ENS Lyon
Magistère d'Informatique et Modélisation
Mémoire de DEA – 2004

Étude du λ-calcul différentiel et extension au cas classique: le $\lambda\mu$-calcul différentiel

Lionel VAUX
Stage supervisé par Thomas EHRHARD et Laurent RÉGNIER
Équipe de Logique de la Programmation
Institut de Mathématiques de Luminy

Remerciements

Le stage s'est déroulé à l'Institut de Mathématiques de Luminy, à Marseille, sous la direction de Thomas Ehrhard Laurent Régnier. Je tiens à les remercier chaleureusement pour leur acceuil, et le plaisir que j'ai pris à travailler avec eux.

Merci cette année encore à Pierre, mon co-bureau durant ce stage, avec qui j'ai chaque jour appris quelque chose (un partage de connaissance et d'intuition petitement rendu).

Un autre merci à Damiano et Michele pour nos échanges sur la logique en général et les réseaux différentiels polarisés en particulier.

Enfin, je tiens remercier l'équipe de logique toute entière pour l'environnement de travail agréable et stimulant qu'elle fournit.

Table des matières

Notations

On note $\mathbf{N}$ l'ensemble des entiers naturels et $\mathbf{N}^* = \mathbf{N} \setminus \{0\}$.

Si S est un ensemble, on note $\mathbf{N}\langle S\rangle$ le $\mathbf{N}$-module généré par S, c'est à dire l'ensemble des multi-ensembles finis d'éléments de S. Par abus de notation, on confond chaque élément s de S avec le multi-ensemble singleton formé d'une copie de s. Si $\mathcal{M}_1, \ldots, \mathcal{M}_n \in \mathbf{N}\langle S\rangle$, on note $\sum_{i=1}^{n} \mathcal{M}_i = \mathcal{M}_1 + \ldots + \mathcal{M}_n$ la somme (réunion disjointe) des $\mathcal{M}_i$. En particulier, $\sum_{s\in S} a_s s$ est le multi-ensemble qui contient a_s copies de s pour chaque $s \in S$ (seul un nombre fini de s vérifie $a_s \neq 0$). On a :

$$\sum_{s\in S} a_s s + \sum_{s\in S} b_s s = \sum_{s\in S} (a_s + b_s) s.$$

L'application du λ-calcul est notée à la Krivine : $(s)\,t$ est l'application du terme s au terme t.

On note $x \in s$ pour « x est libre dans le terme s », c'est à dire qu'on identifie s et l'ensemble de ses variables libres. En particulier, on pourra écrire $x \in s \cup t$ pour « x est libre dans s ou dans t »

Introduction

En mathématiques, la linéarité est une propriété algébrique fondamentale qu'il n'est pas nécessaire de présenter. Dans le λ-calcul en revanche, la linéarité prend un tout autre sens : on appelle terme linéaire un terme du calcul dans lequel toute variable abstraite est utilisée une et une seule fois. Ainsi, l'identité λxx est linéaire en son argument. C'est d'ailleurs le seul entier de Church qui soit linéaire : $\overline{0} = \lambda f \lambda xx$ n'est pas linéaire car f n'est pas utilisé et par exemple $\overline{2} = \lambda f \lambda x (f)\,(f)\,x$ non plus, car f est utilisé deux fois.

Cette notion de linéarité est évidemment très liée à la gestion de la mémoire dans les programmes. Savoir qu'une variable est utilisée linéairement dans un terme permet d'optimiser l'exécution (*i.e.* la réduction), car il n'est jamais nécessaire de copier son contenu, ni de l'effacer.

L'un des progrès de la logique linéaire sur la logique intuitioniste, dans le cadre de la correspondance de Curry-Howard, est justement de prendre en compte la linéarité. Les sémantiques dénotationelles du λ-calcul basées sur des modèles de la logique linéaire permettent précisément d'isoler les termes linéaires.

Le λ-calcul différentiel trouve sa source dans l'étude de modèles de la logique linéaire dans lesquels la linéarité de la logique ou du λ-calcul coïncide avec la linéarité algébrique usuelle. En effet, c'est d'abord dans les espaces de Koethe [Ehr01], puis dans les espaces de finitude [Ehr04], que Thomas Ehrhard a mis en évidence qu'on pouvait donner une sémantique dénotationelle à la logique linéaire dans laquelle les morphismes associés aux preuves sont des fonctions analytiques entre espaces vectoriels, et surtout telle que ceux qui interprètent des termes linéaires sont des fonctions linéaires. Le fait que les morphismes sont analytiques permet en particulier de les dériver, et la dérivée d'un morphisme est, comme d'habitude en mathématiques, sa plus proche approximation linéaire.

C'est le fondement sémantique du λ-calcul différentiel, mis au point par Thomas Ehrhard et Laurent Régnier dans [ER03]. Dans cette extension du λ-calcul, en plus des termes habituels, on peut former des dérivées : si s et u sont des termes, alors $\mathrm{D}s \cdot u$ est un terme qui représente la dérivée de s selon u. À la β-reduction usuelle, on ajoute la différenciation :

$$\mathrm{D}\lambda xs \cdot u = \lambda x \left(\frac{\partial s}{\partial x} \cdot u \right)$$

qui correspond à la notion intuitive de dérivée d'une fonction analytique. En effet, pour une fonction analytique f à une variable de E dans F, la dérivée f' de f est une fonction linéaire de E dans l'espace des fonctions F^E. $\frac{\partial s}{\partial x} \cdot u$ est donc un terme qui dépend linéairement de u. Plus précisément, on veut définir $\frac{\partial s}{\partial x} \cdot u$ comme le terme obtenu en remplaçant une occurence de x par u dans s. On ne peut cependant se contenter d'une telle définition : quelle occurence de x faut-il remplacer ? La notion de somme apparaît naturellement : $\frac{\partial s}{\partial x} \cdot u$ est la somme des termes obtenus en remplaçant l'une des occurences de x par u. Cette notion de terme dérivé est donc très proche celle de polynôme dérivé (et donc plus généralement de la dérivation dans les fonctions analytiques).

La syntaxe véritablement adaptée à la logique linéaire n'est cependant pas le λ-calcul, mais plutôt les réseaux de preuve, ou plus simplement les réseaux d'interaction [Laf95]. Il y a bien sûr un lien fort entre λ-calcul et réseaux (lien très précisément établi par Laurent Régnier dans sa thèse [Ré92]). Il était clair dès l'élaboration du λ-calcul différentiel qu'on pouvait tout aussi bien mener le même raisonnement dans les réseaux pour aboutir à des réseaux d'interaction différentiels. Dans [ER04] Thomas Ehrhard et Laurent RÉGNIER formalisent ces réseaux, mais sans la construction de boîtes, or les boîtes sont nécessaires pour simuler le λ-calcul.

La première partie de ce rapport décrit précisément le λ-calcul différentiel tel qu'il est présenté dans [ER03], allant jusqu'à reprendre la structure globale de l'article. De cette manière, les résultats suivants reposent sur une base formelle bien établie. Dans la deuxième partie, on définit (plutôt informellement) les réseaux différentiels avec boîte, puis on présente la traduction du λ-calcul différentiel dans ces réseaux.

La suite du rapport est consacrée à une hybridation du λ-calcul différentiel et des réseaux différentiels avec le $\lambda\mu$-calcul et son pendant dans les réseaux : les réseaux polarisés. Le $\lambda\mu$-calcul de Michel PARIGOT est une extension du λ-calcul à la logique classique : là où les termes typés du λ-calcul sont isomorphes aux preuves en déduction naturelle intuitioniste, les termes typés du $\lambda\mu$-calcul sont isomorphes aux preuves en logique classique [Par92]. Dans le calcul des séquents, on passe de la logique intuitioniste à la logique classique en autorisant les séquents à avoir plus d'une conclusion. Le $\lambda\mu$-calcul reprend cette idée en permettant l'écriture de termes qui ont en un certain sens plusieurs sorties : la sortie principale qui est représentée par le terme lui-même, et des sorties nommées qui sont des sous-termes gardés par un nom.

Le problème soulevé par l'hybridation du λ-calcul différentiel avec le $\lambda\mu$-calcul est celui de la différenciation de termes à plusieurs sorties : qu'est-ce que dériver une sortie nommée ? La seule réponse qui semble raisonnable est donnée par le $\lambda\mu$-calcul différentiel présenté dans la troisième partie. Cette solution satisfait l'intuition, mais introduit des difficultés techniques. En particulier la confluence du calcul obtenu n'est toujours pas établie.

Enfin, la dernière partie introduit les réseaux différentiels polarisés. Les réseaux polarisés sont des réseaux dans lesquels certaines contraintes sont relâchées sur les constructions négatives de la logique linéaire : affaiblissement et contraction peuvent être utilisés sur toutes les formules négatives au lieu des seules formules $?A$. Trouver les réseaux qui correspondent au $\lambda\mu$-calcul différentiel n'est pas difficile. Le problème intéressant est plutôt d'inventer les constructions duales de l'affaiblissement et de la contraction sur les formules négatives : comment doivent fonctionner le co-affaiblissement et la co-contraction sur les formules positives ? Quelques pistes sont ouvertes pour des travaux futurs.

1 Le λ-calcul différentiel

Informellement, on se donne une infinité de λ-variables, ou simplement variables : $x, y, z, \ldots$, et on forme les termes du calcul par les constructions suivantes :

- variable x ;
- λ-abstraction $\lambda x t$;
- application $(s)\, t$
- différentielle $\mathrm{D}t \cdot u$;
- somme[1] $s + t$.

[1]Dans ce rapport, on se contente de pouvoir faire des sommes de termes, pas des combinaisons linéaires quelconques. Dans [ER03], Thomas EHRHARD et Laurent RÉGNIER étudient un cas plus général, dans lequel on se donne un semi-anneau R et où l'on peut former n'importe quelle combinaison linéaire finie de termes à coefficients dans R. Ceci rend certaines preuves un peu plus complexes, car il devient difficile de raisonner par induction sur les termes. Par exemple, si $-1 \in \mathrm{R}$, on peut former le terme $s - s = 0$ qui est structurellement plus petit que s.

Une telle grammaire ne saurait toutefois définir convenablement la syntaxe du calcul. D'abord, la seule différentielle donnée ci-dessus ne permet ne dériver que par rapport à la première abstraction. Il est donc nécessaire d'ajouter une indication de l'abstraction par rapport à laquelle on dérive : on note $\mathrm{D}_i s \cdot u$ la dérivée de s selon u par rapport à sa i-ème abstraction. De plus, au delà de l'α-convertibilité, on souhaite considérer comme égaux deux termes qui ne diffèrent que par permutation différentielle, c'est à dire $\mathrm{D}_j(\mathrm{D}_i s \cdot u) \cdot v = \mathrm{D}_i(\mathrm{D}_j s \cdot v) \cdot u$ (cela traduit que, sémantiquement, les morphismes sont infiniment différentiables, et donc vérifient le théorème de Schwartz). Par ailleurs, il faut prendre en compte l'associativité et la commutativité de la somme, en tant que construction syntaxique. On voit donc que la notion d'égalité sur les termes de la grammaire grossière donnée ci-dessus est complexe à la fois à définir et à utiliser. On va donc se ramener à une syntaxe de base simple, sur laquelle l'égalité sera plus facilement définie et on donnera le moyen d'écrire les constructions de l'ébauche de grammaire précedente à partir de la syntaxe de base.

1.1 Syntaxe

1.1.1 Syntaxe de base

On suppose donné un ensemble infini dénombrable de variables $(x, y, z, \ldots)$.

