamma \vdash t : A[x] \Rightarrow \Gamma \vdash t : \forall x\, A[x]$ si la $\mathcal{L}$-variable x d'individu n'apparaît pas dans les formules du contexte Γ;

(6) $\Gamma \vdash t : \forall x\, A[x] \Rightarrow \Gamma \vdash t : A[\tau/x]$, où τ est un terme de $\mathcal{L}$;

(7) $\Gamma \vdash t : A[X] \Rightarrow \Gamma \vdash t : \forall X\, A[X]$ si la $\mathcal{L}$-variable X de prédicat n'apparaît pas dans les formules du contexte Γ;

(8) $\Gamma \vdash t : \forall X\, A[X] \Rightarrow \Gamma \vdash t : A[\Phi/Xx_1 \dots x_n]$, où X est une variable de prédicat n-aire, $x_1, \dots, x_n$ des variables d'individu, et Φ est une formule du second ordre de $\mathcal{L}$.

Γ est un contexte, c'est-à-dire une expression de la forme $x_1 : A_1, \dots, x_m : A_m$, où $x_1, \dots, x_m$ sont des λ-variables; t, u sont des λ-termes, et c une λ-constante, fixée une fois pour toutes.

Une preuve en logique *intuitionniste* du second ordre (c'est-à-dire n'utilisant pas la règle (4)) donne un λ-terme usuel (ne comportant pas la λ-constante c).

En particulier, c'est à l'aide de ce système que l'on extrait un λ-terme de la preuve formelle du théorème de complétude effectuée dans la section 1.

Les règles (1) à (4) définissent un système de λ-calcul typé pour la logique classique propositionnelle, que nous noterons LC^0.

Enfin, si $\mathcal{E}$ un ensemble d'équations de la forme $\tau = \tau'$ entre termes de $\mathcal{L}$, on désigne par $LC^2[\mathcal{L}, \mathcal{E}]$ le système de λ-calcul typé défini par les règles (1) à (8) et la règle :

(9) $\Gamma \vdash t : A[\tau/x] \Rightarrow \Gamma \vdash t : A[\tau'/x]$ si $\tau = \tau'$ est une conséquence équationnelle de $\mathcal{E}$.

Il faut maintenant préciser comment on exécute un λ-terme comportant la constante c. Nous appelons *redex* un λ-terme de la forme $(\lambda x\, u)t$ ou $(\mathrm{c})\lambda k\, t$. La variable k qui apparaît dans la deuxième forme est appelée *continuation*.

La stratégie d'exécution adoptée est ce qu'on appelle la réduction faible, qui consiste à réduire le redex de tête, à condition qu'aucun λ ne le précède. On considère donc un λ-terme de la forme $(r)s_1 \dots s_n$ noté aussi $(r)\vec{s}$, où r est un redex (le *redex actif*). La suite finie $(s_1, \dots, s_n) = \vec{s}$ est appelée la *pile courante*. Il y a deux règles d'exécution, suivant la forme du redex r :

(i) $(\lambda x\, u)t\vec{s} \rightsquigarrow (u[t/x])\vec{s}$;

(ii) $(\mathrm{c}\lambda k\, t)\vec{s} \rightsquigarrow (t[k_{\vec{s}}/k])\vec{s}$, où $k_{\vec{s}}$ est une nouvelle constante (on dit que la pile courante $\vec{s}$ a été mémorisée dans la continuation k). Il faut donc donner une règle supplémentaire d'exécution, lorsque la continuation $k_{\vec{s}}$ arrive en tête; c'est :

(iii) $(k_{\vec{s}})t\vec{s'} \rightsquigarrow t\vec{s}$.

La règle (i) n'est autre que la β-réduction usuelle du λ-calcul. C'est une réduction locale, qui ne touche pas à la pile. Par contre, les règles (ii) et (iii), correspondant à C et aux continuations, sont des règles de réduction globales, qui servent à changer de pile courante. Elles correspondent à ce qu'on appelle, en informatique, des "appels système".

On définit enfin une relation d'équivalence, notée $t \sim u$, entre λ-termes comportant la constante C, qui généralise la $\beta\eta$-équivalence ; intuitivement, si $t \sim u$, alors $E[t]$ et $E[u]$ ont le même comportement opérationnel, quel que soit l'environnement E (E est un "λ-terme à trous").

La relation $\sim$ est définie comme la plus petite relation d'équivalence qui contient la $\beta\eta$-équivalence, qui passe au contexte (c'est-à-dire $t \sim t'$ et $u \sim u' \Rightarrow tu \sim t'u'$ et $\lambda xt \sim \lambda xt'$) et telle que :

$$(\mathrm{C})tu \sim (\mathrm{C})\lambda k(t\lambda z(k)(z)u)u;\ (\mathrm{C})\lambda k\, t \sim t \text{ si } k \text{ n'apparaît pas dans } t;$$

$$(k)tu \sim (k)t \text{ et } (k')(k)t \sim (k)t \text{ lorsque } k, k' \text{ sont des continuations.}$$

Ces notions sont proches de celles définies par M. Parigot dans [5, 6] pour sa construction du $\lambda\mu$-calcul.

