Résolution d'équations dans un calcul interprétant la logique classique

Stéphane LE ROUX

stage encadré par Pierre Lescanne, équipe Plume du LIP

Mercredi 30 juin

1 Introduction

La résolution d'équation est un problème très général auquel nous sommes confrontés dès le plus jeune âge. Si on commence dans un cadre algébrique avec des équations du premier degré à une inconnue, on continue avec les équations polynomiales, différentielles, matricielles, etc. L'importance accordée à ce domaine des Mathématiques tiens du fait que bon nombres de problèmes consistent à déterminer les liens existant entre certaines grandeurs (physiques notamment) jugées dignes d'intérêt. La construction de l'équation résulte de la modélisation ou de la résolution d'un problème quelconque modulo des inconnues dont reporte la connaissance à plus tard. On espère alors que les (nouvelles) équations obtenues fourniront suffisamment d'information et permetteront ainsi de déterminer les inconnues.

La problématique des équations et de leur résolution est décrite dans la théorie de l'unification. On se donne un langage dont les objets sont appelés termes et qui contiennent a priori des inconnues. On notera les inconnues en lettres capitales. L'unification la plus simple consiste à égaliser syntaxiquement deux termes dont les inconnues sont seulemement des éléments mais pas des fonctions : si on considère les arbres syntaxiques associés aux termes, les inconnues sont exclusivement situées aux feuilles. On résoud l'équation en remplaçant les inconnues par d'autres termes. Par exemple l'équation $f(a, X) = f(a, b)$ a une solution $X \leftarrow b$. L'unification d'ordre supérieur ne suppose pas la nullité de l'arité des inconnues, c'est-à-dire que les inconnues peuvent donc également représenter des fonctions. Par exemple l'équation $F(a, b) = g(a, b)$ a une solution $F \leftarrow g$. On peut ajouter au problème un système de récriture qui consiste en un ensemble de règles permettant de transformer un terme en un autre terme. Si on se donne un système de récriture, l'unification consiste alors à égaliser deux termes modulo récriture selon le système donné (plus souple que l'égalité syntaxique). Dans un cadre algébrique, le système de récriture peut dériver des propriétés de commutativité et d'associativité par exemple : $x + y$ se récrit en $y + x$. Par exemple l'équation $X + b = b + a$ a une solution $X \leftarrow a$.

La résolution d'équation n'est pas uniquement utile en algèbre. On peut s'intéresser à la résolution d'équation faisant intervenir des fonctions vues comme des programmes. On donne trois exemples tirés de l'informatique : Les programmes informatiques sont aussi des objets que l'on souhaite pouvoir manipuler, transformer, notamment pour l'optimisation de code. Les propriétés qu'on en attend peuvent se traduire par des équations dans un langage dont les termes et les inconnues sont des programmes. Des langages de programmation comme Prolog utilise l'unification pour répondre aux questions qu'on lui pose. L'unification est également nécessaire en déduction automatique (COQ) car elle permet d'automatiser le choix des tactiques intermédaires de la construction d'une preuve. Savoir unifier des programmes est donc souhaitable.

Un programme peut se représenter par un terme du λ-calcul. Or un λ-terme est bien plus facile à manipuler formellement qu'un programme quelconque, d'où un de leurs intérêts. Ainsi en 1975, Gérard Huet (voir [3] et [4]) a résolu le problème de l'unification d'ordre supérieur dans le λ-calcul, avec pour motivation principale l'application à la déduction automatique. Un corollaire de son algorithme est que l'unifiabilité dans ce cadre est semi-décidable. Cet algorithme de Huet a été revisité et expliqué par Snyder [7] et Prehofer [6].

Depuis une dizaine d'années, on voit apparaître des calculs strictement plus puissants que le λ-calcul en ce sens qu'ils interpètent la logique classique alors que le λ-calcul se "cantonne" à la logique intuitioniste. La logique intuitionniste est une logique constructive en ce sens que, par exemple, on doit exhiber quelque chose pour en prouver l'existence. La logique classique propose en plus le raisonnement par l'absurde. Le premier des calculs interprétant la logique classique à apparaître est le $\lambda\mu$ de Parigot ([5]) qui établit une correspondance entre déduction naturelle (naturelle pour un humain) et calcul dans ce langage. Curien et Herbelin [1] ont, quant à eux, défini le $\overline{\lambda}\mu\tilde{\mu}$-calcul qui repose sur le calcul des séquents. Ces nouveaux calculs serviront à définir les langages de programmation de demain. Nous utiliserons la terminologie définie par Ghilezan et Lescanne dans [2] pour parler du $\overline{\lambda}\mu\tilde{\mu}$-calcul.

Ainsi il nous paraît fondamental de comprendre l'unification dans ces nouveaux calculs. Dans une première partie, on commencera par présenter le $\overline{\lambda}\mu\tilde{\mu}$-calcul. Sa non confluence nous incitera à définir un sous-calcul confluent. Dans ce sous-calcul apparaîtront naturellement trois nouvelles règles d'extensionalité. L'extensionalité dénote le fait que "si deux fonctions coïncident alors elles sont égales"(cf la règle η du λ-calcul). Dans une deuxième partie on définira ce qu'on entend par "équation" et "solution". En reprenant une idée de Huet, nous définirons un concept de forme normale adapté au problème qui nous concerne. Dans la troisième partie, cela nous permettra de limiter sans perte de généralité un cadre mathématique rigoureux à l'unification, que ce soit en terme de système étudié ou de solution recherchée. Finalement on présentera en quatrième partie des systèmes corrects et complets de transformation d'équation dédiés à la recherche de solutions. Un théorème nous dira que l'existence de solution est semi-décidable, comme dans le λ-calcul.

Les preuves des résultats sont en annexe si elles ne sont pas mentionnées. Elles apparaissent dans le corps du rapport quand elles sont suffisamment courtes.

2 De nouveaux calculs

2.1 L'existant

2.1.1 Le $\overline{\lambda}\mu\tilde{\mu}$-calcul

Défini par Curien et Herbelin dans [1]. La grammaire est :

Définition 1 (Les termes du $\overline{\lambda}\mu\tilde{\mu}$-calcul)

$$\begin{array}{ll} (calleR) & r ::= x \mid \lambda x.r \mid \mu\alpha.c \\ (calleE) & e ::= \alpha \mid \tilde{\mu}x.c \mid r \bullet e \\ (capsule) & c ::= \langle r \| e \rangle \end{array}$$

Les règles logiques de réduction sont :

Définition 2 (Les règles logiques du $\overline{\lambda}\mu\tilde{\mu}$-calcul)

$$\begin{array}{lrcl} (\lambda) & \langle \lambda x.r \| r' \bullet e \rangle & \longrightarrow & \langle r' \| \tilde{\mu}x.\langle r \| e \rangle \rangle \\ (\mu) & \langle \mu\alpha.c \| e \rangle & \longrightarrow & c[\alpha \leftarrow e] \\ (\tilde{\mu}) & \langle r \| \tilde{\mu}x.c \rangle & \longrightarrow & c[x \leftarrow r] \end{array}$$

On peut interpréter le calcul de cette de cette façon : on peut tout naturellement concaténer la règle (λ) et la règle $(\tilde{\mu})$, ce qui donne la règle :

$$(\lambda') \quad \langle \lambda x.r \| r' \bullet e \rangle \quad \longrightarrow \quad \langle r[x \leftarrow r'] \| e \rangle$$

Cette règle est très proche de la β-réduction du λ-calcul. Ici le calleR est un programme qui nécessite des arguments et le calleE est une pile d'arguments avec un fond de pile. La règle (λ') décrit le transfert de l'argument du haut de la pile choisie vers le programme demandeur.

Un calleR commençant par μ est un programme qui possède a priori des piles d'exécution incomplète et qui cherche à remplacer les fonds de pile par une pile (ou continuation). Il trouve cette pile de remplacement à droite de la capsule où il a été placé.

Un calleE commençant par $\tilde{\mu}$ est un fond de pile actif. On peut dire que le μ et le $\tilde{\mu}$ sont des opérateurs qui sont (parfois) en concurrence pour la prise de contrôle. D'où le non-détermisme.

Pour une interprétation métaphorique animalière et colorée (et pouvant aider à comprendre), voir en annexe A.

Il existe deux règles duales additionelles et qu'on peut qualifier d'extensionnelles, ou comportementales.

Définition 3 (Les règles logiques du $\overline{\lambda}\mu\tilde{\mu}$-calcul)

$$\begin{array}{lrcll} (\eta_\mu) & \mu\alpha.\langle r \| \alpha \rangle & \rightarrow & r & si\ \alpha \notin FV(r). \\ (\eta_{\tilde{\mu}}) & \tilde{\mu}x.\langle x \| e \rangle & \rightarrow & e & si\ x \notin FV(e). \end{array}$$

Les règles de typages ne sont les suivantes :

$$\frac{}{\Gamma, \alpha : A \vdash \alpha : A, \Delta}\,(L-ax) \qquad \frac{}{\Gamma, x : A \vdash x : A, \Delta}\,(R-ax)$$

$$\frac{\Gamma \vdash r : A, \Delta \qquad \Gamma, e : B \vdash \Delta}{\Gamma, r \bullet e : A \to B \vdash \Delta}\,(\to L) \qquad \frac{\Gamma, x : A \vdash r : B, \Delta}{\Gamma \vdash \lambda x.r : A \to B, \Delta}\,(\to R)$$

$$\frac{\Gamma \vdash r : A, \Delta \qquad \Gamma, e : A \vdash \Delta}{\langle r \parallel e \rangle : (\Gamma \vdash \Delta)}\,(cut)$$

$$\frac{c : (\Gamma \vdash \beta : B, \Delta)}{\Gamma \vdash \mu\beta.c : B, \Delta}\,(\mu) \qquad \frac{c : (\Gamma, x : A \vdash \Delta)}{\Gamma, \tilde{\mu}x.c : A \vdash \Delta}\,(\tilde{\mu})$$

2.1.2 L'unification d'ordre supérieure dans le λ-calcul

Pour des rappels voir le livre de Snyder [7] ou de Prehofer [6].

2.2 le $\overline{\lambda\mu\tilde{\mu}}$-calcul

2.2.1 Origine : la notion d'égalité dans le $\overline{\lambda}\mu\tilde{\mu}$-calcul

Si on veut résoudre des équations dans le $\overline{\lambda}\mu\tilde{\mu}$-calcul, il faut d'abord définir "l'égalité". Seulement ensuite, on pourra chercher les substitutions qui égalisent les deux membres d'une équation. Si comme dans le cas du λ-calcul "l'égalité" est définie par la convertibilité relativement à un système de récriture R, i.e $u =_R v$ ssi $u \overset{*}{\leftrightarrow}_R v$, alors un gros problème se pose. En effet soient c et c' deux capsules quelconques. Soient α et x deux variables de types compatibles fraîches, donc n'apparaissant ni dans c ni dans c'. Soit la capsule $\langle \mu\alpha.c \| \tilde{\mu}x.c' \rangle$. Elle se réduit de manière non déterministe de la manière suivante :

$$\begin{array}{ccc} & \langle \mu\alpha.c \| \tilde{\mu}x.c' \rangle & \\ \swarrow & & \searrow \\ c & & c' \end{array}$$

Ainsi, si on définit l'égalité comme dans le λ-calcul, on a $c =_R c'$, cela quels que soient c et c'. Ce n'est pas satisfaisant. Il faut redéfinir le calcul pour qu'il soit confluent. La première idée est alors de ne considérer qu'un sous-calcul du $\overline{\lambda}\mu\tilde{\mu}$-calcul. On comprend que le non-déterminisme a lieu quand à une étape de calcul donnée on peut choisir d'appliquer la règle (μ) ou la règle $(\tilde{\mu})$ au sein de ce qu'on appelle une paire critique (μ et $\tilde{\mu}$ en vis-à-vis). On décide donc de donner la "priorité" à l'une des deux règles.

2.2.2 Définition du $\overline{\lambda\mu\tilde{\mu}}$-calcul

Définition 4 *Soit $c = \langle r \| e \rangle$ une capsule. On appelle r le calleR principal de c et e son calleE principal.*

On définit le calcul $\overline{\lambda\mu\tilde{\mu}}$ en s'interdisant d'appliquer la règle (μ) lorsqu'apparaît un $(\tilde{\mu})$ en "surface" du calleE principal de la capsule. Plus formellement, la grammaire devient donc :

Définition 5 (Les termes du $\overline{\lambda\mu\tilde{\mu}}$-calcul)

$$\begin{array}{l} r ::= x \,|\, \lambda x.r \,|\, \mu\alpha.c \\ e_v ::= \alpha \,|\, r \bullet e_v \\ \tilde{e} ::= \tilde{\mu}x.c \,|\, r \bullet \tilde{e} \\ e ::= e_v \,|\, \tilde{e} \\ c ::= \langle r \| e \rangle \end{array}$$

Les définitions des calleR et des capsules n'ont pas été changées. La ligne concernant les calleE a été remplacée par trois lignes qui distinguent les piles à fond simple et les piles à fond actif. Si la grammaire est différente dans sa présentation (par rapport au $\overline{\lambda}\mu\tilde{\mu}$-calcul), les termes générés sont cependant les mêmes.

Les règles logiques de réduction sont :

Définition 6 (Les règles logiques du $\overline{\lambda\mu\tilde{\mu}}$-calcul)

$$\begin{array}{lrcl} (\lambda) & \langle \lambda x.r \| r' \bullet e \rangle & \longrightarrow & \langle r' \| \tilde{\mu}x.\langle r \| e \rangle \rangle \\ (\mu) & \langle \mu\alpha.c \| e_v \rangle & \longrightarrow & c[\alpha \leftarrow e_v] \\ (\tilde{\mu}) & \langle r \| \tilde{\mu}x.c \rangle & \longrightarrow & c[x \leftarrow r] \end{array}$$

Les règles de typage induites sont les mêmes que celle du $\overline{\lambda}\mu\tilde{\mu}$-calcul.

A partir de maintenant et sauf mention spéciale, on se place dans le calcul simplement typé.

Lemme 1 *Le calcul ainsi défini termine.*

Preuve : C'est un sous-calcul de $\overline{\lambda}\mu\tilde{\mu}$ qui termine. □

Lemme 2 *Le calcul ainsi défini est confluent.*

Preuve : Cette preuve longue et technique sera rédigée ultérieurement. □

Lemme 3 *On peut parler de la forme $\overline{\lambda\mu\tilde{\mu}}$-normale.*

Preuve : Synthèse des lemmes 1 et 2. □

On se rend compte que la règle (λ) introduit un $\tilde{\mu}$ prêt à être utilisé. Or on a la confluence donc on peut tout simplement décider de consommer le $\tilde{\mu}$ dès que l'on a consommé le λ :

$$\langle \lambda x.r \| r' \bullet e \rangle \longrightarrow_{\lambda} \langle r' \| \tilde{\mu}x.\langle r \| e \rangle \rangle \longrightarrow_{\tilde{\mu}} \langle r[x \leftarrow r'] \| e \rangle \quad (x \notin FV(e))$$

Cela revient à modifier la règle (λ) :

Définition 7 (Les nouvelles règles logiques du $\overline{\lambda\mu\tilde{\mu}}$-calcul)

$$\begin{array}{lrcl} (\lambda) & \langle \lambda x.r \| r' \bullet e \rangle & \longrightarrow & \langle r[x \leftarrow r'] \| e \rangle \\ (\mu) & \langle \mu\alpha.c \| e_v \rangle & \longrightarrow & c[\alpha \leftarrow e_v] \\ (\tilde{\mu}) & \langle r \| \tilde{\mu}x.c \rangle & \longrightarrow & c[x \leftarrow r] \end{array}$$

Pour une interprétation métaphorique animalière et colorée (et pouvant aider à comprendre), voir en annexe A.

Dans les grammaires déjà présentées , on avait omis les parenthèses. En effet une syntaxe rigoureuse est souvent peu lisible. Ainsi on présente pour les calleR et calleE deux simplifications successives dans la notation.