Définition 1 (Termes) *On définit par induction sur l'entier k l'ensemble Δ_k des termes simples, de taille inférieure ou égale à k. On fixe $\Delta_0 = \emptyset$ et on construit Δ_{k+1} à partir de Δ_k comme suit.*

- *Si $t \in \Delta_k$ alors $t \in \Delta_{k+1}$.*
- *Si x est une variable, $n \in \mathbf{N}$, $i_1, \ldots, i_n \in \mathbf{N}^*$ et $u_1, \ldots, u_n \in \Delta_k$ alors*

$$\mathrm{D}_{i_1,\ldots,i_n} x \cdot (u_1, \ldots, u_n) \in \Delta_{k+1}.$$

  Ce terme est identifié avec tous les termes de la forme

$$\mathrm{D}_{i_{\sigma(1)},\ldots,i_{\sigma(n)}} x \cdot (u_{\sigma(1)}, \ldots, u_{\sigma(n)})$$

  où σ est une permutation de $\{1, \ldots, n\}$[2]*.*
- *Si x est une variable, $n \in \mathbf{N}$ et $s, u_1, \ldots, u_n \in \Delta_k$ alors*

$$\mathrm{D}_1^n \lambda x s \cdot (u_1, \ldots, u_n) \in \Delta_{k+1}.$$

  Ce terme est identifié avec tous les termes de la forme

$$\mathrm{D}_1^n \lambda x s \cdot (u_{\sigma(1)}, \ldots, u_{\sigma(n)})$$

  où σ est une permutation de $\{1, \ldots, n\}$.
- *Si $s \in \Delta_k$ et $t \in \mathbf{N}\langle\Delta_k\rangle$ alors*

$$(s)\, t \in \Delta_{k+1}.$$

On note $\Delta = \bigcup_{k \in \mathbf{N}} \Delta_k$ l'ensemble des termes simples. On appelle termes généraux, ou plus simplement termes, les éléments de $\bigcup_{k \in \mathbf{N}} \mathbf{N}\langle\Delta_k\rangle = \mathbf{N}\langle\Delta\rangle$.

On travaille évidemment à α-conversion près[3].

[2] C'est à dire qu'il faudrait plutôt considérer un multi-ensemble $\mathcal{M} = (i_1, u_1) + \ldots + (i_n, u_n)$ et construire un terme de la forme $\mathrm{D}_{\mathcal{M}} s$. La notation choisie est cependant plus lisible.

[3] Là encore, il aurait été plus rigoureux de formaliser le calcul en utilisant des indices de De Bruijn, mais on utilise des variables liées pour des raisons évidentes de lisibilité.

1.1.2 Extension de la syntaxe

On généralise les constructions du calcul (abstraction, différentielle et application) à tous les termes de la façon suivante.

Différentielle On construit l'opérateur de différentielle généralisé, noté abusivement $\mathrm{D}_i s \cdot u$, pour s et u des termes simples par induction sur s :

$$\begin{array}{rcl} \mathrm{D}_i(\mathrm{D}_{i_1,\ldots,i_n} x \cdot (u_1,\ldots,u_n)) \cdot v & = & \mathrm{D}_{i,i_1,\ldots,i_n} x \cdot (v, u_1, \ldots, u_n) \\ \mathrm{D}_i(\mathrm{D}_1^n \lambda x s \cdot (u_1,\ldots,u_n)) \cdot v & = & \left\{ \begin{array}{ll} \mathrm{D}_1^{n+1} \lambda x s \cdot (v, u_1, \ldots, u_n) & \text{si } i = 1 \\ \mathrm{D}_1^n \lambda x (\mathrm{D}_{i-1} s \cdot v) \cdot (u_1, \ldots, u_n) & \text{sinon} \end{array} \right. \\ \mathrm{D}_i(s)\, t \cdot u & = & (\mathrm{D}_{i+1} s \cdot u)\, t \end{array}$$

Les objets obtenus sont encore des termes simples. On étend cette construction au cas où s et u sont des termes généraux par bilinéarité :

$$\mathrm{D}_i \left(\sum_{s \in \Delta} a_s s \right) \cdot \left(\sum_{u \in \Delta} b_u u \right) = \sum_{s,u \in \Delta} a_s b_u \, (\mathrm{D}_i s \cdot u)$$

Linéarité des positions fonctionnelles On étend les autres constructeurs du calcul par linéarité :

$$\begin{array}{rcl} \lambda x \left(\sum_{s \in \Delta} a_s s \right) & = & \sum_{s \in \Delta} a_s \lambda x s \\ \left(\sum_{s \in \Delta} a_s s \right) t & = & \sum_{s \in \Delta} a_s (s)\, t \end{array}$$

Il faut porter une attention particulière au fait que l'application d'une fonction à un argument est une construction linéaire en la fonction, mais pas en l'argument — sauf bien sûr si la fonction elle-même est linéaire.

Il est clair qu'on peut construire tous les termes du calcul en utilisant seulement les variables et les opérateurs généralisés. La notation identifiant les opérateurs généralisés (qui sont en fait des fonctions des termes dans les termes) et les constucteurs du calcul est abusive mais inoffensive : lorsqu'une écriture est ambigüe, elle désigne de toute façon le même objet quelle que soit la manière dont on l'interprète.

Il est aisé d'établir que pour tout s, $\mathrm{D}_i(\mathrm{D}_j s \cdot v) \cdot u = \mathrm{D}_j(\mathrm{D}_i s \cdot u) \cdot v$. En notant $\mathrm{D}_{i_1,\ldots,i_n} s \cdot (u_1, \ldots, u_n)$ le terme $\mathrm{D}_{i_1}(\ldots(\mathrm{D}_{i_n} s \cdot u_n)\ldots) \cdot u_1$, on obtient pour toute permutation σ de $\{1, \ldots, n\}$:

$$\mathrm{D}_{i_1,\ldots,i_n} s \cdot (u_1, \ldots, u_n) = \mathrm{D}_{i_{\sigma(1)},\ldots,i_{\sigma(n)}} s \cdot (u_{\sigma(1)}, \ldots, u_{\sigma(n)}).$$

1.1.3 Substitutions

Le calcul comporte deux substitutions : la substitution usuelle du λ-calcul qui remplace toutes les occurences libres d'une λ-variable par un terme, et la substitution linéaire ou dérivation, qui ne remplace moralement[4] qu'une seule occurence.

[4]Il ne s'agit pas toujours d'une «vraie» substitution : voir le cas de la dérivation d'une application.

Définition 2 (Substitution) *On définit par induction sur s la substitution usuelle $s[t/x]$:*

$$\begin{array}{rcl}
(\mathrm{D}_{i_1,\ldots,i_n} y \cdot (u_1,\ldots,u_n))[t/x] & = & \begin{cases} \mathrm{D}_{i_1,\ldots,i_n} t \cdot (u_1[t/x],\ldots,u_n[t/x]) & \textit{si } x = y \\ \mathrm{D}_{i_1,\ldots,i_n} y \cdot (u_1[t/x],\ldots,u_n[t/x]) & \textit{si } x \neq y \end{cases} \\
(\mathrm{D}_1^n \lambda y s \cdot (u_1,\ldots,u_n))[t/x] & = & \mathrm{D}_1^n \lambda y s[t/x] \cdot (u_1[t/x],\ldots,u_n[t/x]) \ \textit{avec } x \neq y \\
((s)\, w)[t/x] & = & (s[t/x])\, w[t/x] \\
(\sum_{s\in\Delta} a_s s)[t/x] & = & \sum_{s\in\Delta} a_s s[t/x]
\end{array}$$

Définition 3 (Dérivation) *On définit par induction sur s la dérivée partielle $\frac{\partial s}{\partial x} \cdot t$:*

$$\begin{array}{rcl}
\dfrac{\partial \mathrm{D}_{i_1,\ldots,i_n} y \cdot (u_1,\ldots,u_n)}{\partial x} \cdot t & = & \delta_{x=y} \mathrm{D}_{i_1,\ldots,i_n} t \cdot (u_1,\ldots,u_n) \\
& & + \displaystyle\sum_{1\leq j\leq n} \mathrm{D}_{i_1,\ldots,i_n} y \cdot \left(u_1,\ldots,\frac{\partial u_j}{\partial x}\cdot t,\ldots,u_n\right) \\
\dfrac{\partial \mathrm{D}_1^n \lambda y s \cdot (u_1,\ldots,u_n)}{\partial x} \cdot t & = & \mathrm{D}_1^n \lambda y \left(\dfrac{\partial s}{\partial x}\cdot t\right) \cdot (u_1,\ldots,u_n) \\
& & + \displaystyle\sum_{1\leq j\leq n} \mathrm{D}_1^n \lambda y s \cdot \left(u_1,\ldots,\frac{\partial u_j}{\partial x}\cdot t,\ldots,u_n\right) \\
\dfrac{\partial (s)\, w}{\partial x} \cdot t & = & \left(\dfrac{\partial s}{\partial x}\cdot t\right) w + \left(\mathrm{D}_1 s \cdot \left(\dfrac{\partial w}{\partial x}\cdot t\right)\right) w \\
\dfrac{\partial \sum_{s\in\Delta} a_s s}{\partial x} \cdot t & = & \displaystyle\sum_{s\in\Delta} a_s \left(\frac{\partial s}{\partial x}\cdot t\right)
\end{array}$$

Le cas intéressant est celui de la dérivation d'une application : on s'assure de la linéarité en utilisant une différentielle sur la première abstraction de s. Sans cette précaution, si on se contente de substituer une occurence de x dans w, rien n'empêche s de dupliquer ou effacer w et donc le terme substitué à x.

Les lemmes suivants établissent que les substitutions « se comportent bien ». Les preuves sont aisées, par induction sur les termes simples.

Différentielle et substitutions

Lemme 1 *Pour tout $i \in \mathbf{N}^+$, toute variable x et tous termes s, u et v, on a :*

$$\begin{array}{rcl}
(\mathrm{D}_i s \cdot u)[v/x] & = & \mathrm{D}_i s[v/x] \cdot u[v/x] \\
\dfrac{\partial \mathrm{D}_i s \cdot u}{\partial x} \cdot v & = & \mathrm{D}_i \left(\dfrac{\partial s}{\partial x}\cdot v\right) \cdot u + \mathrm{D}_i s \cdot \left(\dfrac{\partial u}{\partial x}\cdot v\right)
\end{array}$$

Commutation des substitutions

Lemme 2 *Pour toutes variables $x \neq y$, et tous termes t,u et v, tels que $x \notin v$:*

$$t[u/x][v/y] = t[v/y][u[v/y]/x].$$

Linéarité de la dérivation

Lemme 3 *Pour toute variable x et tous termes s et u, si $x \notin s$ alors* $\frac{\partial s}{\partial x} \cdot u = 0$.