3.2. Le problème de la spécification. Soit maintenant $\mathcal{L}_0$ le langage $\{0, s, \hookrightarrow\}$. Le théorème de complétude peut s'écrire comme une formule du second ordre dans ce langage :

$$\forall x\, \{\forall M\, [\mathrm{Mod}'(M) \to Mx] \to \forall J\, [\mathrm{Ded}(J) \to Jx]\}$$

Dans $\mathrm{Mod}'(M)$ et dans $\mathrm{Ded}(J)$, on a supprimé les deux dernières formules, puisqu'on se limite au théorème de complétude du calcul propositionnel. Désignons par Θ un λ-terme associé à une preuve de ce théorème.

Soit $\mathcal{E}_0$ l'ensemble des équations entre termes clos de $\mathcal{L}_0$ qui sont vraies dans le modèle standard $\mathcal{M}_0$. On considère un terme clos fixé α de $\mathcal{L}_0$ (qui, dans le modèle standard $\mathcal{M}_0$, représente donc une formule A du calcul propositionnel) tel que l'on ait :

$$(1) \qquad \mathcal{E}_0 \vdash \forall M\, [\mathrm{Mod}'(M) \to M\alpha].$$

Si t est un λ-terme de type (1), c'est-à-dire si l'on a

$$\vdash t : \forall M\, [\mathrm{Mod}'(M) \to M\alpha]$$

dans le système $LC^2[\mathcal{L}_0, \mathcal{E}_0]$, alors Θt est du type :

$$(2) \qquad \forall J\, [\mathrm{Ded}(J) \to J\alpha]$$

c'est-à-dire $\vdash \Theta t : \forall J\, (\mathrm{Ded}(J) \to J\alpha)$ dans $LC^2[\mathcal{L}_0, \mathcal{E}_0]$.

Pour déterminer le comportement opérationnel de Θ (et donc la spécification associée au théorème de complétude), il faut déterminer ce qu'est son argument, c'est-à-dire un λ-terme de type (1), et son résultat, c'est-à-dire un λ-terme de type (2).

Pour le type (2), qui est un type de donnée (cf. [3]), il est aisé de déterminer la nature d'un λ-terme de ce type ; il s'agit de la représentation standard, en λ-calcul, d'un arbre fini qui représente une preuve de la formule A dans le système de Hilbert.

On va donc étudier maintenant les λ-termes de type (1). Pour cela, on introduit un autre système de λ-calcul typé noté LC^0_ρ, qui est une variante de LC^0. Il est défini par les règles suivantes, où ρ_1, ρ_2 sont deux λ-constantes fixées :

(1) $\Gamma, x\colon A \vdash x\colon A$;

(2) $\Gamma, x\colon A \vdash t\colon B \Rightarrow \Gamma \vdash (\rho_1)\lceil A\rceil \lambda x\, t\colon A \to B$;

(3) $\Gamma \vdash t\colon A, \Gamma \vdash u\colon A \to B \Rightarrow \Gamma \vdash (\rho_2)ut\colon B$;

(4) $\Gamma, h\colon A \to B \vdash t[\rho_2 h/k]\colon A \Rightarrow \Gamma \vdash (\mathrm{c})\lambda k\, t\colon A$.

$\lceil A\rceil$ désigne l'entier de Church qui est le numéro de A dans l'énumération fixée des formules propositionnelles (énumération qui permet de définir l'interprétation, dans le modèle standard $\mathcal{M}_0$, des symboles de fonction 0 et s du langage $\mathcal{L}_0$).

Dans la suite, on écrira $\vec{\rho}$ pour (ρ_1, ρ_2), et $\vec{\rho}\colon \mathrm{Mod}'(M)$ pour le contexte :

$$\rho_1\colon \forall xy[\mathrm{Ent}(x), (Mx \to My) \to M(x \hookrightarrow y)];$$
$$\rho_2\colon \forall xy[M(x \hookrightarrow y) \to (Mx \to My)].$$

THÉORÈME 6. *Soit α un terme clos de $\mathcal{L}_0$ représentant la formule A du calcul propositionnel. Si $\vec{\rho}\colon \mathrm{Mod}'(M) \vdash t\colon M\alpha$ dans le système $LC^2[\mathcal{L}_0, \mathcal{E}_0]$, alors il existe $t' \sim t$ tel que $\vdash t'\colon A$ dans le système LC^0_ρ.*

Inversement, si $\vdash t\colon A$ dans le système LC^0_ρ, alors on a $\vec{\rho}\colon \mathrm{Mod}'(M) \vdash t\colon M\alpha$ dans le système $LC^2[\mathcal{L}_0, \mathcal{E}_0]$.