Notation 1

$$\begin{array}{rrr} \lambda x_1.(\lambda x_2 \ldots (\lambda x_n.r)\ldots) \textit{ s'écrit} & \lambda x_1 x_2 \ldots x_n.r & \textit{ou } \lambda\overline{x_n}.r \\ r_1 \bullet (r_2 \bullet \ldots \bullet (r_n \bullet e)\ldots) \textit{ s'écrit} & r_1 r_2 \ldots r_n \bullet e & \textit{ou } \overline{r_n} \bullet e \end{array}$$

On étend cette notation $\overline{toto}$ (lire "toto barre") à toute formule comportant de manière non ambigüe une liste et un seul indice principal, i.e les autres indices dépendent de cet indice principal. Cela évite d'écrire "pour i allant de 1 à n etc".

2.3 Similarité comportementale et extensionalité

2.3.1 Le cas du λ-calcul

Dans le λ- calcul, on a rajouté la règle η en plus de la β-réduction qui est l'unique règle logique de réduction.

$$(\eta) \quad \lambda x.Mx \to M \quad \text{ si } x \notin FV(M)$$

Cette règle est en fait une règle comportementale qui prend tout son sens dans le calcul simplement typé : en effet, puisque qu'on écrit Mx c'est que le type de M le destine à recevoir en premier argument un terme du type de x et il en est de même de $\lambda x.Mx$.

Un contexte propice (défini ci-dessous) à ces deux termes est donc de la forme $\ldots[.]a\ldots$ avec a du type de x. Regardons comment se comportent les deux termes dans ce contexte.

$$\ldots(\lambda x.Mx)a\ldots \to_\beta \ldots Ma\ldots$$

Ils engendreront la même forme normale. Ils peuvent donc être considérés comme comportementalement équivalents. On a matérialisé cette équivalence par la règle η.

Voici la définition habituelle de contexte :

Définition 8 (Contexte) *Informellement, un contexte est un terme avec un trou dans lequel on peut placer un autre terme. Formellement, on donne une définition inductive de la grammaire des contextes C :*

$$\begin{array}{l} M, N ::= x \mid \lambda x.M \mid MN \\ C ::= [.] \mid \lambda x.C \mid M[.] \mid [.]N \end{array}$$

Définition 9 (Contexte propice) *Informellement, un contexte propice est un contexte qui donne à un terme considéré (vu comme une fonction) tous les arguments dont ce terme peut avoir besoin. Formellement : soit M de type $\tau_1 \to \ldots \to \tau_n \to \tau_0$ avec τ_0 étant un type de base (sans flèche). Un contexte propice à M est de la forme :*

$$M, N ::= x \mid \lambda x.M \mid MN$$
$$C_p ::= [.]a_1 \ldots a_n \mid \lambda x.C_p \mid MC_p \mid C_pN$$

Avec a_i de type τ_i.

2.3.2 Le cas du $\overline{\lambda}\mu\tilde{\mu}$-calcul

Curien et Herbelin [1] avaient déjà noté une équivalence comportementale entre les termes $\mu\alpha\langle r\|\alpha\rangle$ et r si $\alpha \notin FV(r)$ parce que $\mu\alpha\langle r\|\alpha\rangle$ est de par sa structure destiné à être placé dans une capsule en tant que calleR principal, et à consommer son μ. Dans ce type de contexte "propice" (définit dans le même esprit que pour le λ-calcul) on a la réduction suivante :

$$\langle \mu\alpha.\langle r\|\alpha\rangle \| e\rangle \to_\mu \langle r\|e\rangle$$

On a donc l'équivalence :

$$\mu\alpha.\langle r\|\alpha\rangle \sim r \text{ si } \alpha \notin FV(r)$$

De la même manière :

$$\tilde{\mu}x.\langle x\|e\rangle \sim e \text{ si } x \notin FV(e)$$

2.3.3 Le cas du $\overline{\lambda\mu\tilde{\mu}}$-calcul

Cependant d'autres équivalences comportementales peuvent émerger maintenant que l'on se plonge dans un sous-calcul du $\overline{\lambda}\mu\tilde{\mu}$ (l'application des règles de réduction est restreinte). Cela donnera éventuellement lieu à de nouvelles règles d'extensionnalité.

Dans le λ-calcul simplement typé, la règle η identifie deux termes qui se réduisent en la même forme normale quand on leur donne autant d'arguments que nécessaire. De la même manière dans le $\overline{\lambda\mu\tilde{\mu}}$-calcul simplement typé, on va identifier deux termes qui se réduisent en la même forme normale quand il sont placés dans un contexte propice.

Définition 10 (Contexte propice du $\overline{\lambda\mu\tilde{\mu}}$) *Informellement, c'est un terme à trou de même type que le terme dont il est le contexte propice (pour pouvoir l'insérer) et qui lui propose les arguments qui lui permettent d'évoluer en accord avec sa structure.*

Les cas particuliers ci-dessous donneront sans doute au lecteur une meilleure intuition de ce qu'est un contexte propice.

Dans un premier temps on justifie à nouveau l'équivalence suivante :

$$\mu\alpha.\langle r\|\alpha\rangle \sim r \text{ si } \alpha \notin FV(r)$$

Un contexte propice commun à r et $\mu\alpha\langle r\|\alpha\rangle$ est un contexte permettant au μ d'être consommé. Il est donc de la forme $\langle .\|e_v\rangle$. Le terme $\mu\alpha\langle r\|\alpha\rangle$ s'y comporte comme r. En effet, $\langle\mu\alpha\langle r\|\alpha\rangle\|e\rangle \to_\mu \langle r\|e\rangle$. On décide de matérialiser cette équivalence par une règle de réduction dont le sens est imposé pour des raisons de terminaison :

$$(\eta_\mu) \quad \mu\alpha.\langle r\|\alpha\rangle \to r \text{ si } \alpha \notin FV(r)$$

Si la règle était $r \to \mu\alpha\langle r\|\alpha\rangle$, α frais, alors on pourrait l'appliquer un nombre arbitraire de fois à r, d'où la non terminaison.

De la même manière :

$$(\eta_{\tilde{\mu}}) \quad \tilde{\mu}x.\langle x\|e\rangle \to e \text{ si } x \notin FV(e)$$

Apparition d'une autre équivalence : soient les termes bien typés $s\bullet\tilde{\mu}x.c$ et $\tilde{\mu}x'.c[x \leftarrow \mu\alpha.\langle x'\|s\bullet\alpha\rangle]$. D'après leur type, un contexte propice à ces deux termes est de la forme $\langle\lambda y.r\|.\rangle$. Or $\langle\lambda y.r\|s\bullet\tilde{\mu}x.c\rangle \to\!\to c[x \leftarrow r[y \leftarrow s]]$ et $\langle\lambda y.r\|\tilde{\mu}x'.c[x \leftarrow \mu\alpha.\langle x'\|s\bullet\alpha\rangle]\rangle \to c[x \leftarrow \mu\alpha\langle\lambda y.r\|s\bullet\alpha\rangle] \xrightarrow{*} c[x \leftarrow r[y \leftarrow s]]$ (A noter que l'on utilise la règle (η_μ)).

Pour une interprétation métaphorique animalière et colorée (et pouvant aider à comprendre), voir en annexe A.

On décide de matérialiser cette équivalence par une règle de réduction dont le sens est imposé pour des raisons de confluence :

$$(\eta_e) \quad s\bullet\tilde{\mu}x.c \to \tilde{\mu}x'.c[x \leftarrow \mu\alpha.\langle x'\|s\bullet\alpha\rangle]$$

Si la règle était $\tilde{\mu}x'.c[x \leftarrow \mu\alpha.\langle x'\|s\bullet\alpha\rangle] \to s\bullet\tilde{\mu}x.c$ (+conditions d'application), alors on n'aurait pas la confluence. En effet,

$$\begin{array}{ccc} & \langle y\|\tilde{\mu}x'.\langle z\|\mu\alpha.\langle x'\|s\bullet\alpha\rangle\bullet\beta\rangle\rangle & \\ & \tilde{\mu}\swarrow \quad \searrow\eta_e & \\ \langle z\|\mu\alpha.\langle y\|s\bullet\alpha\rangle\bullet\beta\rangle & & \langle y\|s\bullet\tilde{\mu}x.\langle z\|x\bullet\beta\rangle\rangle \end{array}$$

Encore une équivalence : soient les termes bien typés $\mu\alpha'.c$ et $\lambda x.\mu\alpha.c[\alpha' \leftarrow x\bullet\alpha]$. Les contextes propices à ces deux termes sont de la forme $\langle .\|r\bullet e_v\rangle$. Placés dans ce contexte, les deux termes se réduisent en la même forme normale. On a donc équivalence.

Pour une interprétation, voir en annexe A.

On décide de matérialiser cette équivalence par une règle de réduction dont le sens est imposé pour des raisons de confluence :

$$(\eta_r) \quad \lambda x.\mu\alpha.c[\alpha' \leftarrow x\bullet\alpha] \to \mu\alpha'.c \text{ avec } x,\alpha \notin FV(c).$$

Si la règle était $\mu\alpha'.c \to \lambda x.\mu\alpha.c[\alpha' \leftarrow x\bullet\alpha]$, alors le calcul ne serait pas confluent :

$$\begin{array}{ccc} & \langle\mu\alpha.\langle y\|\alpha\rangle\|\gamma\rangle & \\ & \mu\swarrow \quad \searrow\eta_r & \\ \langle y\|\gamma\rangle & & \langle\lambda x.\mu\alpha'.\langle y\|x\bullet\alpha'\rangle\|\gamma\rangle \end{array}$$

Dernière équivalence : soient les termes $\langle \lambda\overline{x_n}\mu\beta c \| \alpha\rangle$ et $\langle \lambda\overline{x_n}\mu\beta c[\alpha \leftarrow \overline{x_n} \bullet \beta] \| \alpha\rangle$. Dans un contexte propice, le α est censé se transformer, éventuellement en plusieurs étapes de réduction, en $\overline{r_n} \bullet \gamma$ avec une correspondance des types. Les deux termes obtenus se réduisent en la même forme normale. D'où équivalence.

Pour une interprétation, voir en annexe A.

On décide de matérialiser cette équivalence par une règle de réduction dont le sens est imposé pour des raisons de confluence :

$$(\eta_c) \quad \langle \lambda\overline{x_n}.\mu\beta.c \| \alpha\rangle \rightarrow \langle \lambda\overline{x_n}.\mu\beta c[\alpha \leftarrow \overline{x_n} \bullet \beta] \| \alpha\rangle \text{ avec } \alpha \in FV(c)$$

Si la règle était $\langle \lambda\overline{x_n}.\mu\beta.c[\alpha \leftarrow \overline{x_n} \bullet \beta] \| \alpha\rangle \rightarrow \langle \lambda\overline{x_n}.\mu\beta.c \| \alpha\rangle$ (+conditions d'application), alors le calcul ne serait pas confluent :

$$\begin{array}{ccc} & \mu\alpha\langle \lambda x.\mu\beta.\langle \lambda y.\mu\gamma\langle y \| x \bullet \beta\rangle \| \delta\rangle \| \alpha\rangle & \\ {\scriptstyle \eta_\mu}\swarrow & & \searrow{\scriptstyle \eta_c} \\ \lambda x.\mu\beta.\langle \lambda y.\mu\gamma.\langle y \| x \bullet \beta\rangle \| \delta\rangle & & \mu\alpha.\langle \lambda x.\mu\beta.\langle \lambda y.\mu\gamma.\langle y \| \alpha\rangle \| \delta\rangle \| \alpha\rangle \end{array}$$

On a donc pas vraiment le choix quant au sens des règles. On appelle R le système de récriture suivant :

Définition 11 (Les règles du $\overline{\lambda\mu\tilde{\mu}}$-calcul)

(λ)	$\langle \lambda x.r \| r' \bullet e\rangle$	$\longrightarrow$	$\langle r[x \leftarrow r'] \| e\rangle$	
(μ)	$\langle \mu\alpha.c \| e_v\rangle$	$\longrightarrow$	$c[\alpha \leftarrow e_v]$	
$(\tilde{\mu})$	$\langle r \| \tilde{\mu}x.c\rangle$	$\longrightarrow$	$c[x \leftarrow r]$	
(η_μ)	$\mu\alpha.\langle r \| \alpha\rangle$	$\longrightarrow$	r	*si* $\alpha \notin FV(r)$
$(\eta_{\tilde{\mu}})$	$\tilde{\mu}x.\langle x \| e\rangle$	$\longrightarrow$	e	*si* $x \notin FV(e)$
(η_r)	$\lambda x.\mu\alpha.c[\alpha' \leftarrow x \bullet \alpha]$	$\longrightarrow$	$\mu\alpha'.c$	*avec* $x, \alpha \notin FV(c)$
(η_e)	$s \bullet \tilde{\mu}x.c$	$\longrightarrow$	$\tilde{\mu}x'.c[x \leftarrow \mu\alpha.\langle x' \| s \bullet \alpha\rangle]$	
(η_c)	$\langle \lambda\overline{x_n}.\mu\beta.c \| \alpha\rangle$	$\longrightarrow$	$\langle \lambda\overline{x_n}.\mu\beta.c[\alpha \leftarrow \overline{x_n} \bullet \beta] \| \alpha\rangle$	*si* $\alpha \in FV(c)$

Notation 2 *On note $\rightarrow_\eta$ la relation $\rightarrow_{\eta_\mu} \cup \rightarrow_{\eta_r}$.*

2.4 Propriétés

On conjecture la terminaison de $\overline{\lambda\mu\tilde{\mu}}$ muni des cinq règles η_X. ce n'est pas évident même si initialement c'st un sous-calcul de $\overline{\lambda}\mu\tilde{\mu}$ qui termine. En effet, on a rajouté trois règles extensionnelles.

Lemme 4 *$\overline{\lambda\mu\tilde{\mu}}$ muni des cinq règles η_X est confluent.*

Preuve : Il n'existe pas de paire critique. □

Lemme 5 *Les formes R-normales des calleR et capsules ne contiennent pas de $\tilde{\mu}$. Les formes R-normales des calleE en contiennent éventuellement un "en surface", i.e le calleE est de la forme $\overline{r_n} \bullet \tilde{\mu}x.c$ et $\tilde{\mu}$ n'apparaît pas dans c.*

2.5 Sous-calculs du $\overline{\lambda\mu\tilde{\mu}}$-calcul

2.5.1 Le $\overline{\lambda\mu}$-calcul

C'est le $\tilde{\mu}$-free $\overline{\lambda\mu\tilde{\mu}}$-calcul. Pour le définir, il suffit de supprimer dans la grammaire, les règles de réduction et les règles de typage du $\overline{\lambda}\mu\tilde{\mu}$-calcul toutes les partie contenant $\tilde{\mu}$. En fait, c'est aussi le $\tilde{\mu}$-free $\overline{\lambda}\mu\tilde{\mu}$-calcul. Cette remarque nous donne directement et sans coût supplémentaire des règles d'extensionalité du $\overline{\lambda\mu}$-calcul : ce sont celles du $\overline{\lambda\mu\tilde{\mu}}$-calcul n'impliquant pas de $\tilde{\mu}$. Le $\overline{\lambda\mu}$-calcul se présente comme suit :

Définition 12 (Les termes du $\overline{\lambda\mu}$-calcul)

$$\begin{aligned} r &::= x \mid \lambda x.r \mid \mu\alpha.c \\ e &::= \alpha \mid r \bullet e \\ c &::= \langle r \| e \rangle \end{aligned}$$

Définition 13 (Les règles du $\overline{\lambda\mu}$-calcul)

(λ)	$\langle \lambda x.r \| r' \bullet e \rangle$	$\longrightarrow$	$\langle r[x \leftarrow r'] \| e \rangle$	
(μ)	$\langle \mu\alpha.c \| e \rangle$	$\longrightarrow$	$c[\alpha \leftarrow e]$	
(η_μ)	$\mu\alpha.\langle r \| \alpha \rangle$	$\longrightarrow$	r	*si* $\alpha \notin FV(r)$
(η_r)	$\lambda x.\mu\alpha.c[\alpha' \leftarrow x \bullet \alpha]$	$\longrightarrow$	$\mu\alpha'.c$	*avec* $x, \alpha \notin FV(c)$
(η_c)	$\langle \lambda\overline{x_n}.\mu\beta.c \| \alpha \rangle$	$\longrightarrow$	$\langle \lambda\overline{x_n}.\mu\beta.c[\alpha \leftarrow \overline{x_n} \bullet \beta] \| \alpha \rangle$	*si* $\alpha \in FV(c)$

Curien et Herbelin [1] ont montré par une traduction que leur $\overline{\lambda}\mu\tilde{\mu}$-calcul contenait le $\lambda\mu$-calcul de Parigot [5]. La traduction est la suivante :

$$\begin{aligned} & x^{>} = x \\ & (\lambda x.M^{>}) = \lambda x.M^{>} \\ & (MN)^{>} = \mu\alpha.\langle M^{>} \| N^{>} \bullet \alpha \rangle \qquad \alpha \text{ frais.} \\ & (\mu\beta.c)^{>} = \mu\beta.c^{>} \\ & ([\alpha]M^{>}) = \langle M^{>} \| \alpha \rangle \end{aligned}$$

Il est intéressant de noter que tout $\tilde{\mu}$ est absent de cette traduction. On peut donc traduire le $\lambda\mu$-calcul dans le $\overline{\lambda\mu}$-calcul, et donc a fortiori dans le $\overline{\lambda\mu\tilde{\mu}}$-calcul. Ainsi cela montre que le $\overline{\lambda\mu}$-calcul interpète la logique classique (tout comme $\lambda\mu$) et donc a fortiori le $\overline{\lambda\mu\tilde{\mu}}$-calcul aussi. Pour s'en convaincre autrement, il suffit d'exhiber un terme dont le type est la loi de Pierce et ne contenant pas de $\tilde{\mu}$. Celui présenté par Ghilezan et Lescanne dans [2] fera l'affaire.