Lemme 4 *Pour toute variable x et tous termes $t, u_1, \ldots, u_n$, on a :*

$$\frac{\partial t}{\partial x} \cdot \left(\sum_{j=1}^{n} u_j\right) = \sum_{j=1}^{n} \frac{\partial t}{\partial x} \cdot u_j.$$

Commutation des dérivations

Lemme 5 *Pour toutes variables x, y, et tous termes t,u et v, tels que $x \notin v$:*

$$\frac{\partial}{\partial y}\left(\frac{\partial t}{\partial x} \cdot u\right) \cdot v = \frac{\partial}{\partial x}\left(\frac{\partial t}{\partial y} \cdot v\right) \cdot u + \frac{\partial t}{\partial x} \cdot \left(\frac{\partial u}{\partial y} \cdot v\right).$$

En particulier, si $y \notin u$ alors

$$\frac{\partial}{\partial y}\left(\frac{\partial t}{\partial x} \cdot u\right) \cdot v = \frac{\partial}{\partial x}\left(\frac{\partial t}{\partial y} \cdot v\right) \cdot u.$$

C'est un théorème de Schwartz pour les dérivées partielles. En particulier, pour toutes variables $x_1, \ldots, x_n$, et tous termes $s, u_1, \ldots, u_n$, tels que pour tous i, j, $x_i \notin u_j$, pour toute permutation σ de $\{1, \ldots, n\}$, on pourra écrire

$$\frac{\partial^n s}{\partial x_1 \ldots \partial x_n} \cdot (u_1, \ldots, u_n)$$

au lieu de

$$\frac{\partial}{\partial x_{\sigma(1)}}\left(\cdots \frac{\partial s}{\partial x_{\sigma(n)}} \cdot u_{\sigma(n)} \cdots\right) \cdot u_{\sigma(1)}.$$

Dérivation et substitution

Lemme 6 *Pour toutes variables $x \neq y$, et tous termes t,u et v, tels que $x \notin u \cup v$:*

$$\frac{\partial t[u/x]}{\partial y} \cdot v = \left(\frac{\partial t}{\partial y} \cdot v\right)[u/x] + \left(\frac{\partial t}{\partial x} \cdot \left(\frac{\partial u}{\partial y} \cdot v\right)\right)[u/x].$$

En particulier, si $y \notin u$ alors

$$\frac{\partial t[u/x]}{\partial y} \cdot v = \left(\frac{\partial t}{\partial y} \cdot v\right)[u/x].$$

Lemme 7 *Pour toutes variables $x \neq y$, et tous termes t,u et v, tels que $x \notin v$:*

$$\left(\frac{\partial t}{\partial x} \cdot u\right)[v/y] = \frac{\partial t[v/y]}{\partial x} \cdot u[v/y].$$

1.2 Réductions

Cette section décrit la réduction du λ-calcul différentiel. On commence par des généralités sur les relations binaires entre termes.

1.2.1 Relations binaires sur les termes

On appelle relation des termes dans les termes toute partie de $\mathbf{N}\langle\Delta\rangle \times \mathbf{N}\langle\Delta\rangle$ et relation des termes simples dans les termes toute partie de $\Delta \times \mathbf{N}\langle\Delta\rangle$.

Définition 4 *Une relation τ des termes dans les termes est* linéaire *si $0\tau 0$ et $t + u = t' + u'$ dès que $t\ \tau\ t'$ et $u\ \tau\ u'$.*

Définition 5 *Une relation τ des termes dans les termes est* contextuelle *si elle est réflexive et linéaire et vérifie : pour tous termes t, u, t', u' tels que $t\ \tau\ t'$ et $u\ \tau\ u'$,*

$$\begin{array}{rcl} \lambda x t & \tau & \lambda x t' \\ (t)\, u & \tau & (t')\, u' \\ \mathrm{D}_i t \cdot u & \tau & \mathrm{D}_i t' \cdot u' \end{array}$$

Lemme 8 *Si τ est une relation contextuelle des termes vers les termes et t, u, u' sont des termes tels que $u\ \tau\ u'$, alors*

$$\begin{array}{rcl} t[u/x] & \tau & t[u'/x] \\ \dfrac{\partial t}{\partial x} \cdot u & \tau & \dfrac{\partial t}{\partial x} \cdot u' \end{array}$$

Preuve: La preuve est facile, par induction sur t. □

Il y a deux façons pertinentes d'étendre une relation des termes simples vers les termes en une relation des termes vers les termes.

Définition 6 *Soit $\tau \subset \Delta \times \mathbf{N}\langle\Delta\rangle$, on définit $\overline{\tau}$ et $\tilde{\tau} \subset \mathbf{N}\langle\Delta\rangle \times \mathbf{N}\langle\Delta\rangle$ par*
- *$t\ \overline{\tau}\ t'$ si $t = \sum_{i=1}^{n} s_i$ et $t' = \sum_{i=1}^{n} {s'}_i$ avec s_i simple et $s_i\ \tau\ s'_i$ pour tout i ;*
- *$t\ \tilde{\tau}\ t'$ si $t = as + u$ et $t' = as' + u$ avec s simple, $s\ \tau\ s'$ et $a \neq 0$.*

$\overline{\tau}$ est la plus petite relation linéaire des termes vers les termes contenant τ, alors que $\tilde{\tau}$ n'est pas linéaire en général.

1.2.2 β-réduction

La β-réduction en un pas (β^1) est la plus petite relation des termes simples dans les termes close par les constructions syntaxiques et telle que :

$$\begin{array}{rcl} (\lambda x s)\, t & \beta^1 & s[t/x] \\ \mathrm{D}_1 \lambda x s \cdot u & \beta^1 & \lambda x \left(\dfrac{\partial s}{\partial x} \cdot u \right) \end{array}$$

Plus formellement :

Définition 7 *On définit la relation β^1 des termes simples vers les termes par induction sur son premier argument :*
- *$\mathrm{D}_{i_1,\ldots,i_n} x \cdot (u_1, \ldots, u_n)\ \beta^1\ u$ si $u = \mathrm{D}_{i_1,\ldots,i_n} x \cdot ({u'}_1, \ldots, {u'}_n)$ avec $u_j\ \beta^1\ u'_j$ pour un $j \in \{1, \ldots, n\}$ et $u_i = u'_i$ pour tout $i \neq j$.*

- $\mathrm{D}_1^n \lambda x s \cdot (u_1, \ldots, u_n)\ \beta^1\ u$ *si* $u = \mathrm{D}_1^n \lambda x s' \cdot (u_1, \ldots, u_n)$ *avec* $s\ \beta^1\ s'$ *ou* $u = \mathrm{D}_1^n \lambda x s \cdot ({u'}_1, \ldots, {u'}_n)$ *avec* $u_j\ \beta^1\ u'_j$ *pour un* $j \in \{1, \ldots, n\}$ *et* $u_i = u'_i$ *pour tout* $i \neq j$ *ou*
$$u = \mathrm{D}_1^{n-1} \lambda x \left(\frac{\partial s}{\partial x} \cdot u_j \right) \cdot (u_1, \ldots, u_{j-1}, u_{j+1}, \ldots, u_n)$$
pour un certain $j \in \{1, \ldots, n\}$.
- $(s)\,t\ \beta^1\ u$ *si* $u = (s')\,t$ *avec* $s\ \beta^1\ s'$ *ou* $u = (s)\,t'$ *avec* $t\ \tilde{\beta}^1\ t'$ *ou* $u = v[t/x]$ *avec* $s = \lambda x v$.

Définition 8 *On définit la relation β comme la clôture réflexive et transitive de $\tilde{\beta}^1$.*

Lemme 9 *La relation β est contextuelle.*

Preuve: Par définition de β^1, on a déjà :
- $(s)\,t\ \beta^1\ (s')\,t$ si $s\ \beta^1\ s'$;
- $(s)\,t\ \beta^1\ (s)\,t'$ si $t\ \tilde{\beta}^1\ t'$;
- $\lambda x s\ \beta^1\ \lambda x s'$ si $s\ \beta^1\ s'$.

On montre facilement par induction sur t que si $u\ \beta^1\ u'$ alors $\mathrm{D}_i t \cdot u\ \beta^1\ \mathrm{D}_i t \cdot u'$. On montre aussi par induction sur t que si $t\ \beta^1\ t'$ alors $\mathrm{D}_i t \cdot u\ \beta^1\ \mathrm{D}_i t' \cdot u$, en faisant une étude des cas possibles pour la réduction. Enfin, la relation β est clairement linéaire, et la linéarité combinée aux propriétés précédentes implique trivialement la contextualité. □

1.2.3 Réduction parallèle

Comme pour le λ-calcul, on montre la confluence en définissant une réduction parallèle qui vérifie la propriété du diamant, et dont β est la clôture réflexive et transitive.

Définition 9 *On définit la réduction parallèle en un pas π des termes simples vers les termes par induction sur son premier argument :*
- $\mathrm{D}_{i_1,\ldots,i_n} x \cdot (u_1, \ldots, u_n)\ \pi\ u$ *si* $u = \mathrm{D}_{i_1,\ldots,i_n} x \cdot ({u'}_1, \ldots, {u'}_n)$ *avec* $u_j\ \pi\ u'_j$ *pour tout* $j \in \{1, \ldots, n\}$.
- $\mathrm{D}_1^n \lambda x s \cdot (u_1, \ldots, u_n)\ \pi\ u$ *si*
$$u = \mathrm{D}_1^{n-p} \lambda x \left(\frac{\partial^p s'}{\partial x^p} \cdot u'_J \right) \cdot u'_{(I \setminus J)}$$
avec $s\ \pi\ s'$, $u_j\ \pi\ u'_j$ *pour tout* $j \in \{1, \ldots, n\}$ *et* $J \subset \{1, \ldots, n\}$ *de cardinal* p.
- $(s)\,t\ \pi\ u$ *dans chacun des cas suivants :*
  - $u = (s')\,t'$ *avec* $s\ \pi\ s'$ *et* $t\ \overline{\pi}\ t'$;
  - $u = \frac{\partial^n v'}{\partial x^n} \cdot ({u'}_1, \ldots, {u'}_n)[t'/x]$ *avec* $s = \mathrm{D}_1^n \lambda x v \cdot (u_1, \ldots, u_n)$, $v\ \pi\ v'$, $u_j\ \pi\ u'_j$ *pour tout* $j \in \{1, \ldots, n\}$ *et* $t\ \overline{\pi}\ t'$.

En particulier, π est réflexive.

Lemme 10 *La relation $\overline{\pi}$ est contextuelle.*

Preuve: Comme pour β. □

Lemme 11 $\beta^1 \subset \pi \subset \beta$. *En particulier, β est la clôture réflexive et transitive de $\overline{\pi}$.*

Preuve: La première inclusion est triviale, et la seconde se déduit facilement de la contextualité de β. □

1.3 Confluence

1.3.1 Réduction et substitutions

Les deux lemmes suivants sont au cœur de la preuve de confluence. Ils se prouvent tous les deux par induction sur t, en faisant une étude des cas possibles pour la réduction de t en t'.

Lemme 12 *Soient x une variable et t, t', u, u' des termes. Si $t \mathrel{\overline{\pi}} t'$ et $u \mathrel{\overline{\pi}} u'$ alors*

$$t[u/x] \mathrel{\overline{\pi}} t'[u'/x].$$

Lemme 13 *Soient x une variable et t, t', u, u' des termes. Si $t \mathrel{\overline{\pi}} t'$ et $u \mathrel{\overline{\pi}} u'$ alors*

$$\frac{\partial t}{\partial x} \cdot u \mathrel{\overline{\pi}} \frac{\partial t'}{\partial x} \cdot u'.$$

1.3.2 Propriété de Church-Rosser pour β

Lemme 14 *$\overline{\pi}$ est confluente.*

Preuve: On prouve en fait la propriété plus forte : si $s \mathrel{\pi} s_i$ pour $1 \leq i \leq n$ alors il existe s' tel que $s_i \mathrel{\overline{\pi}} s'$ pour tout i. On procède par induction sur s en détaillant les cas de réductions possibles. Les lemmes 12 et 13 sont intensivement mis à profit. □

Théorème 15 *β a la propriété de Church-Rosser.*

Preuve: C'est trivialement un corollaire des lemmes 11 et 14. □

1.4 λ-calcul différentiel typé

On peut donner un système de types simples au λ-calcul différentiel, de façon très naturelle et fidèle à l'intuition. On dispose d'une infinité dénombrable de types atomiques $(X, Y, Z, \ldots)$ et on définit les types simples $(A, B, C, \ldots)$ par :

$$A ::= X \mid A \to A$$

Les contextes de typage sont de la forme :

$$\Gamma ::= \emptyset \mid \Gamma, x : A$$

et les jugements de typage s'écrivent

$$\Gamma \vdash s : A$$

Les règles de typage sont données en figure 1. Il est à noter qu'elles sont définies en utilisant les constructeurs généralisés et non par induction sur les termes. Le typage n'est donc pas exactement dirigé par la syntaxe.