Or, un λ-terme du type (1) est de la forme $\lambda\vec{\rho}\, t[\vec{\rho}]$, et l'on a $\vec{\rho}\colon \mathrm{Mod}'(M) \vdash t[\vec{\rho}]\colon M\alpha$ dans $LC^2[\mathcal{L}_0, \mathcal{E}_0]$. D'après le théorème 6, on voit qu'un λ-terme de type (1) est, à équivalence près, de la forme $\lambda\vec{\rho}\, t[\vec{\rho}]$, et l'on a $\vdash t[\vec{\rho}]$: A dans LC^0_ρ. D'après les règles du système LC^0_ρ, on voit que $t[\vec{\rho}]$ est obtenu en prenant un λ-terme associé à une preuve de A dans LC^0, et en y introduisant les constantes ρ_1, ρ_2 devant chaque abstraction et chaque application respectivement. Toutefois :

(i) les constantes ρ_1, ρ_2 ne sont pas introduites devant les “appels système”, c'est-à-dire qu'on écrit $(\mathrm{c})\lambda k\, u$ au lieu de $((\rho_2)\mathrm{c})(\rho_1)\lambda k\, u$; et ku au lieu de $\rho_2 ku$ lorsque k est une continuation (cf. la règle 4 de LC^0_ρ).

(ii) à la suite de chaque apparition de ρ_1, il faut mettre un entier convenable, qui représente le type de la variable sur laquelle porte l'abstraction : on écrit donc $(\rho_1)\lceil F\rceil \lambda x\, u$ au lieu de $(\rho_1)\lambda x\, u$, où $\lceil F\rceil$ est le numéro du type F de la variable x (cf. la règle 2 de LC^0_ρ).

Du point de vue informatique, on peut interpréter un λ-terme $t[\vec{\rho}]$, typé dans le système LC^0_ρ comme un programme à exécuter en pas à pas : les constantes ρ_1, ρ_2 interrompent à chaque pas l'exécution, et le programme Θ fournit la routine "de service" à exécuter à chaque interruption, par substitution à ρ_1, ρ_2 : l'argument de Θ est, en effet, $\lambda\vec{\rho}\,t[\vec{\rho}]$. L'exécution est, de plus, interactive, car, dans le cas de l'interruption ρ_1, il faut que l'opérateur fournisse une indication de type, sous la forme du numéro d'une formule F.

Par ailleurs, le résultat de l'exécution est un λ-terme du type (2), qui représente le texte d'une preuve, ce qui correspond, en informatique, à un programme source.

On voit, en fin de compte, que le programme Θ opère de la façon suivante : il attend qu'on lui fournisse un programme compilé, il l'exécute en pas à pas, demande en cours de route des indications de type à l'opérateur, et rend finalement un programme source, ayant le même type, c'est-à-dire la même spécification, que le programme donné.

Un tel comportement est tout à fait analogue à celui de l'outil de programmation qu'on appelle couramment "désassembleur".

Cependant, comme on l'a noté en (i) ci-dessus, aucune interruption n'est effectuée avant un appel système. Il est tout à fait remarquable que, dans un véritable désassembleur, on exécute également en pas-à-pas le programme à désassembler, *sauf pour les appels système*. Ceci pour des raisons de sécurité : donner la main à l'opérateur juste avant un appel système, lui permettrait d'agir sur la totalité de la pile, ce qui est inacceptable, puisqu'il pourrait alors bloquer la machine.

On voit donc finalement que la spécification associée au théorème de complétude de la logique classique est celle d'un *désassembleur interactif*, muni du dispositif de protection des appels système qui est d'ordinaire mis en place dans ce genre de programme.

RÉFÉRENCES

[1] V. Danos, J. B. Joinet, and H. Schellinx, *LKQ and LKT: Sequent calculi for second order logic based on dual linear decompositions of classical implication*, ***Proceedings of the workshop on linear logic at Cornell 1993***, Cambridge University Press, 1995.

[2] G. Kreisel, *On weak completeness of intuitionistic predicate logic*, this Journal, vol. 27 (1962), pp. 139–158.

[3] J. L. Krivine, ***Lambda-calcul, types et modèles***, Masson, Paris, 1990, English translation: ***Lambda-calculus, types and models***, Ellis Horwood, 1993.

[4] ———, *Classical logic, storage operators and second order lambda-calculus*, ***Annals of Pure and Applied Logic***, vol. 68 (1994), pp. 53–78.

[5] M. Parigot, *$\lambda\mu$-calculus: an algorithmic interpretation of classical natural deduction*, ***Proceedings of logical programming and automated reasoning, St. Petersbourg***, Lecture Notes in Computer Science, no. 624, Springer-Verlag, 1992, pp. 190–201.

[6] ———, *Classical proofs as programs*, ***Proceedings KGC '93***, Lecture Notes in Computer Science, no. 713, Springer-Verlag, 1993, pp. 263–276.

[7] A. S. Troelstra and D. van Dalen, ***Constructivism in mathematics***, vol. II, North Holland, 1988.

[8] W. Veldman, *An intuitionistic completeness theorem for intuitionistic predicate logic*, this Journal, vol. 41 (1976), pp. 159–166.

EQUIPE DE LOGIQUE MATHÉMATIQUE
UNIVERSITÉ PARIS VII, C.N.R.S.
2 PLACE JUSSIEU 75251 PARIS CEDEX 05, FRANCE

E-mail: krivine@logique.jussieu.fr