On remarque que le $\lambda\mu$-calcul contient syntaxiquement le λ-calcul. Ainsi le λ-calcul est contenu dans tout les calculs pré-cités. Nous allons traduire dans nos nouveaux calculs la règle η du λ-calcul.

$$\lambda x.Mx \to M \quad \text{ si } x \notin FV(M)$$

La traduction est :

$$\begin{aligned} & (\lambda x.Mx)^{>} \to M^{>} \quad \text{ si } x \notin FV(M) \\ & \lambda x.\mu\alpha.\langle M^{>} \| x \bullet \alpha \rangle \to M^{>} \quad \text{ si } x \notin FV(M) \quad \text{ frais.} \end{aligned}$$

Cela ressemble étrangement à la combinaison des règles d'extensionalité η_μ et η_r du $\overline{\lambda\mu}$-calcul.

2.5.2 Le $\overline{\lambda\tilde{\mu}}$

La définition est "duale" de celle du $\overline{\lambda\mu}$-calcul. C'est le μ-free. Notons qu'il n'y a pas de règle d'extensionalité car toutes les règles d'extensionalité implique l'opérateur μ. Le calcul le plus général est $\overline{\lambda}\mu\tilde{\mu}$.

2.6 Récapitulatif

On donne un schéma d'inclusion des calculs mentionnés jusqu'à présent.

$$\begin{array}{ccccccccc} & & & & \overline{\lambda\mu} & \rightarrow & \lambda\mu & \rightarrow & \lambda \\ & & & \nearrow & & & & & \\ \overline{\lambda}\mu\tilde{\mu} & \rightarrow & \overline{\lambda\mu\tilde{\mu}} & & & & & & \\ & & & \searrow & & & & & \\ & & & & \overline{\lambda\tilde{\mu}} & & & & \end{array}$$

3 Définition du problème équationnel

Sauf mention spéciale, toutes les définitions et résultats donnés quant aux $\overline{\lambda\mu\tilde{\mu}}$ valent également pour le $\overline{\lambda\mu}$, à condition d'omettre les $\tilde{\mu}$ quant ils apparaissent.

3.1 Enrichissement du calcul

On enrichit le calcul avec des constantes et des inconnues :

Définition 14 *Pour les calleR comme pour les calleE, et pour chaque type correspondant à un théorème de la logique classique, on suppose disposer d'ensembles infinis de variables, de constantes et d'inconnues. On les note respectivement V_r, V_e, K_r, K_e, UK_r et UK_e. On note $V = V_r \cup V_e$, etc.*

Notation 3
- *Les inconnues sont représentées par des lettres majuscules : R, R'... pour les calleR, E, E'... pour les calleE et X, Y d'une manière générale.*
- *On écrit c en indice pour les constantes et les méta-variables de constantes.*
- *On utilisera des méta-variables de terme indicées par a pour signifier un objet atomique, i.e une variable, une constante ou une inconnue.*

Définition 15 (Support) *Pour une application f quelconque de V (ou UK) dans l'ensemble des termes, $f : V \rightarrow L$ ou $f : UK \rightarrow L$, on appelle support les points où l'application n'est pas l'identité : $support(f) = \{x : f(x) \neq x\}$. Pour les applications f de UK dans l'ensemble des parties de V on appelle support l'ensemble des points d'image non vide : $support(f) = \{x : f(x) \neq \emptyset\}$.*

Définition 16 (Instanciation) *Une instanciation ι est une application à support fini de l'ensemble des variables dans l'ensemble des termes.*

Les inconnues seules ne sont pas des objets de la grammaire, alors que les variables et les constantes le sont. On définit un couple inconnue-instanciation.

Notation 4 *Le couple inconnue-instanciation est noté $\iota(X)$, où X est l'inconnue et ι est l'instanciation. Si ι est l'identité, on note X en lieu et place de $\iota(X)$. En général pour décrire ι on ne présente que sa restriction à son support. L'identité sera ainsi parfois notée $\emptyset$ parce que son support est vide.*

Le terme $\iota(X)$ est une notation, i.e ce n'est pas une le résultat de l'évaluation de l'application ι en X. D'ailleurs, $\iota : V \to L$ et non pas $\iota : UK \to L$. On aurait pu tout aussi bien noter (X, ι) ou X^ι en lieu et place de $\iota(X)$. Le terme $\iota(X)$ est considéré comme atomique. De plus il ne se récrit pas en X sinon on perdrait tout l'intérêt de conserver ι. En effet $\iota(X)$ signifie qu'on attend de savoir quel terme est substitué à X pour instancier les variables libres du nouveau terme grâce à ι. Cela suggère que le terme qu'on substitue à X contienne des variables libres. En effet :

Définition 17 *Soit $A : UK \to P(V)$ une application de l'ensemble des inconnues dans l'ensemble des parties finies de V (V est l'ensemble des variables). A est appelé fonction d'autorisation car elle va autoriser une inconnue X a être substituée par un terme contenant des variables choisies dans $A(X)$.*

La grammaire du calcul devient donc la suivante :

Définition 18 (Les termes enrichis du $\overline{\lambda\mu\tilde{\mu}}$-calcul)

$$
\begin{array}{l}
r_a ::= x \mid r_c \mid \iota(R) \\
r ::= r_a \mid \lambda x.r \mid \mu\alpha.c \\
e_a ::= \alpha \mid e_c \mid \iota(E) \\
e_v ::= e_a \mid r \bullet e_v \\
\tilde{e} ::= \tilde{\mu}x.c \mid r \bullet \tilde{e} \\
e ::= e_v \mid \tilde{e} \\
c ::= \langle r \| e \rangle
\end{array}
$$

On décide d'étendre le domaine des instanciations de l'ensemble des variables à l'ensemble des termes. Quand il n'y aura pas d'ambiguité, on abandonnera la notation $\hat{\iota}$ et on écrira ι.

Définition 19 (Extension de l'instanciation)

$$
\begin{array}{l}
\hat{\iota}(x) = \iota(x) \\
\hat{\iota}(\alpha) = \iota(\alpha) \\
\hat{\iota}(r_c) = r_c \\
\hat{\iota}(e_c) = e_c \\
\hat{\iota}(\lambda x.r) = \lambda x.\hat{\iota}(r) \\
\hat{\iota}(\mu\alpha.c) = \mu\alpha.\hat{\iota}(c) \\
\hat{\iota}(\tilde{\mu}x.c) = \tilde{\mu}x.\hat{\iota}(c) \\
\hat{\iota}(\langle r \| e \rangle) = \langle \hat{\iota}(r) \| \hat{\iota}(e) \rangle \\
\hat{\iota}(\iota'(X)) = \iota \circ \iota'(X)
\end{array}
$$

On définit l'α-conversion de la manière suivante :

Définition 20 (α-conversion)

$$\lambda x.r = \lambda y.r[x \leftarrow y]$$
$$\mu\alpha.c = \mu\beta.c[\alpha \leftarrow \beta]$$
$$\tilde{\mu}x.c = \tilde{\mu}y.c[x \leftarrow y]$$

Cet apport de nouveaux objets atomiques ne remet pas en cause le système des règles de réduction. Les propriétés de terminaison et de confluence n'en sont pas affectés. Notons les fonctions de chacun : les variables servent à "abstraire", les inconnues sont destinées à être remplacées par des termes et les constantes sont immuables.

Définition 21
- *On note L le language défini par la grammaire précédente.*
- *On note L_0 le sous-ensemble de L composé des termes clos, i.e ne comportant pas de variables libres.*
- *On note $L^{\iota=\emptyset}$ le sous-ensemble de L composé des termes ne comportant que des instanciations qui sont l'identité.*

Définition 22 *Soit t un terme. On note $FV(t, t' \ldots)$ et $UK(t, t' \ldots)$ les ensembles des variables libres et des inconnues apparaissant dans les termes $t, t' \ldots$.*

3.2 Formes normales η-longues

3.2.1 Intérêt et définition

Pour manipuler les classes d'équivalence par un représentant pratique, on définit donc les formes normales longues de la façon suivante : Informellement, on sature tous les calleR d'arguments "virtuels" grâce aux règles inverses $(\eta_\mu)^{-1}$ et $(\eta_r)^{-1}$. On définit d'abord de manière inductive la forme η_b d'un terme :

Définition 23 (Formes normales η-longues brutes)

$$\eta_b(e_a) = e_a$$
$$\eta_b(r \bullet e) = \eta_b(r) \bullet \eta_b(e)$$
$$\eta_b(r_a) = \lambda\overline{x_n}.\mu\alpha.\langle r_a \| \overline{x_n} \bullet \alpha\rangle, \quad \alpha \textit{ frais de type de base.}$$
$$\eta_b(\lambda x.r) = \lambda x.\eta_b(r)$$
$$\eta_b(\mu\beta.c) = \lambda\overline{x_n}.\mu\alpha.\langle \mu\beta.\eta_b(c) \| \overline{x_n} \bullet \alpha\rangle, \quad \alpha \textit{ frais de type de base.}$$
$$\eta_b(\tilde{\mu}x.c) = \tilde{\mu}x.\eta_b(c)$$
$$\eta_b(\langle r \| e\rangle) = \langle \eta_b(r) \| \eta_b(e)\rangle$$

L'algorithme précédent termine et est défini pour tout terme, qu'il soit capsule, calleR ou calleE. On définit maintenant la forme normale η-longue pour tout terme t. La notation $\downarrow_{\lambda\mu}$ est justifiée par la terminaison et la confluence du calcul sous-jacent (le $\overline{\lambda\mu}$ sans règle d'extensionalité).

Définition 24 (Formes normales η-longues) $\eta(t) = (\eta_b(t \downarrow_R)) \downarrow_{\lambda\mu}$

Notation 5 *On qualifie un objet de $fn\eta l$ pour signifier qu'il est sous forme normale η-longue.*

3.2.2 Quelques propriétés

Lemme 6 *Pour tout terme s et t tels que $s \overset{*}{\leftrightarrow}_R t$ on a : $\eta(s) = \eta(t)$.*

Lemme 7 *Pour tout terme t on a : $\eta(t) \overset{*}{\rightarrow}_\eta t \downarrow_R$.*

Lemme 8 *η est idempotente : $\eta \circ \eta(t) = \eta(t)$ pour tout terme t.*

> **Preuve :** D'après le lemme 7, $\eta(t) \overset{*}{\leftrightarrow}_R t$ pour tout terme t. Donc d'après le lemme 6 on a $\eta \circ \eta(t) = \eta(t)$. □

Lemme 9 *Si un $\tilde{\mu}$ apparaît dans un terme $fn\eta l$ alors c'est en "surface", i.e le terme s'écrit $\overline{r_n} \bullet \tilde{\mu}x.c$ et $\tilde{\mu}$ n'apparaît pas dans c.*

Définition 25 (Surface) *La surface d'une capsule est composée de son calleR principal et calleE principal. La surface d'un calleR est le plus grand sous-terme ne commençant ni par λ ni par μ. C'est une variable calleR ou une capsule. La surface d'un calleR vu comme une pile est composée des éléments de la pile et du fond de la pile. Plus précisément, si la surface de $\overline{r_n} \bullet \alpha$ est composée des $\overline{r_n}$ et de α. De même, la surface de $\overline{r_n} \bullet \tilde{\mu}x.c$ est composée des $\overline{r_n}$ et de c.*

Lemme 10 (Structure de surface des formes normales η-longues) *Les calleR, calleE et capsules $fn\eta l$ sont exactement ceux de la forme :*

1. *$\lambda\overline{x_n}.\mu\alpha.c$ avec α de type de base et c $fn\eta l$.*
2. *– $\overline{r_n} \bullet e_a$ avec les r_i $fn\eta l$ ou*
   – $\tilde{\mu}x.c$ avec c $fn\eta l$.
3. *– $\langle r_a \| \overline{r_n} \bullet e_a \rangle$ avec $\overline{r_n}$ $fn\eta l$ et e_a de type de base ou*
   – $\langle \lambda\overline{x_n}.\mu\alpha.c \| e_a \rangle$ avec $\lambda\overline{x_n}.\mu\alpha.c$ $fn\eta l$.

3.3 Substitutions

Définition 26 (Substitution) *Une substitution est une application $\theta : UK \rightarrow L^{\iota=\emptyset}$ à support fini et qui conserve le type.*

Définition 27 *Soit θ une substitution. On note $D(\theta) = \{X \in UK : \theta(X) \neq X\}$ et $\iota(\theta) = UK(\theta(X))_{X \in D(\theta}$.*

Définition 28 *On définit l'extension $\hat{\theta}$ d'une substitution θ :*

$$\begin{aligned}
&\hat{\theta}(x) = x \\
&\hat{\theta}(\alpha) = \alpha \\
&\hat{\theta}(r_c) = r_c \\
&\hat{\theta}(e_c) = e_c \\
&\hat{\theta}(\lambda x.r) = \lambda x.\hat{\theta}(r) \\
&\hat{\theta}(\mu\alpha.c) = \mu\alpha.\hat{\theta}(c) \\
&\hat{\theta}(\tilde{\mu}x.c) = \tilde{\mu}x.\hat{\theta}(c) \\
&\hat{\theta}(\langle r \| e \rangle) = \langle \hat{\theta}(r) \| \hat{\theta}(e) \rangle \\
&\hat{\theta}(\iota(X)) = \widehat{\hat{\theta} \circ \iota}(\theta(X))
\end{aligned}$$

Lemme 11 *Soient ι une instanciation et θ une substitution, alors $\hat{\theta} \circ \hat{\iota} = \widehat{(\hat{\theta} \circ \iota)} \circ \hat{\theta}$.*

Le résultat précédent, tout comme la définition $\hat{\theta}(\iota(X)) = \widehat{\hat{\theta} \circ \iota}(\theta(X))$, s'explique de cette manière : les inconnues ne sont pas des variables et $\iota(X)$ ne se récrit pas en X. Ainsi quand on écrit $\hat{\theta}(\iota(X))$ on signifie qu'on aurait voulu réaliser l'instanciation ι sur X avant la substition $\hat{\theta}$ mais qu'il était trop tôt : on ne connaissait pas encore X. Une fois que θ nous donne plus d'informations (éventuellement partielles) sur X, on peut réaliser (éventuellement partiellement) ι selon les information fournies.