La règle intéressante est celle de la différentielle. Elle correspond bien à l'intuition que la dérivée est une fonction linéaire qui renvoie une fonction : $\mathrm{D}_i s \cdot u$ est linéaire en u, et pour u fixé, $\mathrm{D}_i s \cdot u$ est une fonction de même type que s.

Le typage vérifie une propriété d'auto-réduction :

$$\frac{}{\Gamma, x : A \vdash x : A} \qquad \frac{\Gamma, x : A \vdash s : B}{\Gamma \vdash \lambda xs : A \to B} \qquad \frac{\Gamma \vdash s : A \to B \quad \Gamma \vdash t : A}{\Gamma \vdash (s)\,t : B}$$

$$\frac{\Gamma \vdash s : A_1 \to \ldots \to A_i \to B \quad \Gamma \vdash u : A_i}{\Gamma \vdash \mathrm{D}_i s \cdot u : A_1 \to \ldots \to A_i \to B}$$

$$\frac{}{\Gamma \vdash 0 : A} \qquad \frac{\Gamma \vdash s : A \quad \Gamma \vdash t : A}{\Gamma \vdash s + t : A}$$

FIG. 1 – Règles de typage du λ-calcul différentiel.

Théorème 16 *Si* $t\ \beta\ t'$ *et* $\Gamma \vdash t : A$ *alors* $\Gamma \vdash t' : A$.

PREUVE: On prouve l'auto-réduction par induction sur les arbres de typage, à l'aide du résultat suivant :

$$\textit{Si } \Gamma, x : A \vdash s : B \textit{ et } \Gamma \vdash u : A \textit{ alors } \Gamma \vdash s[u/x] : B \textit{ et } \Gamma, x : A \vdash \tfrac{\partial s}{\partial x} \cdot u : B.$$

qui est facilement établi par induction sur s. □

Par ailleurs, Thomas EHRHARD et Laurent RÉGNIER établissent dans [ER03] que les termes typables normalisent fortement pour β. La preuve de ce théorème, qui n'est pas développée ici, utilise une technique classique de réducibilité.

2 Réseaux différentiels

Dans cette section on introduit les réseaux d'interaction différentiels. Une version simple de ces réseaux est présentée dans [ER04] : seuls les réseaux sans boîte promotion sont traités, ce qui correspond au λ-calcul différentiel sans l'application, *i.e.* dans lequel les termes sont seulement de la forme x, λxs ou $\mathrm{D}_i s \cdot u$, et la réduction est la différenciation :

$$\mathrm{D}_1 \lambda xs \cdot u\ \beta\ \lambda x\left(\frac{\partial s}{\partial x} \cdot u\right).$$

2.1 Définition

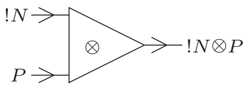
On se donne les connecteurs logiques multiplicatifs ($\otimes$ et $\parr$) et exponentiels (! et ?) de la logique linéaire, et on forme les formules polarisées :

$$\begin{array}{rcl} N & ::= & X \mid ?P \parr N \\ P & ::= & X^{\perp} \mid !N \otimes P \end{array}$$

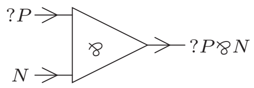
où X est un type atomique. Les formules N sont appelées types de sortie (ce sont des formules négatives) et les formules P types de co-sortie (ce sont des formules positives). On définit aussi des formules d'entrée $?P$ (négatives) et des formules de co-entrée $!N$ (positives). On généralise la négation linéaire par :

$$\begin{array}{rcl} X^{\perp\perp} & = & X \\ (?P \parr N)^{\perp} & = & !P^{\perp} \otimes N^{\perp} \\ (!N \otimes P)^{\perp} & = & ?N^{\perp} \parr P^{\perp} \\ (?P)^{\perp} & = & !P^{\perp} \\ (!N)^{\perp} & = & ?N^{\perp} \end{array}$$

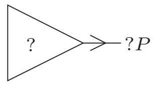
La négation est involutive et échange la polarité des types. On se donne des cellules de la forme :



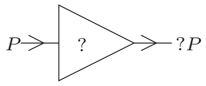
(Tenseur)



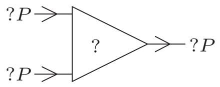
(Par)



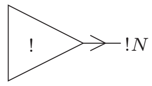
(Affaiblissement)



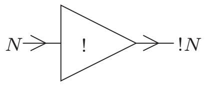
(Déréliction)



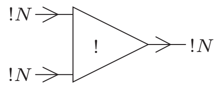
(Contraction)



(Co-affaiblissement)

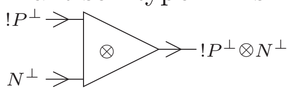


(Co-déréliction)

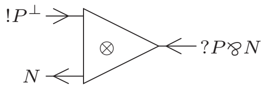


(Co-contraction)

Chaque cellule est munie d'un port principal (la pointe) et de ports auxiliaires (zéro, un ou deux). Chaque port possède une orientation et un type (une formule). On peut changer l'orientation d'un port en niant son type. Ainsi la cellule :

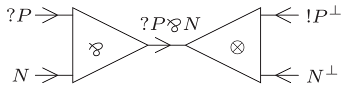


peut s'écrire aussi :



Les ports auxiliaires des cellules $⅋$ et $\otimes$ ne sont pas interchangeables (ils n'ont même pas le même type) ; en revanche, les ports auxilaires des cellules de contraction et de co-contraction le sont.

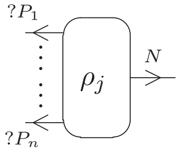
Un réseau différentiel simple de profondeur zéro est un graphe construit sur un nombre fini de cellules, en reliant les ports par des fils. Chaque port est connecté à exactement un fil. Par contre il peut y avoir des fils connectés à un seul port, voire à aucun. Chaque fil possède une orientation et un type. On impose que les types des fils s'accordent avec les types des ports (là encore, on peut changer l'orientation d'un fil en niant son type). Par exemple, la connection suivante est possible :



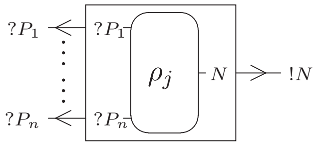
Le type d'un réseau simple est donné par les types des fils pendants (*i.e.* les extrêmités non-connectées) orientés vers l'extérieur. Ainsi, le petit réseau précédent a le type : $!P^\perp, N^\perp, ?P, N$.

Si on a n réseaux simples $\rho_1, \ldots, \rho_n$ on peut former leur somme (*i.e.* le multi-ensemble correspondant) dès qu'ils ont le même type. En particulier, le réseau nul (*i.e.* le multi-ensemble vide) a tous les types. Les sommes de réseaux simples sont appelées réseaux généraux, ou plus simplement réseaux.

Il reste à introduire les réseaux de profondeur non nulle. On ajoute la construction qui consiste à clore un réseau de profondeur au plus k dans une boîte, dès que ce réseau a un type de la forme $?P_1, \ldots, ?P_n, N$. La boîte ainsi formée est une nouvelle cellule dont les ports, orientés vers l'extérieur ont les types $?P_1, \ldots, ?P_n$ et $!N$. Ainsi, à partir du réseau :



on forme la boîte :



Cette opération est appelée promotion.

Supposons qu'on a défini les réseaux simples de hauteur au plus k : ils forment l'ensemble $\mathcal{R}_k$. Les éléments de $\mathcal{R}_{k+1}$ sont les réseaux simples formés en ajoutant les boîtes contenant des réseaux de $\mathbf{N}\langle\mathcal{R}_k\rangle$ aux cellules vues plus haut. On note $\mathcal{R} = \bigcup_k \mathcal{R}_k$ l'ensemble des réseaux simples de hauteur non bornée, et on appelle réseaux (généraux) les éléments de $\mathbf{N}\langle\mathcal{R}\rangle = \bigcup_k \mathbf{N}\langle\mathcal{R}_k\rangle$.

On considèrera que la (co-)contraction est associative et commutative et admet l'affaiblissement (resp. le co-affaiblissement) comme élément neutre. Par ailleurs, les contractions et les affaiblissements peuvent entrer et sortir des boîtes sans changer le réseau. Ces équations sont formalisées en figure 2.

2.2 Connection de deux réseaux

Soient deux réseaux simples ρ et ρ' tels que le type de ρ soit de la forme $A_1, \ldots, A_p$ et celui de ρ' $A'_1, \ldots, A'_r$ avec $A_i = {A'}_i^{\perp}$ pour $1 \leq i \leq n$. On peut connecter ρ et ρ' en identifiant les fils pendants correspondants[5]. Le réseau simple ainsi obtenu a le type $A_{n+1}, \ldots, A_p, A'_{n+1}, \ldots, A'_r$.

La connection de deux réseaux généraux se définit par bilinéarité : la connection de $\rho = \sum_{i=1}^{n} \rho_i$ avec $\rho = \sum_{j=1}^{m} \rho'_j$ est la somme des $n \times m$ réseaux simples obtenus en connectant ρ_i avec ρ'_j pour tout couple (i, j).

Il est clair que tout réseau peut s'obtenir à partir des cellules par connection de réseaux et promotion.

2.3 Réduction des réseaux différentiels

On définit la réduction $\rightsquigarrow$, qui est une relation des réseaux simples vers réseaux. On se donne les couples rédex–réduit des figures 3 et 4. Si le réseau simple ρ contient un rédex ρ_{x}, alors $\rho \rightsquigarrow \rho'$ où ρ' est le réseau (général) obtenu en connectant le réduit ρ_{t} correspondant à la place de ρ_{x}. On s'autorise à réduire dans les boîtes. La réduction sur les réseaux généraux est donnée par :

$$\sum_{i=1}^{n} \rho \rightsquigarrow \sum_{i=1}^{n} \rho'$$

[5]Ce concept est en fait plus compliqué qu'il n'y paraît. En effet, il se peut que l'on en vienne à former des boucles (*i.e.* à identifier les deux extrêmités d'un fil). Une solution consiste à imposer un critère de correction qui empêche la formation de telles boucles lors des réductions. Toutefois, dans le cadre informel de cette présentation, on ferme les yeux sur ce problème important.

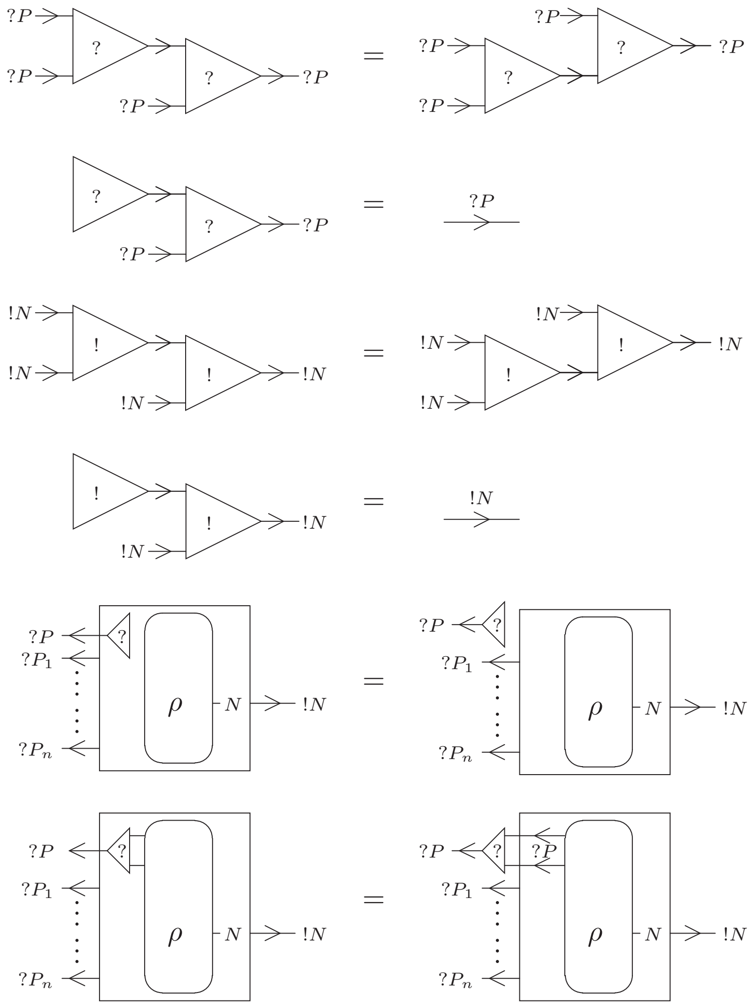


Fig. 2 – Égalité structurelle des réseaux.

si pour tout i, ρ_i est un réseau simple et il existe un j tel que $\rho_j \rightsquigarrow \rho'_j$ et, pour tout $i \neq j$, $\rho_i = \rho'_i$. En particulier, le réseau nul est en forme normale.

Pour une justification « morale » du choix des réductions de la figure 3, on pourra se reporter à [ER04]. Disons simplement qu'un réseau de type $N_1^\perp, \ldots, N_p^\perp, ?N_{p+1}^\perp, \ldots, ?N_n^\perp, N$ représente une fonction analytique à n variables de types N_1 à N_n dont le résultat est de type N, et qui est linéaire en ses p premières variables. La boîte sert à faire des copies de réseaux (lorsqu'elle est réduite sur une contraction) ou à les effacer (lorsqu'elle est réduite sur un affaiblissement), et la déreliction sert à casser la boîte (quand, à cet endroit du réseau, on a besoin d'exactement une copie du contenu de la boîte : retour à la linéarité). Le $\parr$ sert à fabriquer une fonction ($?N_1^\perp \parr N_2$ est le type des fonctions analytiques de N_1 dans N_2) et le tenseur à appliquer une fonction à un argument. Les cellules co-affaiblissement, co-déreliction et co-contraction prennent alors un sens particulier : la co-déréliction représente la dérivation en 0 (quand on la branche sur un réseau de type $N_1^\perp, \ldots, N_p^\perp, ?N_{p+1}^\perp, \ldots, ?N_n^\perp, N$ pour obtenir un réseau de type $N_1^\perp, \ldots, N_{p+1}^\perp, ?N_{p+2}^\perp, \ldots, ?N_n^\perp, N$, on dérive en zéro par rapport à la p-ème variable) la co-contraction représente la somme et le co-affaiblissement représente 0. Avec ces quelques suggestions, le lecteur courageux parviendra peut-être à se convaincre que les réductions des figures 3 et 4 traduisent des équations bien connues sur les dérivées partielles de fonctions infiniment différenciables.

La réduction des réseaux différentiels est confluente[6]. On note $\equiv$ sa côture réflexive, symétrique et transitive.

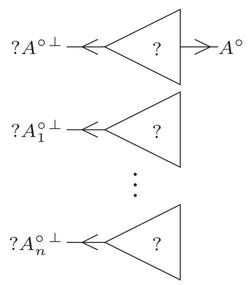
2.4 Traduction du λ-calcul différentiel

De la même façon que le λ-calcul se traduit dans les réseaux de preuve de la logique linéaire, les réseaux différentiels sont un bon formalisme pour traduire le λ-calcul différentiel et sa réduction de façon graphique.

On se place dans le cadre du λ-calcul différentiel simplement typé, et on définit une traduction des jugements de typage du λ-calcul différentiel dans les réseaux par induction sur les arbres de typage, telle qu'un jugement de la forme $x_1 : A_1, \ldots, x_n : A_n \vdash s : A$ est traduit par un réseau de type $?A^\circ{}_1^\perp, \ldots, ?A^\circ{}_n^\perp, A^\circ$, où A° est la traduction négative de A :

$$
\begin{array}{rcl}
X^\circ & = & X \\
(A \to B)^\circ & = & ?A^{\circ\perp} \parr B^\circ
\end{array}
$$

$\overline{\Gamma, x : A \vdash x : A}$ est traduit par :



[6]La confluence n'est pas aussi simple à établir que dans [ER04], de par la présence de boîtes. Il faut comme dans le cas du calcul considérer une réduction parallèle qui a la même clôture et prouver la propriété du diamant pour cette réduction. Cependant, cette dernière preuve est très facile, car la réduction sur les réseaux est définie de façon complètement locale.

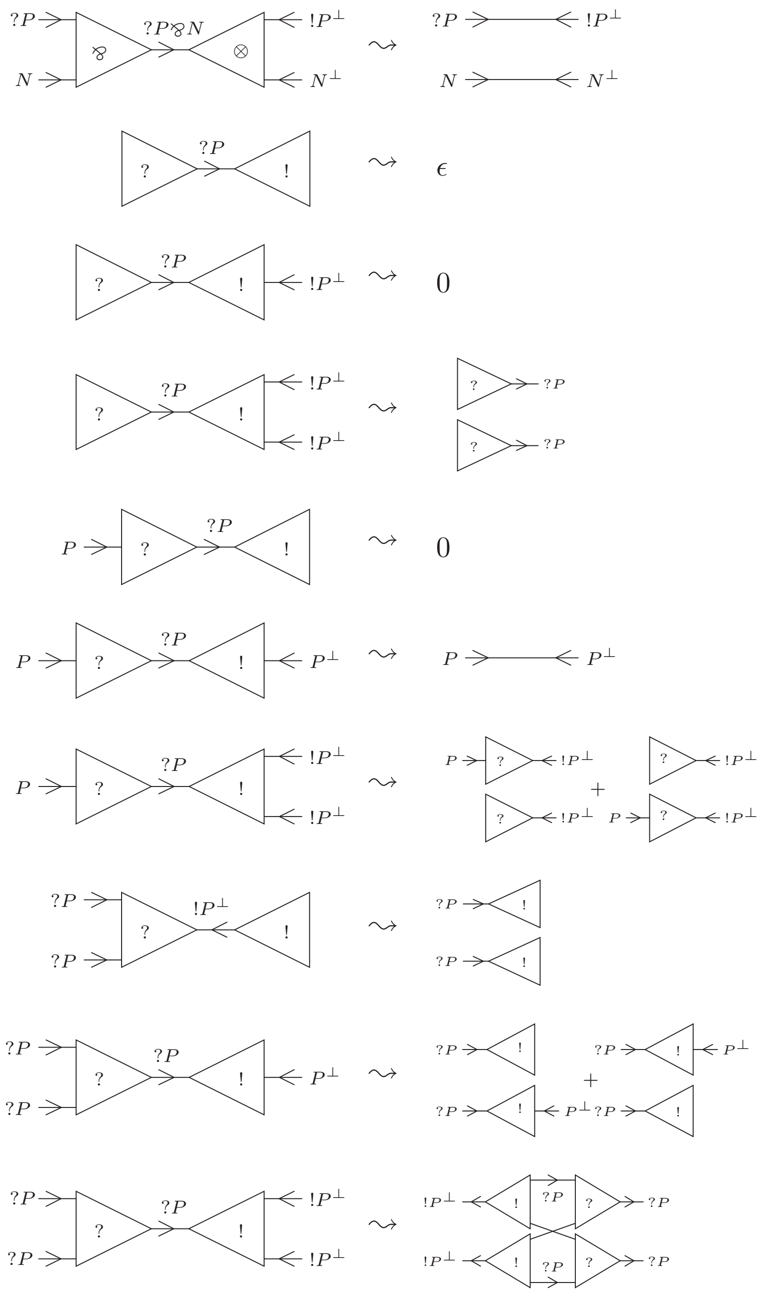


FIG. 3 – Paires rédex/réduit pour les réseaux différentiels simples : cellules un port.

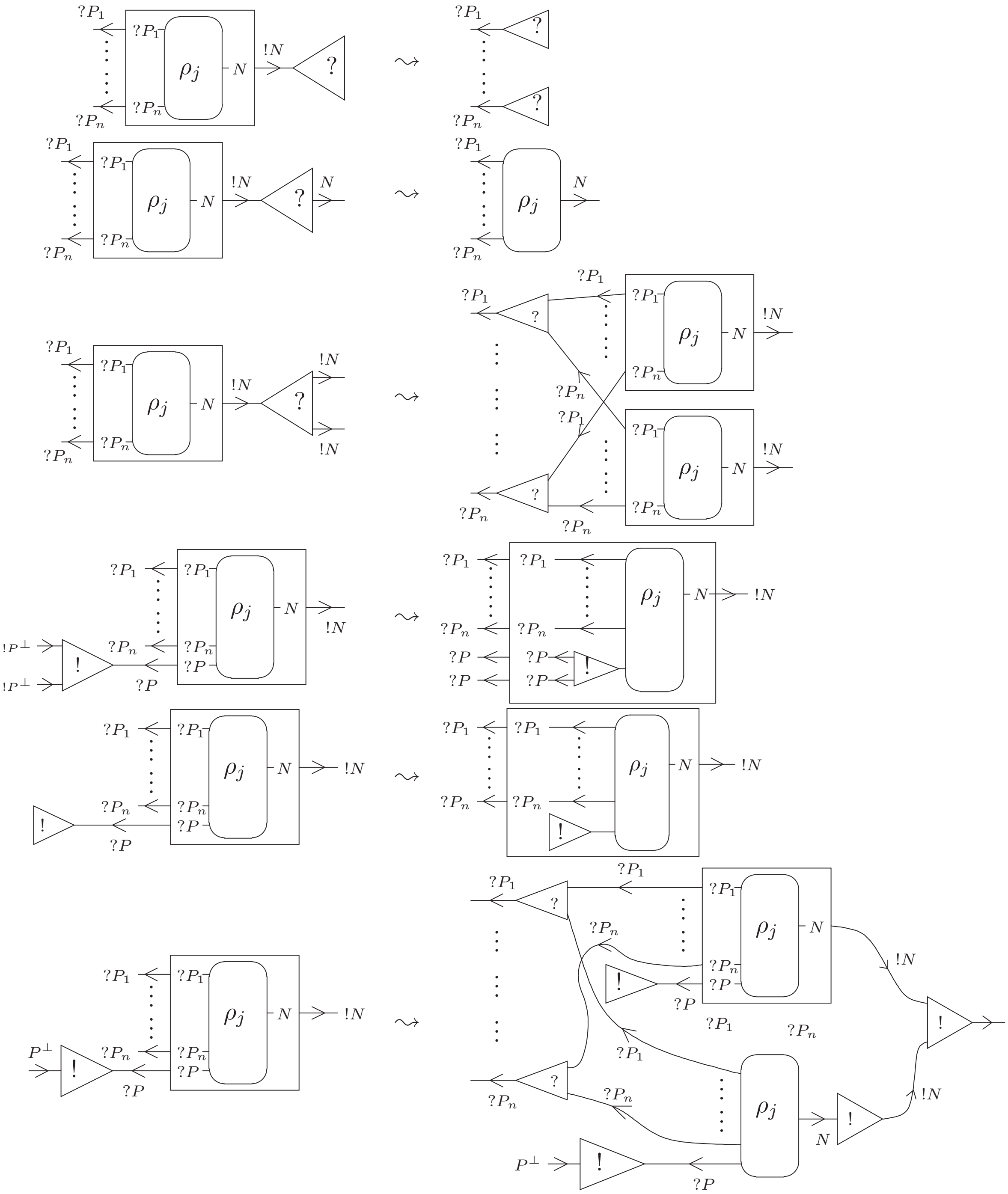
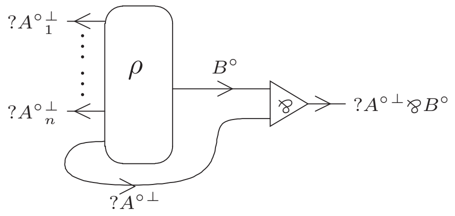


FIG. 4 – Paires rédex/réduit pour les réseaux différentiels simples : boîtes.