Définition 29 *Soit $W \subset UK$ un ensemble d'inconnues et R un système quelconque de récriture. On dit que deux substitutions θ et σ sont R- égales sur W ssi $\forall X \in W$ on a $\theta(X) \overset{*}{\leftrightarrow}_R \sigma(X)$. On note alors $\theta =_R \sigma[W]$. Si $R = \emptyset$ alors on note $=_R$ par $=$ qui est en fait la congruence structurelle.*

Définition 30 *Soit W un ensemble d'inconnues et R un système quelconque de récriture. On dit que la substitution σ est R-plus générale que la substitution θ sur W ssi il existe une autre substitution ρ telle que $\forall X \in W$ on ait $\theta(X) \overset{*}{\leftrightarrow}_R \rho \circ \sigma(X)$. On note alors $\sigma \leq_R \theta[W]$. De même on notera $\leq_\emptyset$ par $\leq$.*

Notation 6 *On écrira parfois $[t_1/X_1, \ldots, t_n/X_n]$ ou $[X_i \mapsto t_i]_{i=1}^n$ en lieu et place de θ si $D(\theta) = \cup\{X_i\}$ et $\theta(X_i) = t_i$.*

Notation 7 *Soient deux substitutions θ et σ. Si $D(\theta) \cap D(\sigma) = \emptyset$, on notera*

$$\theta \cup \sigma = X \mapsto \begin{cases} \theta(X) & si\ X \in D(\theta) \\ \sigma(X) & si\ X \in D(\sigma) \\ X & sinon \end{cases}$$

Lemme 12 *[Dans le $\overline{\lambda\mu}$-calcul] Soient θ une substitution et u, v deux termes tels que $u \rightarrow_R v$. Alors $\theta(u) \overset{*}{\rightarrow}_R \theta(v)$.*

Lemme 13 *[Dans le $\overline{\lambda\mu}$-calcul] Soient θ une substitution et u, v deux termes tels que $u \overset{*}{\leftrightarrow}_R v$. Alors $\theta(u) \overset{*}{\leftrightarrow}_R \theta(v)$.*

Preuve : On le montre par récurrence en utilisant le lemme 12. □

Lemme 14 *[Dans le $\overline{\lambda\mu\tilde{\mu}}$-calcul] Soient u, v deux termes tels que $u \rightarrow_R v$ et θ une substitution dont la restriction à $UK(u, v)$ n'introduit ni de $\tilde{\mu}$ ni de variable calleR (i.e $FV(\theta(UK(u, v))) \subset V_e$). Alors $\theta(u) \overset{*}{\rightarrow}_R \theta(v)$.*

Lemme 15 *[Dans le $\overline{\lambda\mu\tilde{\mu}}$-calcul] Soient u, v deux termes tels que $u \overset{*}{\leftrightarrow}_R v$ et θ une substitution dont la restriction à $UK(u, v)$ n'introduit ni de $\tilde{\mu}$ ni de variable calleR (i.e $FV(\theta(UK(u, v))) \subset V_e$). Alors $\theta(u) \overset{*}{\leftrightarrow}_R \theta(v)$.*

Preuve : On le montre par récurrence en utilisant le lemme 14. □

Lemme 16 *Soit θ et σ deux substitutions. Si $\forall X \in UK$, $\hat{\theta}(X) =_R \hat{\sigma}(X)$ alors $\theta =_R \sigma$.*

Preuve : Par construction de $\widehat{\theta}$. □

Lemme 17 *Soit $\sigma = [t_1/X_1 \dots t_n/X_n]$ une substitution et θ une autre substitution telle que $\forall i\, \theta(t_i) \overset{*}{\leftrightarrow}_R \theta(X_i)$. Alors $\theta \circ \sigma \overset{*}{\leftrightarrow}_R \theta$.*

Définition 31 (Substitution normalisée) *Soit θ une substitution. On définit θ_n de la manière suivante :*

$$\theta_n = X \mapsto \begin{cases} X & si\ \theta(X) \overset{*}{\rightarrow}_R X \\ \eta(\theta(X)) & sinon \end{cases}$$

3.4 Systèmes d'équations

Définition 32
- *Soit $A : UK \rightarrow P(V)$ une application de l'ensemble des inconnues dans l'ensemble des parties de V (V est l'ensemble des variables). A est appelé fonction d'autorisation car elle va autoriser une inconnue X à être substituée par un terme contenant des variables choisies dans $A(X)$. Ceci est un privilège car normalement on substitue uniquement des termes clos aux inconnues pour des raisons de capture. Quand on travaille dans $\overline{\lambda\mu\tilde{\mu}}$, alors on impose $A : UK \rightarrow P(V_e)$ où V_e est l'ensemble des variables calleE.*
- *Une équation est une paire (u, v) où u et v sont des termes de notre calcul. On notera souvent $u \overset{?}{=} v$ une telle paire. La notation $\overset{?}{=}$ est symétrique.*
- *On note S un multi-ensemble d'équations.*
- *Un système d'équations est une paire A, S. Parfois on le notera S_A*

La fonction d'autorisation A n'est pas nécessaire dans le λ-calcul mais elle est indispensable ici (on le justifie dans le prochain paragraphe). Ensuite, d'une manière générale, quand on se donne un système d'équations, on a bien l'intention de substituer les inconnues par des termes clos, d'où $A = \emptyset$, ce qui revient à ne pas en parler. En fait A va servir pour les inconnues intermédiaires qu'on introduit dans le système et qui n'apparaîtront plus dans la solution finale. Cependant on trouve plus joli de pouvoir résoudre le problème dans un cadre très général.

A un moment donné de l'unification dans le λ-calcul, on a besoin de factoriser les variables libres d'un terme quelconque pour faire apparaître un terme clos qu'on peut alors remplacer par une inconnue :

$$t =_{\beta\eta} (\lambda\overline{y_n}.t[x_i \leftarrow y_i])\overline{x_n}$$

Les $\overline{x_n}$ ont été factorisés et le terme clos résultant est suffisamment petit pour garantir la terminaison du processus qui consiste à remplacer une inconnue par une partie connue et d'autres inconnues plus "petite". (voir [7] et [6])

Dans le $\overline{\lambda\mu}$ cependant il y a des variables calleR et calleE. La factorisation la plus simple qu'on ait trouvée (c'est vraisemblablement la plus simple possible) a lieu comme suit :

$$s =_R \mu\beta.\langle \lambda z\overline{y_n}.\mu\gamma.\langle z \| s[\overline{x_n} \leftarrow \overline{y_n}, \alpha \leftarrow \gamma] \bullet \delta\rangle \| \lambda x \mu\epsilon.\langle x \| \bullet \beta\rangle \bullet \overline{x_n} \bullet \alpha\rangle$$

Le terme clos résultant est $\lambda z\overline{y_n}.\mu\gamma.\langle z \| s[\overline{x_n} \leftarrow \overline{y_n}, \alpha \leftarrow \gamma] \bullet \delta\rangle$. Dans une tentative de mimétisme de l'unification dans le λ-calcul, on obtient des règles dont certaines font apparaître des inconnues (termes clos) qui ne décroissent pas en taille. Il nous manque alors une propriété de terminaison à un certain endroit.

Reste que l'égalité ci-dessus nous donne une méthode générale pour facoriser autant de variables calleR et calleE hors d'un terme quelconque. Cela servira peut-être un autre jour...

Définition 33 ($L_0(W)$) *Soit W un ensemble quelconque de variables. On définit $L_0(W)$ par induction :*

- $t \in L_0(W)$ *et* $W \subset W' \quad \Longrightarrow \quad t \in L_0(W')$
- $x \in L_0(\{x\})$
- $\alpha \in L_0(\{\alpha\})$
- $r_c, e_c \in L_0(\emptyset)$
- $r \in L_0(W) \quad \Longrightarrow \quad \lambda x.r \in L_0(W - \{x\})$
- $c \in L_0(W) \quad \Longrightarrow \quad \mu\alpha.c \in L_0(W - \{\alpha\})$
- $c \in L_0(W) \quad \Longrightarrow \quad \tilde{\mu}x.c \in L_0(W - \{x\})$
- $r, e \in L_0(W) \quad \Longrightarrow \quad \langle r \| e\rangle \in L_0(W)$
- $X \in L_0(A(X))$ *et* $\forall W \not\supseteq A(X)$ *on a* $X \notin L_0(W)$.

Définition 34 (Compatibilité) *On note $Comp(\theta, A, W)$ pour signifier que θ est compatible avec la fonction d'autorisation A sur le sous-ensemble des inconnues W, i.e :* $\forall X \in W \quad \theta(X) \in L_0(A(X))$.

Lemme 18 *Les trois propositions suivantes sont équivalentes :*

1. $Comp(\theta, A, W)$
2. $Comp(\theta, A \mid_W, W)$
3. $Comp(\theta, A, W \cap D(\theta))$

Lemme 19 *Soit A une fonction d'autorisation, alors pour toute inconnue X et pour toute substitution θ telle que $Comp(\theta, A, \{X\})$ on a $\theta(\iota(X)) = \theta(\iota \mid_{A(X)} (X))$.*

Le lemme précédent nous permet de confondre $\iota(X)$ et $\iota \mid_{A(X)} (X)$.

Définition 35 *On enrichit la congruence structurelle (notée "=" par abus) de la règle $\iota(X) = \iota \mid_{A(X)} (X)$.*

Lemme 20 *Soit t un terme. Si $Comp(\theta, A, UK(t))$ alors $t \in L_0(W) \quad \Longrightarrow \quad \theta(t) \in L_0(W)$.*

Lemme 21 *Soit t un terme. $t \in L_0(A) \quad \Longrightarrow \quad \eta(t) \in L_0(A)$.*

Preuve : Les mêmes variables libres et inconnues apparaissent dans les deux termes alors que les variables introduite par $\eta(t)$ sont fraîches. □

Définition 36 (Solutions d'un système d'équations) *Une substitution θ est dite solution d'un système d'équations A, S si :*

1. *θ unifie S, i.e $\forall \{u \stackrel{?}{=} v\} \in S$, $\theta(u) \stackrel{*}{\leftrightarrow}_R \theta(v)$ et*
2. *$Comp(\theta, A, UK(S))$.*

On note $U(A, S)$ l'ensemble des substitutions solutions de A, S n'introduisant pas de $\tilde{\mu}$ et $U_t(A, S)$ l'ensemble des solutions. Dans le cas du $\overline{\lambda\mu}$-calcul, $U_t(A, S) = U(A, S)$.

Définition 37 *Soit A, S un système d'équations et W un ensemble d'inconnues. On dit que U est un ensemble complet d'unificateurs de A, S hors de W (on note "U est $CompSU(A, S)[W]$") si :*

1. *$\forall \sigma \in U, \quad D(\sigma) \subset UK(S)$ et $\iota(\sigma) \cap (W \cup D(\sigma)) = \emptyset$,*
2. *$U \subset U_t(A, S)$,*
3. *$\forall \theta \in U_t(A, S), \quad \exists \sigma \in U$ tel que $\sigma \leq_R \theta[UK(S)]$*

Notation 8 *Soit f une transformation qui s'applique à la fois à des capsules, caller et calleE (par exemple une susbtitution, η ou $\downarrow_R$). On note $f(S)$ le système obtenu à partir de S et en appliquant f aux deux membres de chacune des ses équations. De la même manière $FV, UK(S) = \cup_{\{t \stackrel{?}{=} t'\} \in S} FV, UK(t, t')$.*

Définition 38 (Formes résolues) *Soit A, S un système d'équations :*

- *On dit que l'inconnue X est résolue dans A, S si $X \in UK(S)$ et si X n'apparaît qu'une fois dans S sous la forme $\{X \stackrel{?}{=} t\}$, avec $t \in L_0(A(X))$.*
- *On dit qu'une équation $\{u \stackrel{?}{=} v\}$ est résolue dans A, S si u ou v est une inconnue résolue dans A, S.*
- *Un système A, S est dit résolu si toutes ses équations sont résolues dans A, S.*

Définition 39 *Soit A, S un système d'équations. on note $UK^*(S_A)$ les inconnues non résolues de A, S et S^*_A l'ensemble des équations non résolues de A, S*

Définition 40 *Soit $S = \{\{X_1 = t_1\} \ldots \{X_n = t_n\}\}$ un système d'équations. On note σ_S la substitution $[t_1/X_1, \ldots t_n/X_n]$.*

Lemme 22 *Soit A, S un système résolu. On a :*

1. *$\sigma_S \in U_t(A, S)$ et*
2. *$\forall \theta \in U_t(A, S), \quad \sigma_S \leq_R \theta$.*

4 Restrictions

4.1 Restrictions quant aux solutions recherchées

Lemme 23 *Soit A, S un système d'équations, $\theta \in U_t(A,S)$ et W un ensemble d'inconnues qu'on dit protégées. Alors il existe une substitution σ telle que :*

1. *σ $fn\eta l$,*
2. *$D(\sigma) \subset UK(S)$ et $\iota(\sigma) \cap (W \cup D(\sigma)) = \emptyset$.*
3. *$\sigma \in U_t(A,S)$.*
4. *$\sigma \leq_R \theta[UK(S)]$ et $\theta \leq_R \sigma[UK(S)]$.*

Définition 41 *On appelle solution standard d'un système A, S toute solution vérifiant les deux premières propriétés du lemme 23 . On note $U_{ts}(S)$ l'ensemble de ces solutions. $U_s(A,S)$ les solutions standard n'introduisant pas de $\tilde{\mu}$.*

Ainsi, quand on est confronté à un système d'équations A, S, on ne recherche plus que les solutions dans $U_{ts}(A,S)$. On retrouve les autres solutions par renommage et R-conversion.

4.2 Restrictions quant au système étudié

Lemme 24 *[Dans $\overline{\lambda\mu\tilde{\mu}}$ et dans $\overline{\lambda\mu}$] Soient S et S' deux multi-ensembles d'équations. Si $S =_R S'$ alors $U(A,S) = U(A,S')$ $\forall A$ tel que A, S soit un système d'équation.*

Le lemme ci-dessus nous permet de manipuler un système à notre guise à R-convertibilité près.

Lemme 25 $U_t(A,S) = U_t(A\mid_{UK(S}, S)$.

Le lemme ci-dessus nous dit qu'une fonction d'autorisation nous parlant d'inconnues n'existant pas ne sert à rien.

4.3 Pré-traitemement d'un système du $\overline{\lambda\mu\tilde{\mu}}$-calcul

On présente ci-dessous trois transformations qui, appliquées successivement à un système d'équations du $\overline{\lambda\mu\tilde{\mu}}$-calcul, le réduise à la résolution d'un système du $\overline{\lambda\mu}$-calcul.

Définition 42 *Soit A, S un système d'équations du $\overline{\lambda\mu\tilde{\mu}}$-calcul. On note $UK_e(S) = \{E_1 \dots E_n\}$. Soit P_n l'ensemble des parties de $\{1 \dots n\}$. Soit $J \in P_n$. On définit $A_J = A \cup \{R_j \mapsto A(E_j)\}_{j \in J}$ et $S_J = \{E_j \stackrel{?}{=} \tilde{\mu}x_j.\langle R_j \| x_j \bullet e_0\rangle\} \cup S[E_j \leftarrow \tilde{\mu}x_j.\langle R_j \| x_j \bullet e_0\rangle]$, où e_0 est un calleE constant de type de base. Notons que A_J est à valeur dans $P(V_e)$: A_J, S_J est donc bien un système d'équations du $\overline{\lambda\mu\tilde{\mu}}$-calcul. Soit la transformation suivante (Engagement tilde partiel pour une inconnue calleE) :*

$$(\widetilde{PEC}) \quad A, S \quad \Longrightarrow \quad A_J, S_J$$

Lemme 26 *La transformation* $\widetilde{PEC}$ *est correcte, i.e* $U_s(A_J, S_J) \subset U_{ts}(A, S)$.

Lemme 27 *La transformation* $\widetilde{PEC}$ *est complète, i.e* $\forall \rho \in U_{ts}(A, S) \quad \exists J \in P_n$ *et* $\exists \rho' \in U_s(A_J, S_J)$ *tel que* $\rho = \rho' \mid_{UK(S)}$.

Les deux lemmes précédents nous disent qu'on peut transformer la recherche de solutions d'un système en la recherche de solutions n'introduisant pas de $\tilde{\mu}$ de plusieurs systèmes.