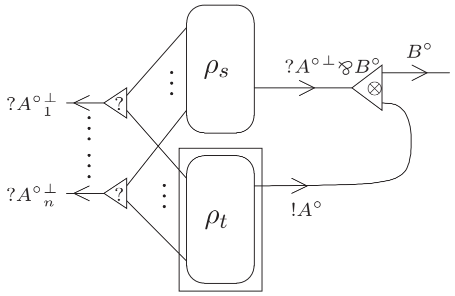
avec $\Gamma = x_1 : A_1, \ldots, x_n : A_n$.

$\dfrac{\Gamma, x : A \vdash s : B}{\Gamma \vdash \lambda x s : A \to B}$ est traduit par :



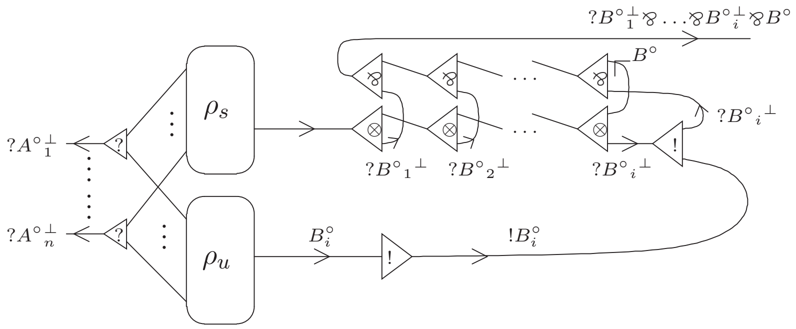
si $\Gamma, x : A \vdash s : B$ est traduit par ρ.

$\dfrac{\Gamma \vdash s : A \to B \quad \Gamma \vdash t : A}{\Gamma \vdash (s)\, t : B}$ est traduit par :



si $\Gamma \vdash s : A \to B$ est traduit par ρ_s et $\Gamma \vdash t : A$ est traduit par ρ_t.

$\dfrac{\Gamma \vdash s : B_1 \to \ldots \to B_i \to B \quad \Gamma \vdash u : B_i}{\Gamma \vdash \mathrm{D}_i s \cdot u : B_1 \to \ldots \to B_i \to C}$ est traduit par :



si $\Gamma \vdash s : B_1 \to \ldots \to B_i \to B$ est traduit par ρ_s et $\Gamma \vdash u : B_i$ est traduit par ρ_u. La suite de portes ⅋ et $\otimes$ correspond à une eta-expansion qui fait passer la i-ème abstraction de s en tête. La différentielle à proprement parler correspond aux portes co-contraction et co-déréliction.

$\overline{\Gamma \vdash 0 : A}$ est traduit par le réseau nul, et $\dfrac{\Gamma \vdash s : A \quad \Gamma \vdash t : A}{\Gamma \vdash s + t : A}$ est traduit par la somme des traductions de $\Gamma \vdash s : A$ et $\Gamma \vdash t : A$.

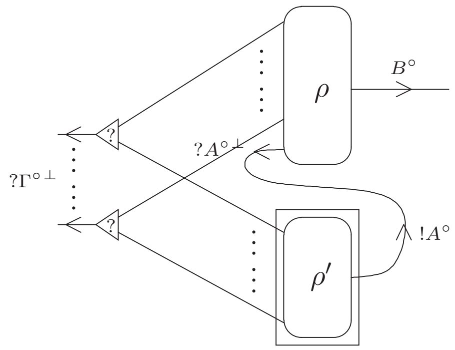
2.5 Correction de la traduction

La traduction du λ-calcul différentiel dans les réseaux différentiels est correcte, c'est à dire que :

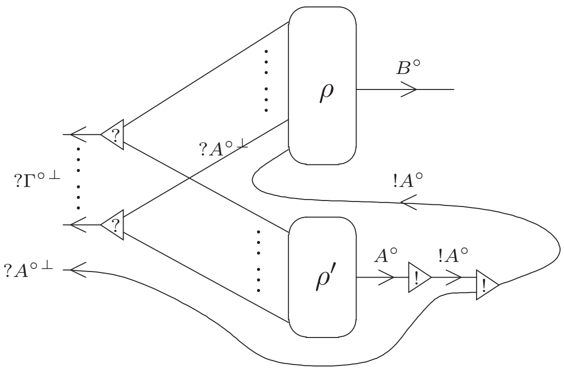
Théorème 17 *Si $\Gamma \vdash t : A$ est traduit par ρ et $t \beta t'$ alors $\Gamma \vdash t' : A$ est traduit par un ρ' tel que $\rho \equiv \rho'$.*

PREUVE:La preuve se fait par induction sur les arbres de dérivation, à l'aide des deux résultats suivants :

(i) Si ρ traduit $\Gamma, x : A \vdash s : B$ et ρ' traduit $\Gamma \vdash t : A$, alors $\Gamma \vdash s[t/x]$ est traduit par :



(ii) Si ρ traduit $\Gamma, x : A \vdash s : B$ et ρ' traduit $\Gamma \vdash u : A$, alors $\Gamma, x : A \vdash \frac{\partial s}{\partial x} \cdot u$ est traduit par :



□

3 Le $\lambda\mu$-calcul différentiel

Cette partie présente une extension classique du λ-calcul différentiel, résultat d'une hybridation avec le $\lambda\mu$-calcul. On commence par rappeler la définition du $\lambda\mu$-calcul de Michel PARIGOT [Par92], puis on reproduit la structure de la première partie.

3.1 Le $\lambda\mu$-calcul

On se donne une infinité dénombrable de μ-variables, ou noms : $\alpha, \beta, \gamma, \ldots$, et on ajoute aux constructions de termes du λ-calcul la μ-abstraction : $\mu\alpha[\beta]t$. $[\beta]t$ est un terme nommé. La réduction du $\lambda\mu$-calcul ajoute à la β-réduction usuelle la règle suivante :

$$(\mu\alpha\nu)\, t = \mu\alpha(\nu[t@\alpha])$$

où le terme nommé $\nu[t@\alpha]$ est obtenu à partir de ν en applicant t à tous les sous-termes nommés α de ν. Ainsi, si $\alpha \neq \beta$,

$$([\alpha]\lambda x\mu\beta[\alpha]x)[y@\alpha] = [\alpha]((\lambda x\mu\beta[\alpha](x)\, y)\, y)$$

On peut donner un système de types simples au $\lambda\mu$-calcul tel que les termes typés du $\lambda\mu$-calcul sont isomorphes aux preuves en déduction naturelle classique. On utilise un

$$\frac{}{\Gamma, x : A \vdash x : A|\Delta} \qquad \frac{\Gamma, x : A \vdash s : B|\Delta}{\Gamma \vdash \lambda x s : A \to B|\Delta} \qquad \frac{\Gamma \vdash s : A \to B|\Delta \quad \Gamma \vdash t : A|\Delta}{\Gamma \vdash (s)\, t : B|\Delta}$$

$$\frac{\Gamma \vdash s : B|\alpha : A, \Delta \quad (\beta : B) \in \alpha : A, \Delta}{\Gamma \vdash \mu\alpha[\beta]s : A|\Delta}$$

FIG. 5 – Règles de typage du $\lambda\mu$-calcul.

contexte de variables (Γ) et un contexte de noms (Δ). Les règles de typage du $\lambda\mu$-calcul sont données en figure 5. Par exemple, on a un terme typé par la formule de Pierce :

$$\vdash \lambda f \mu\alpha[\alpha](f)\, \lambda x \mu\beta[\alpha]x : ((A \to B) \to A) \to A|$$

Le $\lambda\mu$-calcul est confluent et les termes typés sont fortement normalisables.

3.2 Syntaxe du $\lambda\mu$-calcul différentiel

On étend le $\lambda\mu$-calcul, comme on a étendu le λ-calcul pour obtenir le λ-calcul différentiel, en introduisant un opérateur de différentielle. La définition formelle du $\lambda\mu$-calcul différentiel suit.

3.2.1 Syntaxe de base

On suppose donnés deux ensembles infinis dénombrables distincts, l'un de variables $(x, y, z, \ldots)$ et l'autre de noms $(\alpha, \beta, \gamma, \ldots)$.

Définition 10 *On définit par induction sur l'entier k l'ensemble Δ_k des termes simples et l'ensemble Υ_k des termes simples nommés, de taille inférieure ou égale à k. On fixe $\Delta_0 = \Upsilon_0 = \emptyset$ et on construit Δ_{k+1} et Υ_{k+1} à partir de Δ_k et Υ_k comme suit.*

- *Si $t \in \Delta_k$ alors $t \in \Delta_{k+1}$.*
- *Si x est une variable, $n \in \mathbf{N}$, $i_1, \ldots, i_n \in \mathbf{N}^*$ et $u_1, \ldots, u_n \in \Delta_k$ alors*

  $$\mathrm{D}_{i_1,\ldots,i_n} x \cdot (u_1, \ldots, u_n) \in \Delta_{k+1}.$$

  Ce terme est identifié avec tous les termes de la forme

  $$\mathrm{D}_{i_{\sigma(1)},\ldots,i_{\sigma(n)}} x \cdot (u_{\sigma(1)}, \ldots, u_{\sigma(n)})$$

  où σ est une permutation de $\{1, \ldots, n\}$.
- *Si α est un nom, $n \in \mathbf{N}$, $i_1, \ldots, i_n \in \mathbf{N}^*$, $\nu \in \Upsilon_k$ et $u_1, \ldots, u_n \in \Delta_k$ alors*

  $$\mathrm{D}_{i_1,\ldots,i_n} \mu\alpha\nu \cdot (u_1, \ldots, u_n) \in \Delta_{k+1}.$$

  Ce terme est identifié avec tous les termes de la forme

  $$\mathrm{D}_{i_{\sigma(1)},\ldots,i_{\sigma(n)}} \mu\alpha\nu \cdot (u_{\sigma(1)}, \ldots, u_{\sigma(n)})$$

  où σ est une permutation de $\{1, \ldots, n\}$.

– *Si x est une variable, $n \in \mathbf{N}$ et $s, u_1, \ldots, u_n \in \Delta_k$ alors*

$$\mathrm{D}_1^n \lambda x s \cdot (u_1, \ldots, u_n) \in \Delta_{k+1}.$$

Ce terme est identifié avec tous les termes de la forme

$$\mathrm{D}_1^n \lambda x s \cdot (u_{\sigma(1)}, \ldots, u_{\sigma(n)})$$

où σ est une permutation de $\{1, \ldots, n\}$.

– *Si $s \in \Delta_k$ et $t \in \mathbf{N}\langle \Delta_k \rangle$ alors*

$$(s)\, t \in \Delta_{k+1}.$$

– *Si α est un nom et $s \in \Delta_{k+1}$ alors $[\alpha]s \in \Upsilon_{k+1}$.*

On note $\Delta = \bigcup_{k \in \mathbf{N}} \Delta_k$ l'ensemble des termes simples. On appelle termes généraux, ou plus simplement termes, les éléments de $\bigcup_{k \in \mathbf{N}} \mathbf{N}\langle \Delta_k \rangle = \mathbf{N}\langle \Delta \rangle$.

3.2.2 Extension de la syntaxe

On généralise les constructions du calcul à tous les termes de la façon suivante.