Soit la transformation suivante :

$$(\tilde{\eta}) \quad A, S \quad \Longrightarrow \quad A, \eta(S)$$

Lemme 28 *La transformation* $(\tilde{\eta})$ *est correcte et complète :* $U_s(A, S) = U_s(A, \eta(S))$.

Preuve : D'après le lemme 24. □

Maintenant, savoir trouver les solution sans $\tilde{\mu}$ d'un système $fn\eta l$ nous permet de résoudre le problème général.

On définit la règle de transformation suivante :

$$\widetilde{ED} \quad A, \{\tilde{\mu}x.c \stackrel{?}{=} \tilde{\mu}x.c'\} \cup S \quad \Longrightarrow \quad A, \{c \stackrel{?}{=} c'\} \cup S$$

Lemme 29 *La règle* $\widetilde{ED}$ *induit un système de récriture terminant et confluent et la forme normale d'un système* $fn\eta l$ *selon ce système de récriture ne contient pas de* $\tilde{\mu}$.

Preuve : Clair sachant que si un $\tilde{\mu}$ apparaît dans un terme $fn\eta l$ alors ce terme est un calleE de la forme $\tilde{\mu}x.c$, sans autre $\tilde{\mu}$ à l'intérieur. Cf lemme 9. □

Lemme 30 *La règle* $\widetilde{ED}$ *est correcte et complète : si* $A, S \quad \Longrightarrow_{\widetilde{ED}} A', S' \quad$ *alors* $U_s(A, S) = U_s(A', S')$.

Preuve : Clair. □

On obtient donc finalement un système sans $\tilde{\mu}$ et on en recherche des solutions sans $\tilde{\mu}$. Ainsi après ces trois transformation, résoudre un problème équationnel du $\overline{\lambda\mu\tilde{\mu}}$ revient donc à résoudre un problème équationnel du $\overline{\lambda\mu}$ (d'un genre particulier puisque le A de départ est à valeur dans $P(V_e)$).

5 Résolution dans $\overline{\lambda\mu}$

5.1 Résolution générales avec le système GET

On propose d'abord un système de règles qui est loin d'être optimal en terme de recherche de solutions mais qui est relativement simple dans sa lecture. Il est correct et il "trouve" $U_s(A, S)$. Cela signifie la complétude .

On donne d'abord les acronymes des règles et à quelles opérations elles correspondent, puis les règles brutes, et enfin leur conditions d'application :

UE Elimination d'inconnue.
EE Elimination d'équation.
EA Addition d'équation.

$$\begin{array}{lrcl} UE & A, \{X \stackrel{?}{=} t\} \cup S & \Rightarrow & A, \{X \stackrel{?}{=} t\} \cup S[X \leftarrow t] \\ EE & A \cup A', \{t \stackrel{?}{=} t'\} \cup S & \Rightarrow & A, S \\ EA & A, S & \Rightarrow & A \cup A', \{t \stackrel{?}{=} t'\} \cup S \end{array}$$

UE Pas de condition.
EE $t =_R t'$ et $support(A') = support(A \cup A') - UK(S)$.
EA $support(A') \subset UK(t, t') - UK(S)$.
On dprésente maintenant la correction et la complétude de la transformation.

Lemme 31 (Correction) *Le système de récriture GET est correct, i.e si* $A, S \Rightarrow_{GET} A', S'$ *et* $\theta \in U(A', S')$ *alors* $\theta \in U(A, S)$

Lemme 32 (Complétude) *Soient* A, S *un système d'équation et* $\theta \in U_s(A, S)$*, alors il existe une série de transformations* $A, S \stackrel{*}{\Longrightarrow}_{GET} A, S'$ *tel que* A, S' *soit résolu et* $\sigma_{S'} = \theta$*.*

Cependant deux problèmes apparaissent : une recherche utilisant ces règles serait de profondeur et de largeur infinie, i.e à chaque étape on peut avoir un nombre infini de choix et il peut exister des réductions de nombre infini d'étapes.

5.2 Pré-unification

Le branchement infini étant un problème, on cherche à pré-unifier. Pré-unifier consiste à transformer le système de manière raisonnable (correcte, complète) jusqu'à arriver à une forme dont on sait qu'elle a des solutions.

5.2.1 Définition et intérêt

Définition 43 (Flexible-flexible) *Une équation est dite flexible-flexible si elle est de l'une des deux formes suivantes :*

- $\iota(E) \stackrel{?}{=} \iota'(E')$ *(*E *et* E' *sont des méta-variables d'inconnue), ou*
- $\langle \iota(R) \| ... \rangle \stackrel{?}{=} \langle \iota'(R') \| ... \rangle$ *(*R *et* R' *sont des méta-variables d'inconnue).*

Définition 44 (Pré-solution) *Une substitution* θ *est dite pré-solution d'un système d'équations* A, S *si*

- $\forall \{u \stackrel{?}{=} v\} \in S \quad \theta(u) =_R \theta(v)$ *ou* $\{\theta(u) \downarrow_R \stackrel{?}{=} \theta(v) \downarrow_R\}$ *est flexible-flexible.*
- $Comp(\theta, A, UK(S))$

On note $PU(A,S)$ l'ensemble des pré-solutions de A,S.

Lemme 33 *Soit A,S un système d'équations, $\theta \in PU(A,S)$ et W un ensemble d'inconnues qu'on dit protégées. Alors il existe une substitution σ telle que :*

1. *σ $fn\eta l$,*
2. *$D(\sigma) \subset UK(S)$ et $\iota(\sigma) \cap (W \cup D(\sigma)) = \emptyset$.*
3. *$\sigma \in PU(A,S)$.*
4. *$\sigma \leq_R \theta[UK(S)]$ et $\theta \leq_R \sigma[UK(S)]$.*

Définition 45 *On appelle pré-solution standard d'un système A,S toute pré-solution vérifiant les deux premières propriétés du lemme 33 . On note $PU_s(S)$ l'ensemble de ces solutions.*

Le lemme 33 nous dit qu'il suffit de connaître $PU_s(A,S)$ pour connaître PU_s. Dorénavant on ne recherche plus que les pré-solutions standard.

Définition 46 (Forme pré-résolue) *Une équation d'un système A,S est dite pré-résolue dans A,S si elle est résolue dans A,S ou flexible-flexible. Un système d'équation A,S est dit pré-résolu si toutes ses équations sont pré-résolues dans A,S.*

Lemme 34 *Tout système pré-résolu est unifiable et a une infinité de pré-solutions standard.*

Définition 47 *Soit $S = \{X_i \stackrel{?}{=} t_i\} \cup \{\iota_j(E_j) = \iota'_j(E'_j)\} \cup \{\langle \iota_k(R_k) \| e_k \rangle \stackrel{?}{=} \langle \iota'_k(R'_k) \| e'_k \rangle\}$ un multi-ensemble d'équations. On appelle σ_S la substitution $[X_i \mapsto t_i]$.*

Lemme 35 *Soit A,S un système pré-résolu et $\theta \in PU(A,S)$ alors $\sigma_S \leq_R \theta$.*

Preuve : Comme pour le cas résolu en à peine plus compliqué. □

5.2.2 Transformations d'un système d'équations du $\overline{\lambda\mu}$ (BET)

Sans perte de généralité (d'après les lemmes 24 et 25 et du bon sens) on se restreint à l'étude des systèmes bien formés définis ci-dessous :

Définition 48 *On dit d'un système d'équations A,S qu'il est bien formé s'il vérifie les trois conditions suivantes :*

1. *$support(A) \subset UK(S)$.*
2. *Les membres des équations sont soit $fn\eta l$ soit des inconnues calleR résolues.*
3. *Les membres des équations sont deux à deux compatibles en type.*

Définition 49 *Soit A,S un système. On appelle η'_{S_A} l'application des termes vers les termes qui est l'identité pour les inconnues calleR résolues dans S_A et η pour le reste. S'il n'y a pas ambiguité on notera simplement η'.*

Notation 9 *On appelle :*
- $ct_r(r_a) = \lambda\overline{x_n}\mu\alpha.\langle r_a \| \overline{R_m} \bullet E\rangle$.
- $ct_e(e_a) = \lambda\overline{x_n}\mu\alpha.\langle R' \| \ e_a\rangle$

Les ct ne sont pas des fonctions, mais juste une commodité d'écriture.

On donne d'abord les noms des règles du système BET, puis leur contenu, et enfin leurs conditions d'application.

EAE	Elimination d'équation atomique.
CD	Décomposition de capsule.
ED	Décomposition de calleE.
RD	Décomposition de calleR.
EAC	Engagement atomique pour un calleE.
PEC	Engagement partiel pour un calleE.
PRC	Engagement partiel pour un calleR.

$$EAE$$
$$A, \{t_a \stackrel{?}{=} t_a\} \cup S$$
$$\Downarrow$$
$$A\mid_{UK(S)}, S$$

$$CD_{ra}$$
$$A, \{\langle r_a \| e\rangle \stackrel{?}{=} \langle r_a \| e'\rangle\} \cup S$$
$$\Downarrow$$
$$A, \{e \stackrel{?}{=} e'\} \cup S$$

$$CD_{ea}$$
$$A, \{\langle \lambda\overline{x_n}.\mu\alpha.c \| e_a\rangle \stackrel{?}{=} \langle \lambda\overline{x_n}.\mu\alpha.c' \| e'_a\rangle\} \cup S$$
$$\Downarrow$$
$$A, \{r \stackrel{?}{=} r', e_a \stackrel{?}{=} e'_a\} \cup S$$

$$ED$$
$$A, \{r \bullet e \stackrel{?}{=} r' \bullet e'\} \cup S$$
$$\Downarrow$$
$$A, \{r \stackrel{?}{=} r', e \stackrel{?}{=} e'\} \cup S$$

$$RD$$
$$A, \{\lambda\overline{x_n}.\mu\alpha.c \stackrel{?}{=} \lambda\overline{x_n}.\mu\alpha.c'\} \cup S$$
$$\Downarrow$$
$$A, \{c \stackrel{?}{=} c'\} \cup S$$

$$
\begin{array}{c}
EAC \\
A, \{\iota(E) \stackrel{?}{=} e_a\} \cup S \\
\Downarrow \\
A, \{E \stackrel{?}{=} e'_a\} \cup S[E \leftarrow e'_a]
\end{array}
$$

$$
\begin{array}{c}
PEC \\
A, S \\
\Downarrow \\
A \cup \{R' \mapsto A(R), E' \mapsto A(R)\}, \\
\{E \stackrel{?}{=} \eta(R') \bullet E'\} \cup \eta'(S[E \leftarrow \eta(R') \bullet E'])
\end{array}
$$

$$
\begin{array}{c}
PRC_{ri} \\
A, \{\langle \iota(R) \| e \rangle \stackrel{?}{=} \langle r_c \| e' \rangle\} \cup S \\
\Downarrow \\
A \cup \{\overline{R_m}, E \mapsto A(R) \cup \{\overline{x_n}, \alpha\}\}, \\
\{R \stackrel{?}{=} ct_r(r_c)\} \cup \eta'(\{\langle \iota(ct_r(r_c)) \| e \rangle \stackrel{?}{=} \langle r_c \| e' \rangle\} \cup S[R \leftarrow ct_r(r_c)])
\end{array}
$$

$$
\begin{array}{c}
PRC_{rp} \\
A, \{\langle \iota(R) \| e \rangle \stackrel{?}{=} c\} \cup S \\
\Downarrow \\
A \cup \{\overline{R_m}, E \mapsto A(R) \cup \{\overline{x_n}, \alpha\}\}, \\
\{R \stackrel{?}{=} ct_r(y)\} \cup \eta'(\{\langle \iota(ct_r(y)) \| e \rangle \stackrel{?}{=} c\} \cup S)[R \leftarrow ct_r(y)]
\end{array}
$$

$$
\begin{array}{c}
PRC_{ei} \\
A, \{\langle \iota(R) \| e \rangle = \{\langle r \| e_c \rangle\} \cup S \\
\Downarrow \\
A \cup \{R' \mapsto A(R) \cup \{\overline{x_n}, \alpha\}\}, \\
\{R \stackrel{?}{=} ct_e(e_c)\} \cup \eta'(\{\langle \iota(ct_e(e_c)) \| e \rangle \stackrel{?}{=} \langle r \| e_c \rangle\} \cup S[R \leftarrow ct_e(e_c)])
\end{array}
$$

$$
\begin{array}{c}
PRC_{ep} \\
A, \{\langle \iota(R) \| e \rangle \stackrel{?}{=} c\} \cup S \\
\Downarrow \\
A \cup \{R' \mapsto A(R) \cup \{\overline{x_n}, \alpha\}\}, \\
\{R \stackrel{?}{=} ct_e(\beta)\} \cup \eta'(\{\langle \iota(ct_e(e_a)) \| e \rangle \stackrel{?}{=} c\} \cup S[R \leftarrow ct_e(\beta)])
\end{array}
$$

EAE Pas de condition.
CD r_a n'est pas une inconnue.
ED Pas de condition.
RD Pas de condition.
EAC E non résolues dans $A, \{\iota(E) = e_a\} \cup S$, $e'_a \in \hat{\iota}^{-1}(\{e_a\}) \cap (A_e(E) \cup K)$.
PEC E non résolue dans $A, \{\iota(E) = e_a\} \cup S$. Types compatibles.
PRC $y, \beta \in A(R')$.

Pour les quatre PRC, le i et le p signifie respectivement "imitation" et "projection" comme cela était le cas dans le λ-calcul. Mais ici il y a deux fois plus de règles...

Lemme 36 *Le système de récriture BET conserve la bonne forme.*

Preuve : On le fait règle par règle.
- EAE : On restreint A à $A\mid_{UK(S)}$ et pour le reste rien ne change...
- CD_ra :
- IDEM

□

Lemme 37 *Le système de récriture BET est correct. Si $A, S \Rightarrow_{BET} A', S'$ et $\theta \in PU_s(A', S')$ alors $\theta \in PU_s(A, S)$.*

Preuve : On le fait règle par règle. □

Pour prouver la complétude de BET, on a besoin de définir un système SET de transformation de paires substitution-équations. Les transformations sont donc de cette forme : $\theta, A, S \quad \Rightarrow_{SET} \quad \theta', A', S'$ avec la contrainte $A, S \quad \Rightarrow_{BET} \quad A', S'$. On donne ci-dessous les règles dans lesquelles on ne fait apparaître que la substitution :

$$EAE, CD, ED, RD, EAC \quad \theta \quad \Rightarrow \quad \theta$$

$$PEC \quad [E \mapsto r \bullet e] \cup \theta \quad \Rightarrow \quad [E \mapsto r \bullet e] \cup [R' \mapsto r, E' \mapsto e] \cup \theta$$

$$\frac{PRC_{ri}, PRC_{rp}}{\theta \cup [R \mapsto \lambda\overline{x_n}\mu\alpha.\langle r_a \text{ ou } y \| \overline{s_m} \bullet e\rangle] \Downarrow \theta \cup [R \mapsto \lambda\overline{x_n}\mu\alpha.\langle r_a \text{ ou } y \| \overline{s_m} \bullet e\rangle] \cup [\overline{R_m} \mapsto \overline{s_m}, E \mapsto e]}$$

$$\frac{PRC_{ei}, PRC_{ep}}{\theta \cup [R \mapsto \mu\alpha.\langle r \| e_a \text{ ou } \beta\rangle] \Downarrow \theta \cup [R \mapsto \mu\alpha.\langle r \| e_a \text{ ou } \beta\rangle] \cup [R' \mapsto r]}$$

Définition 50 – *On appelle $|t|$ le nombre de sous-termes d'un terme t. Soit $S_A = A, S$ un système d'équations. On définit plusieurs mesures sur une paire ρ, S_A (on rappelle que placer une étoile $*$ en exposant signifie qu'on en considère que ce qui n'est pas résolu et que $\{\{.\}\}$ signifie multi-ensemble) :*

- $m_1(\rho, S_A) = \{\{|\rho(X)|, X \in UK^*(S_A)\}\}$
- $m_2(\rho, S_A) = \Sigma_{\{u \stackrel{?}{=} v\} \in S_A} |u| + |v|$.

Lemme 38 *Le système de récriture SET termine.*

Lemme 39 (Complétude) *Soit A, S un système d'équations bien formé et non pré-résolu et $\rho \in PU_s(A, S)$, alors il existe une transformation du système SET $\rho, A, S \Rightarrow_{SET} \rho', A', S'$ telle que :*

1. *$\rho' = \theta[UK(S)]$ et*
2. *$\rho' \in PU(A', S')$.*

Lemme 40 *Soit A, S un système d'équations. Le nombre de transformations BET applicables à ce système est fini.*

> **Preuve :** On a un nombre fini de classes de transformation et chaque classe a un nombre fini d'instances (car toutes les fonctions manipulées sont à support fini) pour un système A, S donné. Or chaque instance ne peut s'appliquer qu'à un nombre fini d'endroits dans chacune des équations et les équations sont en nombre fini. □

Théorème 1 *[Complétude] Soit A, S un système d'équations et $\theta \in PU_s(A, S)$. Alors il existe $A, S = A_1, S_1 \Rightarrow A_2, S_2 \Rightarrow \ldots \Rightarrow A_n, S_n$ une suite finie de transformations de BET telle que A_n, S_n soit pré-résolu, $\sigma_{S_n} \in PU_s(A, S)$ et $\sigma_{S_n} \leq_R \theta[UK(S)]$.*

Théorème 2 (Décidabilité) *L'existence de solution d'un système d'équation est semi-décidable.*

> **Preuve :** Il suffit d'effectuer un parcours en largeur de l'arbre dont les noeuds sont les systèmes (le système initial à la racine) et dont les arêtes (orientées vers les fils) sont les transformations de BET. Chaque noeud a un nombre fini de fils d'après le lemme 40, donc s'il existe des solutions une d'entre elle apparaîtra d'après le lemme 1. □

6 Conclusion

En regard de l'importance de l'unification pour la manipulation des programmes d'aujourd'hui, nous nous sommes intéressés à l'unification dans les programmes de demain. Nous avons défini un calcul confluent où on peut donc parler d'égalité, et néanmoins très général en ce sens qu'il interpète la logique classique et qu'il contient d'autres sous-calculs eux mêmes intéressants (dont le $\lambda\mu$ de Parigot). Ces calculs étendent

d'ailleurs la correspondance de Curry-Howard qui ne disait jusqu'à présent que "preuve=programme" dans un cadre intuitioniste. La question de l'extensionalité dans ce calcul a mené à des règles qui pourraient éclairer l'extensionalité dans les calculs déjà existants. Des traductions de certains calculs dans d'autres ont commencé à faire apparaître les liens qui existent entre ces différents formalismes.

En ce qui concerne l'objectif initial du stage, nous avons donné des règles de recherche de solution qui sont correctes et complètes (l'algoritme dit la vérité, toute la vérité rien que la vérité). Cela a mené à un théorème de semi-décidabilité, comme dans le λ-calcul. On ne peut pas vraiment espérer plus car l'unifiabilité est indécidable dès l'ordre 2 (or ici on est en ordre arbitraire). Nous sommes arrivés à ce résultat en nous inspirant du travail de Gérard Huet mais, les embûches étant nombreuses, nous avons parfois dû imaginer des solutions différentes. On citera notamment l'introduction d'un fonction d'autorisation qui nous permet de remplacer les inconnues par des termes non nécesairement clos, i.e certaines variables libres peuvent y apparaître. Sans cette fonction d'autorisation, on arrive presque au résultat mais on échoue quant à démontrer la semi-décidabilité de l'existence de solutions...

Tout en répondant à la question posée par le sujet de stage, le rapport suggère donc de nombreuses pistes de recherche connexes au sujet traité.

7 Perspectives

Pour arriver à notre résultat, nous avons fait des choix mais, parfois, d'autres solutions étaient envisageables. Nous en mentionnons certaines ci-dessous. Nous avons également ouvert des pistes de réflexion que seul le temps nous a contraint à mettre de côté provisoirement.

Les règles de réduction du $\overline{\lambda}\mu\tilde{\mu}$-calcul correspondent à des modifications de preuve dans le calcul des séquents (Curry-Howard généralisé). Le $\overline{\lambda\mu\tilde{\mu}}$-calcul introduit de nouvelles règles d'extensionalité. On devra donc étudier en détail l'influence de ces règles sur l'arbre de typage des termes correspondants. Elles modifient l'arbre de typage qui est un arbre de preuve et en général cela se traduit par une simplification de la preuve.

On s'attachera également à comprendre dans quelle mesure toutes ou partie de ces règles sont également valables dans le $\overline{\lambda}\mu\tilde{\mu}$-calcul.

Il faudra prouver la terminaison et la confluence quand on n'a fait que les supposer.

Je conjecture que pour la définition que j'ai donnée de l'extensionalité , les règles établies capturent cette notion. Il faudra réfléchir plus en profondeur à l'extensionalité et valider ainsi la définition actuelle.

Il sera bon de vérifier dans quelle mesure les règles du $\lambda\mu$-calcul et du $\overline{\lambda\mu}$-calcul (notamment les règles d'extensionalité) correspondent par l'intermédiaires de la traduction de [1]. Cela peut permettre de trouver de nouvelle règle par analogie ou de mieux comprendre certaines règles existantes (notamment la dernière du $\lambda\mu$!).

La question se pose de savoir si le $\overline{\lambda\mu}$-calcul peut être plongé dans le $\lambda\mu$-calcul ainsi que celle de savoir si le $\lambda\mu$-calcul est peut être plongé dans tout calcul interprétant la logique classique.

D'après la grammaire du $\overline{\lambda\mu\tilde{\mu}}$, on a choisi de considérer $\iota(E)$ comme un e_v, i.e un fond de pile simple, sans $\tilde{\mu}$. Or une substitution ultérieure pourra remplacer $\iota(E)$ par un $\tilde{\mu}x.c$ et ainsi faire passer $\iota(E)$ dans la sorte $\tilde{e}$. Cette incertitude associée à $\iota(E)$ fait qu'il ne correspond réellement ni à l'un ni à l'autre des fonds de pile. On pourra se demander ce qu'il se passe quand on considère que $\iota(E)$ est un $\tilde{e}$.

Je conjecture que dans le $\overline{\lambda\mu}$, si on se donne t un terme $fn\eta l$ et θ une substitution normalisée. Alors $\theta(t)\downarrow_{\lambda\mu}$ est $fn\eta l$.

On a défini le $\overline{\mu\tilde{\mu}}$-calcul . Il ne possède pas de règle d'extensionalité hormis $(\eta_{\tilde{\mu}})$. La résolution des équations dans ce calcul est donc facilitée par le plongement dans le $\overline{\lambda\mu\tilde{\mu}}$-calcul : on résoud l'équation dans $\overline{\lambda\mu\tilde{\mu}}$ et on ne conserve (au plus) que les solutions dont la forme R-normale (dans $\overline{\lambda\mu\tilde{\mu}}$) est un $\overline{\lambda\tilde{\mu}}$-terme. C'est le même principe que la résolution des équations du second degré avec les nombres complexes. Il faudra étudier cela plus en détail.

Références

[1] Pierre-Louis Curien and Hugo Herbelin. The duality of computation. In *Proceedings of the 5 th ACM SIGPLAN International Conference on Functional Programming (ICFP'00)*, pages 233–243. ACM, 2000.

[2] S. Ghilezan and P. Lescanne. Classical proofs and typed processes : intersection types and strong normalization. In *TYPES 03 workshop*, volume 3085 of *Lecture Notes in Computer Science*. Springer-Verlag, 2004.

[3] G. Huet. A unification algorithm for typed lambda calculus. *Theoretical Computer Science*, 1(1) :27–57, 1975.

[4] G. Huet. *Résolution d'équations dans les langages d'ordre 1,2, ...,ω*. Thèse de Doctorat d'Etat, Université de Paris 7 (France), 1976.

[5] M. Parigot. An algorithmic interpretation of classical natural deduction. In *Proc. of Int. Conf. on Logic Programming and Automated Reasoning, LPAR'92*, volume 624 of *Lecture Notes in Computer Science*, pages 190–201. Springer-Verlag, 1992.

[6] C. Prehofer. *Solving Higher-Order Equations From Logic to Programming*. Progress in Theoretical Computer Science. Birkhäuser, 1997.

[7] Wayne Snyder. *A proof theory for general unification*, volume 11 of *Progress in computer science and applied logic*. Birkhäuser, 1991.

A Les métaphores des calculs

A.1 μ et $\tilde{\mu}$ dans le $\overline{\lambda}\mu\tilde{\mu}$

Un calleR commençant par μ est un programme qui n'a pas la patience de prendre les arguments un par un sur la pile : il avale d'un coup la pile entière ainsi que le fond de pile. Une variable calleE est un fond de pile simple sur lequel on pourra ultérieurement replacer des calleR. Un calleE commençant par $\tilde{\mu}$ est un fond de pile carnivore qui

attend sagement qu'un calleR dépile la pile jusqu'à tomber sur le $\tilde{\mu}$. Quand un calleR impatient et un calleE carnivore sont en vis-à-vis, on ne sait pas lequel des deux mange l'autre : il y a non-déterminisme.

A.2 μ et $\tilde{\mu}$ dans le $\overline{\lambda\mu\tilde{\mu}}$

Il est temps de donner une interprétation aux règles : un calleR commençant par λ est un cannibale du premier ordre car il mange des calleR qu'il a acheté au distributeur (le calleE). Un calleR commençant par μ est un cannibale du deuxième ordre qui gobe les distributeurs et les calleR s'y trouvant (sans doute a-t-il très faim mais pas de monnaie à insérer dans le distributeur). un calleE avec un $\tilde{\mu}$ au fond est un distributeur carnivore (mais patient) qui une fois vidé dévore le client (le calleR) qui vient d'acheter un calleR à manger (on appelle se genre de distributeur un "calleE piégé"). Notons que les cannibales du deuxième ordre, bien qu'affamés, ne mange pas les distributeurs carnivores qui sont sans doute un peu trop gros.

A.3 Première nouvelle règle extensionelle

On interprète : $s \bullet \tilde{\mu}x.c$ est un distributeur carnivore qui veut vendre un calleR s puis manger le client. $\tilde{\mu}x'.c[x \leftarrow \mu\alpha.\langle x' \| s \bullet \alpha\rangle]$ est distributeur carnivore qui veut manger directement le client puis, dans son estomac, lui donner un calleR s. Cela ne revient-il pas au même ? Notons que le client en question doit être un cannibal du premier ordre, i.e pas un mangeur de distributeur, car sinon rien ne se passe a priori avec le calleE $s \bullet \tilde{\mu}x.c$.

A.4 Deuxième nouvelle règle extensionelle

On interpète : $\mu\alpha'.c$ est un simple mangeur de distributeur. $\lambda x.\mu\alpha.c[\alpha' \leftarrow x \bullet \alpha]$ est un cannibal du pemier ordre qui veut d'abord manger le calleR placé sur le haut de la pile du distributeur. Une fois fait, ce client a encore faim mais plus de monnaie : poussé à la folie, il se transforme en cannibal du second ordre et avale le distributeur. Dans son estomac il replace alors le calleR qu'il avait mangé quelques instants avant et le replace sur le haut de la pile du distributeur. La différence n'est pas grande avec $\mu\alpha'.c$ qui mange tout d'un coup.

A.5 Troisième nouvelle règle extensionelle

On interpète : la première situation $\langle \lambda\overline{x_n}\mu\beta c \| \alpha\rangle$ décrit un cannibal en face d'un distributeur vide. Le cannibal veut manger un certain nombre de calleR mais comme sa faim est plus grande que son porte-monnaie, on sait déjà qu'il finira par vouloir avaler le distributeur. Le distributeur ne devra donc pas être rempli avec un fond de pile carnivore. Notons que c'est une entreprise qui s'occupe de fournir les distributeurs. Comme l'estomac de $\lambda\overline{x_n}\mu\beta c$ contient aussi des distributeurs à fond de pile α, ils seront tous réaprovisionnés simultanément. Dans la deuxième situtation, l'entreprise ne fournit que l'unique α du terme et ensuite seulement, le calleR va dépiler petit à petit le

distributeur, puis gober le reste pour recomposer le distributeur à l'intérieur de son estomac. Les deux situations sont bien similaires.

B Les preuves

Preuve : 5 Soit un terme t sous forme R-normale et $\tilde{\mu}x.c$ un sous-terme de t. On rappelle que d'après la grammaire, le sur-terme strict minimal (s'il existe) d'un calleE e est un calleE $r \bullet e$ ou une capsule $\langle r \| e \rangle$. On itère tant qu'on n'atteint pas de capsule ou le terme t entier. Si on atteint une capsule alors cette capsule a pour calleE principal $\overline{r_n} \bullet \tilde{\mu}x.c$ Ce n'est pas une forme R-normale sinon on aurait pu appliquer la règle η_e. Ainsi on doit atteindre t sans traverser de capsule donc $t = \overline{r_n} \bullet \tilde{\mu}x.c$. □

Preuve : 6 La confluence d'un calcul est équivalente à la propriété de Church-Rosser. Ainsi $s \downarrow_R t$. Or $\eta(s) = \eta_b(s \downarrow_R) \downarrow_{\lambda\mu}$ d'où le résultat souhaité. □

Preuve : 7 D'après le lemme 6 , il suffit de le démontrer pour les formes R-normales. Par induction sur la structure des termes.

Initialisation :

- $\eta(e_a) = \eta_b(e_a) \downarrow_{\lambda\mu} = \alpha \downarrow_{\lambda\mu} = e_a \xrightarrow{*}_\eta e_a = e_a \downarrow_R$ et
- $\eta(r_a) = \lambda\overline{x_n}.\mu\alpha.\langle r_a \| \overline{x_n} \bullet \alpha \rangle \xrightarrow{n}_{\eta_r} \mu\alpha'\langle r_a \| \alpha' \rangle \rightarrow_{\eta_\mu} r_a$ donc $\eta(r_a) \xrightarrow{*}_\eta r_a = r_a \downarrow_R$.

Induction :

- $\eta(r \bullet e) = \eta_b(r \bullet e) \downarrow_{\lambda\mu} = (\eta_b(r) \bullet \eta_b(e)) \downarrow_{\lambda\mu} = \eta_b(r) \downarrow_{\lambda\mu} \bullet \eta_b(e) \downarrow_{\lambda\mu} = \eta(r) \bullet \eta(e) \xrightarrow{*}_\eta r \bullet e$.
- $\eta(\tilde{\mu}x.c) = \eta_b(\tilde{\mu}x.c) \downarrow_{\lambda\mu} = \tilde{\mu}x.\eta_b(c) \downarrow_{\lambda\mu} = \tilde{\mu}x.\eta(c) \xrightarrow{*}_\eta \tilde{\mu}x.c$
- $\eta(\lambda x.r) = \lambda x.\eta_b(r) \downarrow_{\lambda\mu} = \lambda x.\eta(r) \xrightarrow{*}_\eta \lambda x.r$.
- $\eta(\mu\beta.c) = \lambda\overline{x_n}.\mu\alpha.\langle \mu\beta.\eta_b(c) \| \overline{x_n} \bullet \alpha \rangle \downarrow_{\lambda\mu} = \lambda\overline{x_n}.\mu\alpha.\langle \mu\beta.\eta_b(c) \downarrow_{\lambda\mu} \| \overline{x_n} \bullet \alpha \rangle = \lambda\overline{x_n}.\mu\alpha.\langle \mu\beta.\eta(c) \| \overline{x_n} \bullet \alpha \rangle \xrightarrow{*}_\eta \lambda\overline{x_n}.\mu\alpha.\langle \mu\beta.c \| \overline{x_n} \bullet \alpha \rangle$.
- $\eta(\langle r \| e \rangle) = \langle \eta_b(r) \| \eta_b(e) \rangle \downarrow_{\lambda\mu} = \langle \eta_b(r) \downarrow_{\lambda\mu} \| \eta_b(e) \downarrow_{\lambda\mu} \rangle \downarrow_{\lambda\mu} = \langle \eta(r) \| \eta(e) \rangle \downarrow_{\lambda\mu} \xrightarrow{*}_\eta \langle r \| e \rangle \downarrow_{\lambda\mu} = \langle r \| e \rangle$. La dernière réduction se fait par hypothèse d'induction.