Différentielle On construit l'opérateur de différentielle généralisé pour s et u des termes simples par induction sur s :

$$\begin{array}{rcl}
\mathrm{D}_i(\mathrm{D}_{i_1,\ldots,i_n} x \cdot (u_1, \ldots, u_n)) \cdot v & = & \mathrm{D}_{i,i_1,\ldots,i_n} x \cdot (v, u_1, \ldots, u_n) \\
\mathrm{D}_i(\mathrm{D}_{i_1,\ldots,i_n} \mu\alpha\nu \cdot (u_1, \ldots, u_n)) \cdot v & = & \mathrm{D}_{i,i_1,\ldots,i_n} \mu\alpha\nu \cdot (v, u_1, \ldots, u_n) \\
\mathrm{D}_i(\mathrm{D}_1^n \lambda x s \cdot (u_1, \ldots, u_n)) \cdot v & = & \left\{ \begin{array}{ll} \mathrm{D}_1^{n+1} \lambda x s \cdot (v, u_1, \ldots, u_n) & \text{si } i = 1 \\ \mathrm{D}_1^n \lambda x (\mathrm{D}_{i-1} s \cdot v) \cdot (u_1, \ldots, u_n) & \text{sinon} \end{array} \right. \\
\mathrm{D}_i(s)\, t \cdot u & = & (\mathrm{D}_{i+1} s \cdot u)\, t
\end{array}$$

On étend cette construction au cas où s et u sont des termes généraux par bilinéarité :

$$\mathrm{D}_i \left(\sum_{s \in \Delta} a_s s \right) \cdot \left(\sum_{u \in \Delta} b_u u \right) = \sum_{s,u \in \Delta} a_s b_u \, (\mathrm{D}_i s \cdot u)$$

Linéarité des positions fonctionnelles On étend les autres constructeurs du calcul par linéarité :

$$\begin{array}{rcl}
\lambda x \left(\sum_{s \in \Delta} a_s s \right) & = & \sum_{s \in \Delta} a_s \lambda x s \\
\mu\alpha \left(\sum_{\nu \in \Upsilon} a_\nu \nu \right) & = & \sum_{\nu \in \Upsilon} a_\nu \mu\alpha\nu \\
[\alpha] \left(\sum_{s \in \Delta} a_s s \right) & = & \sum_{s \in \Delta} [\alpha] s \\
\left(\sum_{s \in \Delta} a_s s \right) t & = & \sum_{s \in \Delta} a_s (s)\, t
\end{array}$$

Comme pour le λ-calcul différentiel, il est clair qu'on peut construire tous les termes du calcul en utilisant seulement les variables et les opérateurs généralisés, et il est aisé d'établir que pour tout s, $\mathrm{D}_i(\mathrm{D}_j s \cdot v) \cdot u = \mathrm{D}_j(\mathrm{D}_i s \cdot u) \cdot v$.

3.2.3 Substitutions

Le $\lambda\mu$-calcul différentiel comporte quatre substitutions différentes : la substitution du λ-calcul qui remplace toutes les occurences libres d'une variable par un terme, l'application nommée du $\lambda\mu$-calcul qui applique un terme à tous les sous-termes portant un nom donné, et une variante linéaire de chacune de ces substitutions.

Définition 11 (λ-substitution) *On définit par induction sur s la λ-substitution $s[t/x]$:*

$$
\begin{array}{rcl}
(\mathrm{D}_{i_1,\ldots,i_n} y \cdot (u_1,\ldots,u_n))[t/x] & = & \left\{ \begin{array}{ll} \mathrm{D}_{i_1,\ldots,i_n} t \cdot (u_1[t/x],\ldots,u_n[t/x]) & si\ x = y \\ \mathrm{D}_{i_1,\ldots,i_n} y \cdot (u_1[t/x],\ldots,u_n[t/x]) & si\ x \neq y \end{array} \right. \\
(\mathrm{D}_{i_1,\ldots,i_n} \mu\beta\nu \cdot (u_1,\ldots,u_n))[t/x] & = & \mathrm{D}_{i_1,\ldots,i_n} \mu\beta\nu[t/x] \cdot (u_1[t/x],\ldots,u_n[t/x]) \\
([\beta]s)[t/x] & = & [\beta](s[t/x]) \\
(\mathrm{D}_1^n \lambda y s \cdot (u_1,\ldots,u_n))[t/x] & = & \mathrm{D}_1^n \lambda y s[t/x] \cdot (u_1[t/x],\ldots,u_n[t/x]) \ \ avec\ x \neq y \\
((s)\,w)[t/x] & = & (s[t/x])\,w[t/x] \\
(\sum_{s\in\Delta} a_s s)[t/x] & = & \sum_{s\in\Delta} a_s s[t/x]
\end{array}
$$

Définition 12 (μ-application) *On définit par induction sur s la μ-application $s[t@\alpha]$:*

$$
\begin{array}{rcl}
(\mathrm{D}_{i_1,\ldots,i_n} y \cdot (u_1,\ldots,u_n))[t@\alpha] & = & \mathrm{D}_{i_1,\ldots,i_n} y \cdot (u_1[t@\alpha],\ldots,u_n[t@\alpha]) \\
(\mathrm{D}_{i_1,\ldots,i_n} \mu\beta\nu \cdot (u_1,\ldots,u_n))[t@\alpha] & = & \mathrm{D}_{i_1,\ldots,i_n} \mu\beta\nu[t@\alpha] \cdot (u_1[t@\alpha],\ldots,u_n[t@\alpha]) \ \ avec\ \alpha \neq \beta \\
([\beta]s)[t@\alpha] & = & \left\{ \begin{array}{ll} [\alpha]((s[t@\alpha])\,t) & si\ \alpha = \beta \\ [\beta](s[t@\alpha]) & si\ \alpha \neq \beta \end{array} \right. \\
(\mathrm{D}_1^n \lambda y s \cdot (u_1,\ldots,u_n))[t@\alpha] & = & \mathrm{D}_1^n \lambda y s[t@\alpha] \cdot (u_1[t@\alpha],\ldots,u_n[t@\alpha]) \\
((s)\,w)[t@\alpha] & = & (s[t@\alpha])\,w[t@\alpha] \\
(\sum_{s\in\Delta} a_s s)[t@\alpha] & = & \sum_{s\in\Delta} a_s s[t@\alpha]
\end{array}
$$

Définition 13 (λ-dérivation) *On définit par induction sur s la λ-dérivation $\frac{\partial s}{\partial x} \cdot t$:*

$$
\begin{array}{rcl}
\dfrac{\partial(\mathrm{D}_{i_1,\ldots,i_n} y \cdot (u_1,\ldots,u_n))}{\partial x} \cdot t & = & \delta_{x=y} \mathrm{D}_{i_1,\ldots,i_n} t \cdot (u_1,\ldots,u_n) \\
& & + \sum_{1\leq j\leq n} \mathrm{D}_{i_1,\ldots,i_n} y \cdot \left(u_1,\ldots,\dfrac{\partial u_j}{\partial x} \cdot t,\ldots,u_n\right) \\
\dfrac{\partial(\mathrm{D}_{i_1,\ldots,i_n} \mu\beta\nu \cdot (u_1,\ldots,u_n))}{\partial x} \cdot t & = & \mathrm{D}_{i_1,\ldots,i_n} \mu\beta\left(\dfrac{\partial \nu}{\partial x} \cdot t\right) \cdot (u_1,\ldots,u_n) \\
& & + \sum_{1\leq j\leq n} \mathrm{D}_{i_1,\ldots,i_n} \mu\beta\nu \cdot \left(u_1,\ldots,\dfrac{\partial u_j}{\partial x} \cdot t,\ldots,u_n\right) \\
\dfrac{\partial[\beta]s}{\partial x} \cdot t & = & [\beta]\left(\dfrac{\partial s}{\partial x} \cdot t\right) \\
\dfrac{\partial(\mathrm{D}_1^n \lambda y s \cdot (u_1,\ldots,u_n))}{\partial x} \cdot t & = & \mathrm{D}_1^n \lambda y\left(\dfrac{\partial s}{\partial x} \cdot t\right) \cdot (u_1,\ldots,u_n) \\
& & + \sum_{1\leq j\leq n} \mathrm{D}_1^n \lambda y s \cdot \left(u_1,\ldots,\dfrac{\partial u_j}{\partial x} \cdot t,\ldots,u_n\right) \\
\dfrac{\partial(s)\,w}{\partial x} \cdot t & = & \left(\dfrac{\partial s}{\partial x} \cdot t\right) w + \left(\mathrm{D}_1 s \cdot \left(\dfrac{\partial w}{\partial x} \cdot t\right)\right) w
\end{array}
$$

$$\frac{\partial \sum_{s\in\Delta} a_s s}{\partial x} \cdot t = \sum_{s\in\Delta} a_s \left(\frac{\partial s}{\partial x}\cdot t\right)$$

Définition 14 (μ-différentielle) *On définit par induction sur s la μ-différentielle $\frac{\delta_k s}{\delta\alpha}\cdot t$:*

$$\begin{aligned}
\frac{\delta_k(\mathrm{D}_{i_1,\ldots,i_n} y\cdot(u_1,\ldots,u_n))}{\delta\alpha}\cdot t &= \sum_{1\le j\le n} \mathrm{D}_{i_1,\ldots,i_n} y\cdot\left(u_1,\ldots,\frac{\delta_k u_j}{\delta\alpha}\cdot t,\ldots,u_n\right)\\
\frac{\delta_k(\mathrm{D}_{i_1,\ldots,i_n}\mu\beta\nu\cdot(u_1,\ldots,u_n))}{\delta\alpha}\cdot t &= \mathrm{D}_{i_1,\ldots,i_n}\mu\beta\left(\frac{\delta_k\nu}{\delta\alpha}\cdot t\right)\cdot(u_1,\ldots,u_n)\\
&\quad+\sum_{1\le j\le n}\mathrm{D}_{i_1,\ldots,i_n}\mu\beta\nu\cdot\left(u_1,\ldots,\frac{\delta_k u_j}{\delta\alpha}\cdot t,\ldots,u_n\right)\\
\frac{\delta_k[\beta]s}{\delta\alpha}\cdot t &= \delta_{\alpha=\beta}[\alpha](\mathrm{D}_k s\cdot t)+[\beta]\left(\frac{\delta_k s}{\delta\alpha}\cdot t\right)\\
\frac{\delta_k(\mathrm{D}_1^n\lambda ys\cdot(u_1,\ldots,u_n))}{\delta\alpha}\cdot t &= \mathrm{D}_1^n\lambda y\frac{\delta_k s}{\delta\alpha}\cdot t\cdot(u_1,\ldots,u_n)\\
&\quad+\sum_{1\le j\le n}\mathrm{D}_1^n\lambda ys\cdot\left(u_1,\ldots,\frac{\delta_k u_j}{\delta\alpha}\cdot t,\ldots,u_n\right)\\
\frac{\delta_k(s)\,w}{\delta\alpha}\cdot t &= \left(\frac{\delta_k s}{\delta\alpha}\cdot t\right)w+\left(\mathrm{D}_1 s\cdot\frac{\delta_k w}{\delta\alpha}\cdot t\right)w\\
\frac{\delta_k\sum_{s\in\Delta}a_s s}{\delta\alpha}\cdot t &= \sum_{s\in\Delta}a_s\frac{\delta_k s}{\delta\alpha}\cdot t
\end{aligned}$$

C'est cette dernière définition qui est intéressante. La différentielle est le pendant linéaire de l'application. Si la μ-application $s[v@\alpha]$ consiste à appliquer v à tous les sous-termes de s nommé par α, l'opération linéaire correspondante consiste à différencier *l'un* des sous-termes de s nommés par α. Ainsi $\frac{\delta_i s}{\delta\alpha}\cdot u$ est la somme des termes obtenus en différenciant selon u et par rapport à sa i-ème abstraction chaque sous terme de s nommé par α.