□

Preuve : 9 D'après le lemme 5 c'est vrai pour les formes R-normales. La construction de η_b et la réduction qui suit n'introduise pas de $\tilde{\mu}$. Or ce $\tilde{\mu}$ qui était en surface n'a pas non plus été consommé donc il se retrouve à la même place dans la forme normale η-longue. □

Preuve : 10

1. Soit un r calleR quelconque. Deux cas se présentent : soit il est de la forme $\lambda\overline{x_p}.x$ soit il est de la forme $\lambda\overline{x_p}.\mu\beta.c$. Dans le premier cas $\eta(r) = \lambda\overline{x_p}.\eta(x) = \lambda\overline{x_p}.\lambda\overline{y_n}.\mu\alpha.\langle x \| \overline{y_n} \bullet \alpha \rangle$ et α de type de base. Dans le deuxième cas, $\eta(r) = \lambda\overline{x_p}.\lambda\overline{y_n}.\mu\alpha.(\langle \mu\beta.\eta(c) \| \overline{y_n} \bullet \alpha \rangle) \downarrow_{\lambda\mu}$ avec α de type

de base. Dans les deux cas on a le résultat voulu. Réciproquement soit un calleR r de cette forme. On vérifie que $\eta(r) = r$ grâce notamment au lemme 8.

2. Soit e un calleE quelconque. Premier cas : il est décomposable en $\overline{r_n} \bullet e_a$, auquel cas $\eta(e) = \overline{\eta(r_n)} \bullet e_a$ qui est bien de la forme proposée. deuxième cas : il s'écrit $\tilde{\mu}x.c$, auquel cas $\eta(e) = \eta(\tilde{\mu}x.c) = \tilde{\mu}x.\eta(c)$. Réciproquement...
3. Soit $c = \langle r \| e \rangle$ une capsule R-normale (sans perte de généralité). Elle est de la forme $\langle r_a \| \overline{r_n} \bullet e_a \rangle$ ou $\langle \lambda\overline{x_n}.\mu\alpha.c' \| e_a \rangle$. Dans le premier cas, $\eta(c) = \langle r_a \| \overline{\eta(r_n)} \bullet e_a \rangle$. Dans le deuxième cas, $\eta(c) = \langle \eta(\lambda\overline{x_n}.\mu\alpha.c') \| e_a \rangle$. d'où le résultat. Et réciproquement...

□

Preuve : 11 Initialisation :

- $\hat{\theta} \circ \hat{\iota}(X) = \widehat{\hat{\theta} \circ \iota} \circ \theta(X) = \widehat{\hat{\theta} \circ \iota} \circ \hat{\theta}(X)$,
- $\hat{\theta} \circ \hat{\iota}(cst) = cst = \widehat{\hat{\theta} \circ \iota} \circ \hat{\theta}(cst)$, pour cst constante calleR ou calleE.
- $\hat{\theta} \circ \hat{\iota}(v) = \hat{\theta}(\hat{\iota}(v)) = \hat{\theta}(\iota(v)) = \hat{\theta} \circ \iota(v) = \widehat{\hat{\theta} \circ \iota}(v) = \widehat{\hat{\theta} \circ \iota} \circ \hat{\theta}(v)$, pour v variable calleR ou calleE.

Induction :

- $\hat{\theta} \circ \hat{\iota}(\lambda x.r) = \hat{\theta}(\lambda x.\hat{\iota}(r)) = \lambda x.\hat{\theta} \circ \hat{\iota}(r) = \lambda x.\widehat{\hat{\theta} \circ \iota} \circ \hat{\theta}(r) = \widehat{\hat{\theta} \circ \iota} \circ \hat{\theta}(\lambda x.r)$,
- Idem pour les autres.

□

Preuve : 12 On le fait par induction sur la structure de u.

- Si u est atomique alors v n'existe pas.
- Si $u = \lambda x.r \to_R v$ alors premier cas : la réduction a lieu dans r, auquel cas $v = \lambda x.r'$ avec $r \to_R r'$. $\theta(\lambda x.r) = \lambda x.\theta(r)$ par définition de l'extension d'une substitution. Or par hypothèse d'induction $\theta(r) \overset{*}{\leftrightarrow}_R \theta(r')$ donc $\theta(\lambda x.r) \overset{*}{\leftrightarrow}_R \lambda x.\theta(r') = \theta(\lambda x.r')$. Deuxième cas : la réduction implique le λx. On a donc affaire à la règle η_r. Donc u est de la forme $u = \lambda x.\mu\alpha.c[\alpha' \leftarrow x \bullet \alpha]$ et $v = \mu\alpha'.c$ avec $x, \alpha \notin FV(c)$. Ainsi $\theta(\lambda x.\mu\alpha.c[\alpha' \leftarrow x \bullet \alpha]) = \lambda x.\mu\alpha.\theta(c[\alpha' \leftarrow x \bullet \alpha])$. Or $\theta(c[\alpha' \leftarrow x \bullet \alpha]) = \theta(c)[\alpha' \leftarrow \theta(x \bullet \alpha)]$ d'après le lemme 11 et $\theta(x \bullet \alpha) = x \bullet \alpha$, donc $\lambda x.\mu\alpha.\theta(c[\alpha' \leftarrow x \bullet \alpha]) = \lambda x.\mu\alpha.\theta(c)[\alpha' \leftarrow x \bullet \alpha] \to_{\eta_r} \mu\alpha'.\theta(c) = \theta(\mu\alpha'.c)$.
- Si $u = \mu\alpha.c \to_R v$ alors premier cas : la réduction a lieu dans c, auquel cas $v = \mu\alpha.c'$ et $c \to_R c'$. Les hypothèses d'induction font le reste. Deuxième cas $u = \mu\alpha.\langle r \| \alpha \rangle$ et $\alpha \notin FV(r)$, auquel cas $v = r$. $\theta(u) = \theta(\mu\alpha.\langle r \| \alpha \rangle) = \mu\alpha.\langle \theta(r) \| \alpha \rangle$. Si $\alpha \notin FV(\theta(r))$, auquel cas $\mu\alpha.\langle \theta(r) \| \alpha \rangle \to_{\eta_\mu} \theta(r)$. Sinon, $\theta(r)$ de la forme $\lambda\overline{x_n}\mu\beta.c$ car c'est un calleR non atomique (si $\theta(r)$ est atomique alors c'est α car $\alpha \in FV(r)$, ce qui est impossible). On peut donc appliquer la règle η_c. $\mu\alpha.\langle \theta(r) \| \alpha \rangle \to_{eta_c} \mu\alpha.\langle \theta(r)[\alpha \leftarrow \overline{x_n} \bullet \beta] \| \alpha \rangle$. Or $\theta(r)[\alpha \leftarrow \overline{x_n} \bullet \beta] =$

$\theta(r)[\alpha \leftarrow \theta(\overline{x_n} \bullet \beta)] = \theta(r[\alpha \leftarrow \overline{x_n} \bullet \beta])$ d'après le lemme 11. De plus $r[\alpha \leftarrow \overline{x_n} \bullet \beta] = r$ car par hypothèse $\alpha \notin FV(r)$, d'où le résultat.

- Si $u = r \bullet e$ alors premier cas : la réduction a lieu dans e, auquel cas $v = r \bullet e'$, $e \rightarrow_R e'$ et $\theta(r \bullet e) = \theta(r) \bullet \theta(e) \rightarrow \theta(r) \bullet \theta(e') = \theta(r \bullet e')$. Deuxiéme cas : la réduction a lieu dans r, auquel cas on a aussi le résultat souhaité.
- Si $u = \langle \lambda x.r \| s \bullet e \rangle$ alors premier cas : la réduction a lieu dans r, s ou e, auquel cas les hypothèses d'induction permettent de conclure. Deuxième cas : $v = \langle r[x \leftarrow s] \| e \rangle$. Alors $theta(\langle \lambda x.r \| s \bullet e \rangle) = \langle \lambda x.\theta(r) \| \theta(s) \bullet \theta(e) \rangle \rightarrow_\lambda \langle \theta(r)[x \leftarrow \theta(s)] \| \theta(e) \rangle$. Or $\theta(r)[x \leftarrow \theta(s)] = \theta(r[x \leftarrow s])$ d'après le lemme 11, ce qui permet de conclure.
- Si $u = \langle \lambda \overline{x_n} \mu\alpha.c \| \alpha \rangle$ et $\alpha \in FV(c)$ alors premier cas : la réduction a lieu dans c et les hypothèses d'induction permettent de conclure. Deuxième cas $v = \langle \lambda \overline{x_n} \mu\beta.c[\alpha \leftarrow \overline{x_n} \bullet \beta] \| \alpha \rangle$. De plus, $\theta(\langle \lambda \overline{x_n} \mu\alpha.c \| \alpha \rangle) = \langle \lambda \overline{x_n} \mu\alpha.\theta(c) \| \alpha \rangle$ et on a $\alpha \in FV(\theta(c))$. Ainsi $\langle \lambda \overline{x_n} \mu\alpha.\theta(c) \| \alpha \rangle \rightarrow_{\eta_c} \langle \lambda \overline{x_n} \mu\alpha.\theta(c)[\alpha \leftarrow \overline{x_n} \bullet \beta] \| \alpha \rangle = \theta(\langle \lambda \overline{x_n} \mu\beta.c[\alpha \leftarrow \overline{x_n} \bullet \beta] \| \alpha \rangle)$
- Si $u = \langle \mu\alpha.c \| e \rangle$ alors premier cas : la réduction a lieu dans c ou e, auquel cas les hypothèses d'induction permettent de conclure. Deuxième cas : $v = c[\alpha \leftarrow e]$, ce qui signifie que e ne contient pas de $\tilde{\mu}$ alors $\theta(e)$ non plus par hypothèse sur θ. Ainsi $\theta(\langle \mu\alpha.c \| e \rangle) = \langle \mu\alpha.\theta(c) \| \theta(e) \rangle \rightarrow_\mu \theta(c)[\alpha \leftarrow \theta(e)] = \theta(c[\alpha \leftarrow e])$.

□

Preuve : 14 On le fait par induction sur la structure de u, sachant que bon nombre de cas on déjà été vérifiés dans la preuve du lemme 12

- Si $u = \tilde{\mu}x.c$ alors premier cas : la réduction a lieu dans c, auquel cas $v = \tilde{\mu}x.c'$ avec $c \rightarrow_R c'$. Les hypothèses d'induction font le reste. Deuxième cas $u = \tilde{\mu}x.\langle x \| e \rangle$ avec $x \notin FV(e)$. Or θ n'introduit pas de variable calleR donc $x \notin FV(\theta(e))$. Ainsi $\theta(\tilde{\mu}x.\langle x \| e \rangle)$
- Si $u = r \bullet e$ alors $u = r \bullet \tilde{\mu}x.c$ et $v = \tilde{\mu}x'.c[x \leftarrow \mu\alpha\langle x' \| r \bullet \alpha \rangle]$. Ainsi $\theta(r \bullet \tilde{\mu}x.c) = \theta(r) \bullet \tilde{\mu}x.\theta(c) \rightarrow_{\eta_e} \tilde{\mu}x'.\theta(c)[x \leftarrow \mu\alpha\langle x' \| \theta(r) \bullet \alpha \rangle] = \tilde{\mu}x'.\theta(c[x \leftarrow \mu\alpha\langle x' \| r \bullet \alpha \rangle]) = \theta(\tilde{\mu}x'.c[x \leftarrow \mu\alpha\langle x' \| r \bullet \alpha \rangle])$.
- Si $u = \langle r \| \tilde{\mu}x.c \rangle$ alors premier cas : la réduction a lieu dans r ou c, auquel cas les hypothèses d'induction permettent de conclure. Deuxième cas : $v = c[x \leftarrow r]$. Alors $\theta(\langle r \| \tilde{\mu}x.c \rangle) = \langle \theta(r) \| \tilde{\mu}x.\theta(c) \rangle \rightarrow_{\tilde{\mu}} \theta(c)[x \leftarrow \theta(r)] = \theta(c[x \leftarrow \theta(r)])$.

□

Preuve : 17 Soit Y une inconnue différente des X_i. $\theta \circ \sigma(Y) = \theta(Y)$ car $\sigma(Y) = Y$. D'un autre côté, $\theta \circ \sigma(X_i) = \theta(t_i) \overset{*}{\leftrightarrow}_R \theta(X_i)$, donc pour toute inconnue X, $\theta \circ \sigma(X) \overset{*}{\leftrightarrow}_R \theta(X)$. Le lemme 16 permet de conclure. □

Preuve : 18

- 1) $\rightarrow$ 2) : $\forall X \in W$, on a $A(X) = A \mid_W$ et $\theta(X) \in L_0(A(X))$ donc $\theta(X) \in L_0(A \mid_W (X))$.

– 2) $\rightarrow$ 3) : $\forall X \in W \cap D(\theta)$, on a $X \in W$ donc $X \in L_0(A \mid_W (X))$. Or $L_0(A \mid_W (X)) \subset L_0(A(X))$, donc $X \in L_0(A(X))$.
– 3) $\rightarrow$ 1) : $\forall X \in W$ si $X \notin D(\theta)$ alors $\theta(X) = X \in L_0(A(X))$ et si $X \in D(\theta)$ alors $\theta(X) \in L_0(A(X))$ par hypothèse.

□

Preuve : 19 D'après le lemme 11 on a $\hat{\theta}(\hat{\iota}(X)) = \widehat{\hat{\theta} \circ \iota}(\theta(X))$. Or $\theta(X) \in L_0(A(X))$ donc $\widehat{\hat{\theta} \circ \iota}(\theta(X)) = \widehat{\hat{\theta} \circ \iota \mid_{A(X))}}(\theta(X)) = \widehat{\hat{\theta} \circ \iota \mid_{A(X))}}(\theta(X)) = \hat{\theta}(\widehat{\iota \mid_{A(X)}}(X))$. □

Preuve : 20 Initialisation :

– Soit s une variable ou une constante. $\theta(s) = s$, d'où le résultat.
– Soit X une inconnue. Soit W tel que $X \in L_0(W)$ alors $A(X) \subset W$ par définition. Or $\theta(X) \in L_0(A(X))$ donc $\theta(X) \in L_0(W)$ aussi par définition.

Induction :

– Soit $\lambda x.r \in L_0(W)$, alors $\theta(\lambda x.r) = \lambda x.\theta(r)$ avec $\theta(r) \in L_0(W \cup \{x\})$, que x appartienne à $FV(r)$ ou non. Ainsi $\theta(\lambda x.r) \in L_0(W \cup \{x\} - \{x\}) = L_0(W)$.
– Idem pour les autres...

□

Preuve : 22 A, S est un système résolu donc S est de la forme $\cup\{X_i \stackrel{?}{=} t_i\}$. Il est clair que σ_S unifie S. De plus, pour $X = X_k$ on a $\sigma_S(X) = t_k \in L_0(A(X)$ par définition de la forme résolue, et pour $X \notin \{X_i\}$ on a $\sigma_S(X) = X \in L_0(A(X)$. Ainsi $\sigma_S \in U_t(A, S)$. Le lemme 17 permet de conclure. □

Preuve : 23 $\iota(\theta)$ étant de cardinal fini, on note $X_1, \ldots, X_n$ ses éléments (l'ordre n'a pas d'importance). Soit $Y = \{Y_1, \ldots, Y_n\} \subset UK$ tel que $Y \cap (\iota(\theta) \cup W \cup support(A)) = \emptyset$. Soient les substitutions $\rho_1 = [Y_i/X_i]_{i=1}^n$ et $\rho_2 = [X_i/Y_i]_{i=1}^n$. Soit $\sigma' = \rho_1 \circ \theta \mid_{UK(S)}$. Soit $\sigma = \sigma'_n$. On vérifie dans l'ordre que σ satisfait les propriétés 1, 2, 3 et 4.

1. Par définition.
2. $D(\sigma) \subset D(\sigma')$ par définition de la normalisation. Or $D(\sigma')) \subset UK(S)$ donc $D(\sigma) \subset UK(S)$. De plus $\iota(\sigma) = \iota(\sigma') \subset Y$ et $Y \cap (\iota(\theta) \cup W) = \emptyset$ par construction, donc $\iota(\sigma) \cap (W \cup D(\sigma)) = \emptyset$.
3. On montre d'abord que σ unifie S. Soit $\{u \stackrel{?}{=} v\} \in S$. On a $\theta \mid_{UK(S)} (u) =_R \theta \mid_{UK(S)} (v)$ donc $\sigma(u) = \sigma(v)$ (en effet on effectue d'abord un renommage, sans influence sur la R-convertibilité, suivi d'une η-normalisation ou d'une R normalisation). De plus $\forall X \in UK(S)\, \theta(X) \in L_0(A(X))$ implique $\rho_1 \circ \theta(X) \in L_0(A(X))$ d'après le lemme 20 (ρ_1 est compatible avec n'importe quoi car $Y_j \in L_0$). De plus $\sigma(X) = \eta(\rho_1 \circ \theta(X))$ ou $\sigma(X) = X$ donc $\sigma(X) \in L_0(A(X))$ d'après le lemme 21.

4. $\forall X \in UK(S) \quad \sigma(X) =_R \sigma'(X) = \rho_1 \circ \theta(X) = \rho_1 \circ \theta \mid_{UK(S)} (X)$ donc $\theta \leq_R \sigma[UK(S)]$. De même $\forall X \in UK(S) \quad \rho_2 \circ \sigma(X) \overset{*}{\leftrightarrow}_R \sigma(X)[Y_i \leftarrow X_i] \overset{*}{\leftrightarrow}_R \theta(X)[X_i \leftarrow Y_i][Y_i \leftarrow X_i]$. Or $\theta(X)[X_i \leftarrow Y_i][Y_i \leftarrow X_i] \overset{*}{\leftrightarrow}_R \theta(X)$ car $Y \cap (\iota(\theta) \cup W) = \emptyset$ par construction, ce qui implique que $Y \cap \iota(\theta) = \emptyset$. On a donc montré que $\sigma \leq_R \theta[UK(S)]$.

□

Preuve : 24 On montre que $U(S) \subset U(S')$ et la symétrie fait le reste. Soit $\theta \in U(S)$ et $\{u' \overset{?}{=} v'\} \in S'$. Par hypothèse, il existe $\{u \overset{?}{=} v\} \in S$ tel que $u =_R u'$ et $v =_R v'$. D'après le lemme 13 pour le cas du $\overline{\lambda\mu}$ et le lemme 15 (sachant $\theta \in U(S)$) pour le cas du $\overline{\lambda\mu\tilde{\mu}}$, on a $\theta(u) =_R \theta(u')$ et $\theta(v) =_R \theta(v')$. Or par hypothèse $\theta(u) =_R \theta(v)$ donc la transitivité de $=_R$ permet de conclure. □

Preuve : 25 Soit $\theta \in U_t(A \mid_{UK(S)}, S)$. Pour tout X dans $UK(S)$, on a $\theta(X) \in L_0(A(X))$ et $A(X) = A \mid_{UK(S)} (X)$ donc $\theta(X) \in L_0(A \mid_{UK(S)} (X)$. Réciproqement soit $\theta \in U_t(A, S)$. Or $Comp(\theta, A \mid_{UK(S)}, S)$ d'après le lemme 18, ce qui permet de conclure. □

Preuve : 26 Soit $J \in P_n$ et $\rho \in U_s(A_J, S_J)$. On a ρ unifie S_J donc pour toute équation $u \overset{?}{=} v$ de S on a $\rho(u[E_j \leftarrow \tilde{\mu}x_j.\langle R_j \| x_j \bullet e_0\rangle]_{j\in J}) =_R \rho(v[E_j \leftarrow \tilde{\mu}x_j.\langle R_j \| x_j \bullet e_0\rangle]_{j\in J})$. Or $\rho(E_j) =_R \rho(\tilde{\mu}x_j.\langle R_j \| x_j \bullet e_0\rangle)$ par hypothèse donc d'après le lemme 17 on a $\rho =_R \rho \circ [E_j \leftarrow \tilde{\mu}x_j.\langle R_j \| x_j \bullet e_0\rangle]_{j\in J}$. Ainsi $\rho(u) =_R \rho(v)$, ρ unifie S. Finalement soit $X \in UK(S)$ alors $\rho(X) \in L_0(A_J(X))$ par hypothèse et $A_J(X) = A(X)$ car $X \neq R_j$. Cela nous donne $\rho(X) \in L_0(A(X))$. □

Preuve : 27 Soit $\rho \in U_{ts}(A, S)$. On décompose ρ sous la forme $\rho = [E_j \mapsto \tilde{\mu}x_j.c_j]_{j\in J} \cup \theta$. Soit $\rho' = \rho \cup [R_j \mapsto \lambda y.\mu\alpha.c_j[x_j \leftarrow y]]$ avec α de type base pour correspondre avec e_0. On a $\rho' \mid_{UK(S)} = \rho$ comme prévu, donc ρ' unifie S. De plus ρ' unifie le reste de S_J par construction. En effet $\tilde{\mu}x_j.\langle R_j \| x_j \bullet e_0\rangle[R_j \leftarrow \lambda y.\mu\alpha.c_j[x_j \leftarrow y]] \overset{*}{\rightarrow}_R \tilde{\mu}x_j.c_j$. Finalement $\forall X \in UK(S) \cup \{R_j\}_{j\in J}$ si $X \in UK(S)$ alors $\rho'(X) = \rho(X) \in L_0(A(X)) = L_0(A_J(X))$ et si $X = R_j$ alors $\rho'(X) = \rho'(R_j) = \lambda y.\mu\alpha.c_j[x_j \leftarrow y]$. Or $\tilde{\mu}x_j.c_j \in L_0(A(E_j))$ donc $c_j \in L_0(A(E_j) \cup \{x_j\})$ donc $c_j[x_j \leftarrow y] \in L_0(A(E_j) \cup \{y\})$ donc $\mu\alpha.c_j[x_j \leftarrow y] \in L_0(A(E_j) \cup \{y\})$ donc $\rho'(R_j) = \lambda y.\mu\alpha.c_j[x_j \leftarrow y] \in L_0(A(E_j))$. □

Preuve : 31 On le prouve règle par règle. A chaque fois il faut s'assurer que les deux conditions (unification et compatibilité) sont vérifiées.

EE : θ unifie S donc θ unifie $(S \cup \{t = t'\})$. En effet $t =_R t' \quad \Rightarrow \theta(t) =_R \theta(t')$ d'après le lemme 13 . De plus, $\forall X \in UK(S, t, t') \cap D(\theta)$ on a $X \in UK(S)$ donc $\theta(X) \in L_0(A(X)) = L_0(A \cup A'(X))$.

UE : θ unifie $(\{X \overset{?}{=} t\} \cup S[X \leftarrow t])$ donc $\theta(X) = \theta(t)$. Ainsi $\forall \{u \overset{?}{=} v \in S$, on a $\theta(u) = \theta(u[X \leftarrow t])$ d'après le lemme 17. Or $\theta(u[X \leftarrow t]) =_R \theta(v[X \leftarrow t])$ par hypothèse, donc $\theta(u) =_R \theta(v)$. Ainsi θ unifie $(\{X \overset{?}{=} t\} \cup S)$. De plus $\forall X \in UK(S, X, t) \quad \theta(X) \in L_0(A(X))$ par hypothèse.

EA : θ unifie $(S \cup \{t \stackrel{?}{=} t'\}) \quad \Rightarrow \quad \theta$ unifie S. De plus $\forall X \in UK(S), \quad X \notin support(A')$ par hypothèse. Ainsi $A \cup A'(X) = A(X)$. D'où $\forall X \in UK(S), \quad \theta(X) \in L_0(A(X))$.

□

Preuve : 32 Soient $X_1, \ldots, X_n$ les éléments de $D(\theta)$ (donc inclus dans $UK(S)$). $S' = \{X \stackrel{?}{=} \theta(X)\}_{X \in D(\theta)}$. On a $A, S \stackrel{*}{\Longrightarrow}_{EA} A, S' \cup S \stackrel{*}{\Longrightarrow}_{UE} A, S' \cup S[X_1 \leftarrow \theta(X_1)] \ldots [X_n \leftarrow \theta(X_n)]$ (par exemple dans cet ordre).Or $S[X_1 \leftarrow \theta(X_1)] \ldots [X_n \leftarrow \theta(X_n)] = S[X \leftarrow \theta(X)]_{X \in D(\theta)}$ car $\iota(\theta) \cap (D(\theta)) = \emptyset$. Or $S[X \leftarrow \theta(X)]_{X \in D(\theta)} = \theta(S)$ et pour chacune des équations de $\theta(S)$ les deux membres sont R-convertibles car $\theta \in U_s(A, S)$ donc on peut appliquer EE pour éliminer $\theta(S)$: $A, S' \cup \theta(S) \stackrel{*}{\Longrightarrow}_{EE} A, S'$.

A, S' est de plus effectivement résolu car $\iota(\theta) \cap D(\theta) = \emptyset$ et on vérifie également que $\sigma_{S'} = \theta$. □

Preuve : 33 En fait seule la condition 3) a changé par rapport au lemme 23 dont on reprend les notations. On n'a donc que "$\sigma \in PU(A, S)$" à prouver. Soit $\{u \stackrel{?}{=} v\} \in S$.

- Si $\theta(u) =_R \theta(v)$ alors $\sigma(u) =_R \sigma(v)$ également.
- Si $\theta(u) \downarrow_R \stackrel{?}{=} \theta(v) \downarrow_R$ est flexible-flexible :
  1. Cas $\iota(E) \stackrel{?}{=} \iota'(E')$: $\rho_1(\theta(u)) =_R \rho_1(\iota(E)) = \widehat{\rho_1 \circ \iota}(\rho_1(E))$. Or $\rho_1(E)$ est une inconnue calleE car ρ_1 est un renommage. Idem pour v. On a donc une équation flexible-flexible.
  2. Cas $\langle \iota(R) \| e \rangle \stackrel{?}{=} \langle \iota'(R') \| e' \rangle$. On a $\rho_1 \circ \theta(u) =_R \langle \rho_1(\iota(R)) \| \rho_1(e) \rangle$ qui est sous forme R-normale car ρ_1 est un renommage. Ainsi $\sigma(u) \downarrow_R = \langle \widehat{\rho_1 \circ \iota}(\rho_1(R)) \| \rho_1(e) \rangle$. Avec $\rho_1(R)$ qui est une calleR car ρ_1 est un renommage, donc $\langle \widehat{\rho_1 \circ \iota}(\rho_1(R)) \| \rho_1(e) \rangle$ est une forme normale. Idem pour v. On a donc une équation flexible-flexible.

La justification de compatibilité ($Comp(\theta, A, UK(S))$) est la même que celle du lemme 23. □

Preuve : 34 Soit A, S un tel système. Il peut s'écrire $A, \{X_i \stackrel{?}{=} t_i\} \cup \{\iota_j(E_j) = \iota'_j(E'_j)\} \cup \{\langle \iota_k(R_k) \| e_k \rangle \stackrel{?}{=} \langle \iota'_k(R'_k) \| e'_k \rangle\}$. La substitution suivante est $fn\eta l$, $\iota(\theta) = \emptyset$ et à valeur dans L_0 donc compatible avec toute fonction d'autorisation. Soit $\theta = [X_i \mapsto t_i] \cup [E_j, E'_j \mapsto e_{j,c}] \cup [R_k, R'_k \mapsto \lambda \overline{x_{n_k}}.\mu\alpha.\langle r_c \| \overline{x_n} \bullet e_c \rangle]$. On vérifie que $\theta \in U_s(A, S)$. □

Preuve : 38 Les règles EAC, PEC et PRC_X font diminuer strictement la mesure m_1. EAE et les XD ne font pas augmenter la mesure m_1. En outre elle font diminuer la mesure m_2. Or l'ordre lexicographique $< m_1, m_2 >$ est noethérien, d'où le résultat. □

Preuve : 39 Le système A, S est non pré-résolu donc il contient une équation $\{u \stackrel{?}{=} v\}$ non pré-résolue dans A, S.

– Si u et v sont des calleR, alors ils sont sous forme normale car l'équation n'est pas résolue, donc de la forme $\lambda\overline{x_n}\mu\alpha....$ Ils présentent exactement la même tête car ils sont de même type. On peut donc appliquer RD. Dans ce cas $\rho' = \rho$ est également pré-solution du système obtenu.
– Si u et v sont des calleE, alors on distingue quatre cas possibles. Il n'y en a pas plus d'après les contraintes de pré-unifiablilité et de non-résolution du système considéré :
  1. $e_a \stackrel{?}{=} e_a$: on peut appliquer EAE. Dans ce cas $\rho' = \rho$ est également pré-solution du système obtenu.
  2. $r \bullet e \stackrel{?}{=} r' \bullet e'$: on peut appliquer ED. Dans ce cas $\rho' = \rho$ est également pré-solution du système obtenu.
  3. $\iota(E) \stackrel{?}{=} r' \bullet e'$. Dans ce cas on a nécessairement $\rho(E) = r \bullet e$ et on peut appliquer PEC. On vérifie que $\rho' \in PU(A', S')$.
  4. $\iota(E) \stackrel{?}{=} e_a$. Dans ce cas on peut appliquer EAC.
– Si u et v sont des capsules, alors on distingue cas possibles. Il n'y en a pas plus car d'après les contraintes de pré-unifiablilité et de non-résolution du système considéré :
  1. $\langle r_a \| e\rangle \stackrel{?}{=} \langle r_a \| e'\rangle$ et r_a n'est pas une inconnue : on peut appliquer CD_{ra}. Dans ce cas $\rho' = \rho$ est également pré-solution du système obtenu.
  2. $\langle \lambda\overline{x_n}.\mu\alpha.c \| e_a\rangle \stackrel{?}{=} \langle \lambda\overline{x_n}.\mu\alpha.c' \| e'_a\rangle$: on peut appliquer CD_{ea}. Dans ce cas $\rho' = \rho$ est également pré-solution du système obtenu.
  3. $\{\langle r \| \iota(E)\rangle \stackrel{?}{=} c\}$: on peut appliquer PEC si E n'est pas de type de base. On vérifie que $\rho' \in PU(A', S')$ et $\rho' = \rho[UK(S)]$.
  4. $\langle \iota(R) \| e\rangle \stackrel{?}{=} c$ et $\rho(R) = ct_r(r_c)$: si $c = \langle r \| \iota(E)\rangle$ reculez d'une case. Sinon $c = \langle r_a \| e'\rangle$ et $\eta \circ \rho(\langle \iota(R) \| e\rangle) = \langle r_c \| e\rangle$ donc $r_a = r_c$ donc on peut appliquer PRC_{ri}. On vérifie que $\rho' \in PU(A', S')$ et $\rho' = \rho[UK(S)]$.
  5. $\langle \iota(R) \| e\rangle \stackrel{?}{=} c$ et $\rho(R) = ct_r(y)$: on peut appliquer PRC_{rp}. On vérifie que $\rho' \in PU(A', S')$ et $\rho' = \rho[UK(S)]$.
  6. IDEM pour les deux autres PRC.

□

Preuve : 1 Soit $\theta, A, S = \theta_1, A_1, S_1 \Rightarrow \theta_2, A_2, S_2 \Rightarrow \ldots \Rightarrow \theta_n, A_n, S_n$ une suite de transformations de SET telle que plus aucune transformation ne s'applique. Une telle suite existe d'après le lemme 38. Le lemme 39 nous garantit que A_n, S_n est pré-résolu sinon on peut encore transformer, donc $\sigma_{S_n} \in PU_s(A_n, S_n)$. Par correction (lemme 37)on en déduit que $\sigma_{S_n} \in PU_s(A, S)$. Le lemme 39 et une récurrence montre que $\theta_n = \theta[UK(S)]$. De plus $\theta_n \in PU_s(A_n, S_n)$ donc d'après le lemme 35 on a $\sigma_{S_n} \leq_R \theta_n$, d'où $\sigma_{S_n} \leq_R \theta[UK(S)]$. □

Table des matières