Les lemmes suivants correspondent aux lemmes 1 à 7 pour le λ-calcul différentiel. Les résultats à établir sont plus nombreux à cause du plus grand nombre de substitutions, mais les preuves, pour fastidieuses qu'elles soient, restent faciles.

Différentielle et substitutions

Lemme 18 *Pour tous $i,j\in\mathbf{N}^+$, toute variable x, tout nom α et tous termes s,u et v, on a :*

$$\begin{aligned}
(\mathrm{D}_i s\cdot u)[v/x] &= \mathrm{D}_i s[v/x]\cdot u[v/x]\\
(\mathrm{D}_i s\cdot u)[v@\alpha] &= \mathrm{D}_i s[v@\alpha]\cdot u[v@\alpha]\\
\frac{\partial}{\partial x}(\mathrm{D}_i s\cdot u)\cdot v &= \mathrm{D}_i\left(\frac{\partial s}{\partial x}\cdot v\right)\cdot u+\mathrm{D}_i s\cdot\left(\frac{\partial u}{\partial x}\cdot v\right)\\
\frac{\delta_j}{\delta\alpha}(\mathrm{D}_i s\cdot u)\cdot v &= \mathrm{D}_i\left(\frac{\delta_j s}{\delta\alpha}\cdot v\right)\cdot u+\mathrm{D}_i s\cdot\left(\frac{\delta_j u}{\delta\alpha}\cdot v\right)
\end{aligned}$$

Linéarité des dérivations

Lemme 19 *Pour toute variable x et tous termes s et u, si $x \notin s$ alors $\frac{\partial s}{\partial x} \cdot u = 0$.*

Lemme 20 *Pour toute variable x et tous termes $t, u_1, \ldots, u_n$, on a :*

$$\frac{\partial t}{\partial x} \cdot \left(\sum_{j=1}^{n} u_j\right) = \sum_{j=1}^{n} \frac{\partial t}{\partial x} \cdot u_j.$$

Lemme 21 *Pour tout $i \in \mathbf{N}^*$, tout nom α et tous termes s et u, si $\alpha \notin s$ alors $\frac{\delta_i s}{\delta \alpha} \cdot u = 0$.*

Lemme 22 *Pour tout $i \in \mathbf{N}^*$, tout nom α et tous termes $t, u_1, \ldots, u_n$, on a :*

$$\frac{\delta_i t}{\delta \alpha} \cdot \left(\sum_{j=1}^{n} u_j\right) = \sum_{j=1}^{n} \frac{\delta_i t}{\delta \alpha} \cdot u_j.$$

Commutations des substitutions

Lemme 23 *Pour tout $i, j \in \mathbf{N}^+$, toutes variables x et y, tous noms α et β et tous termes t,u et v, on a les résultats suivants.*
Si $x \neq y$ et $x \notin v$ alors

$$t[u/x][v/y] = t[v/y][u[v/y]/x]$$

Si $x \notin v$

$$t[u/x][v@\alpha] = (t[v@\alpha])[u[v@\alpha]/x]$$

Si $x \neq y$ et $x \notin u \cup v$

$$\frac{\partial t[u/x]}{\partial y} \cdot v = \left(\frac{\partial t}{\partial y} \cdot v\right)[u/x] + \left(\frac{\partial t}{\partial x} \cdot \left(\frac{\partial u}{\partial y} \cdot v\right)\right)[u/x]$$

Si $x \notin u \cup v$

$$\frac{\delta_i t[u/x]}{\delta \alpha} \cdot v = \left(\frac{\delta_i t}{\delta \alpha} \cdot v\right)[u/x] + \left(\frac{\partial t}{\partial x} \cdot \left(\frac{\delta_i u}{\delta \alpha} \cdot v\right)\right)[u/x]$$

Si $\alpha \notin v$

$$t[u@\alpha][v/x] = t[v/x][u[v/x]@\alpha]$$

Si $\alpha \neq \beta$ et $\alpha \notin v$

$$t[u@\alpha][v@\beta] = t[v@\beta][u[v@\beta]@\alpha]$$

Si $\alpha \notin u \cup v$

$$\frac{\partial t[u@\alpha]}{\partial x} \cdot v = \left(\frac{\partial t}{\partial x} \cdot v\right)[u@\alpha] + \left(\frac{\delta_1 t}{\delta \alpha} \cdot \left(\frac{\partial u}{\partial x} \cdot v\right)\right)[u@\alpha]$$

Si $\alpha \neq \beta$ et $\alpha \notin u \cup v$

$$\frac{\delta_i t[u@\alpha]}{\delta \beta} \cdot v = \left(\frac{\delta_i t}{\delta \beta} \cdot v\right)[u@\alpha] + \left(\frac{\delta_1 t}{\delta \alpha} \cdot \left(\frac{\delta_i u}{\delta \beta} \cdot v\right)\right)[u@\alpha]$$

Si $x \neq y$ et $x \notin v$

$$\left(\frac{\partial t}{\partial x} \cdot u\right)[v/y] = \frac{\partial t[v/y]}{\partial x} \cdot u[v/y]$$

Si $x \notin v$

$$\left(\frac{\partial t}{\partial x} \cdot u\right)[v@\alpha] = \frac{\partial t[v@\alpha]}{\partial x} \cdot u[v@\alpha]$$

Si $x \notin v$

$$\frac{\partial}{\partial y}\left(\frac{\partial t}{\partial x} \cdot u\right) \cdot v = \frac{\partial}{\partial x}\left(\frac{\partial t}{\partial y} \cdot v\right) \cdot u + \frac{\partial t}{\partial x} \cdot \left(\frac{\partial u}{\partial y} \cdot v\right)$$

Si $x \notin v$

$$\frac{\delta_i}{\delta\alpha}\left(\frac{\partial t}{\partial x} \cdot u\right) \cdot v = \frac{\partial}{\partial x}\left(\frac{\delta_i t}{\delta\alpha} \cdot v\right) \cdot u + \frac{\partial t}{\partial x} \cdot \left(\frac{\delta_i u}{\delta\alpha} \cdot v\right)$$

Si $\alpha \notin v$

$$\left(\frac{\delta_i t}{\delta\alpha} \cdot u\right)[v/x] = \frac{\delta_i t[v/x]}{\delta\alpha} \cdot u[v/x]$$

Si $\alpha \neq \beta$ *et* $\alpha \notin v$

$$\left(\frac{\delta_i t}{\delta\alpha} \cdot u\right)[v@\beta] = \frac{\delta_i t[v@\beta]}{\delta\alpha} \cdot u[v@\beta]$$

Si $\alpha \notin v$

$$\frac{\partial}{\partial x}\left(\frac{\delta_i t}{\delta\alpha} \cdot u\right) \cdot v = \frac{\delta_i}{\delta\alpha}\left(\frac{\partial t}{\partial x} \cdot v\right) \cdot u + \frac{\delta_i t}{\delta\alpha} \cdot \left(\frac{\partial u}{\partial x} \cdot v\right)$$

Si $\alpha \notin v$

$$\frac{\delta_j}{\delta\beta}\left(\frac{\delta_i t}{\delta\alpha} \cdot u\right) \cdot v = \frac{\delta_i}{\delta\alpha}\left(\frac{\delta_j t}{\delta\beta} \cdot v\right) \cdot u + \frac{\delta_i t}{\delta\alpha} \cdot \left(\frac{\delta_j u}{\delta\beta} \cdot v\right)$$

En particulier, si $u_1, \ldots, u_n$ et s sont des termes, alors pour toutes variables $x_1, \ldots, x_n$ telles que pour tous i, j, $x_i \notin u_j$, on pourra écrire

$$\frac{\partial^n s}{\partial x_1 \ldots \partial x_n} \cdot (u_1, \ldots, u_n)$$

au lieu de

$$\frac{\partial}{\partial x_{\sigma(1)}}\left(\ldots \frac{\partial s}{\partial x_{\sigma(n)}} \cdot u_{\sigma(n)} \ldots\right) \cdot u_{\sigma(1)}$$

pour toute permutation σ de $\{1, \ldots, n\}$; pour tous noms $\alpha_1, \ldots, \alpha_n$ et tous $i_1, \ldots, i_n \in \mathbf{N}^*$ tels que pour tous i, j, $\alpha_i \notin u_j$, on pourra écrire

$$\frac{\delta_{i_1,\ldots,i_n} s}{\delta\alpha_1 \ldots \delta\alpha_n} \cdot (u_1, \ldots, u_n)$$

au lieu de

$$\frac{\delta_{i_{\sigma(1)}}}{\delta\alpha_{\sigma(1)}}\left(\ldots \frac{\delta_{i_{\sigma(n)}} s}{\delta\alpha_{\sigma(n)}} \cdot u_{\sigma(n)} \ldots\right) \cdot u_{\sigma(1)}$$

pour toute permutation σ de $\{1, \ldots, n\}$.

3.3 Réductions

3.3.1 β-réduction

La β-reduction en un pas (β^1) est la plus petite relation close par les constructions syntaxiques et telle que :

$$\begin{array}{rcl} (\lambda x s)\, t & \beta^1 & s[t/x] \\ (\mu\alpha\nu)\, t & \beta^1 & \mu\alpha(\nu[t@\alpha]) \\ \mathrm{D}_1\lambda x s \cdot u & \beta^1 & \lambda x\left(\dfrac{\partial s}{\partial x}\cdot u\right) \\ \mathrm{D}_i\mu\alpha\nu \cdot u & \beta^1 & \mu\alpha\left(\dfrac{\delta_i \nu}{\delta\alpha}\cdot u\right) \end{array}$$

Commentons ce dernier cas. Appliquer un terme à une μ-abstraction sur α revient à l'appliquer à tous les sous-termes nommés par α. Le pendant linéaire de ce procédé est naturellement formulé comme suit : différencier une μ-abstraction sur α revient à différencier *l'un* de ses sous-termes nommés par α. Comme pour la dérivation, on obtient en fait la somme des termes ainsi dérivés : c'est exactement ce que fait la μ-différentielle.

Plus formellement :

Définition 15 *On définit la relation β^1 des termes simples vers les termes par induction sur son premier argument :*
- $\mathrm{D}_{i_1,\ldots,i_n}x\cdot(u_1,\ldots,u_n)\ \beta^1\ u$ *si* $u = \mathrm{D}_{i_1,\ldots,i_n}x\cdot({u'}_1,\ldots,{u'}_n)$ *avec* $u_j\ \beta^1\ u'_j$ *pour un* $j\in\{1,\ldots,n\}$ *et* $u_i = u'_i$ *pour tout* $i\neq j$.
- $\mathrm{D}_{i_1,\ldots,i_n}\mu\alpha\nu\cdot(u_1,\ldots,u_n)\ \beta^1\ u$ *si* $u = \mathrm{D}_{i_1,\ldots,i_n}\mu\alpha\nu'\cdot(u_1,\ldots,u_n)$ *avec* $\nu\ \beta^1\ \nu'$ *ou* $u = \mathrm{D}_{i_1,\ldots,i_n}\mu\alpha\nu\cdot({u'}_1,\ldots,{u'}_n)$ *avec* $u_j\ \beta^1\ u'_j$ *pour un* $j\in\{1,\ldots,n\}$ *et* $u_i = u'_i$ *pour tout* $i\neq j$ *ou*

  $$u = \mathrm{D}_{i_1,\ldots,i_{j-1},i_{j+1},\ldots,i_n}\mu\alpha\left(\frac{\delta_{i_j}\nu}{\delta\alpha}\cdot u_j\right)\cdot(u_1,\ldots,u_{j-1},u_{j+1},\ldots,u_n)$$

  pour un certain $j\in\{1,\ldots,n\}$.
- $\mathrm{D}_1^n\lambda x s\cdot(u_1,\ldots,u_n)\ \beta^1\ u$ *si* $u = \mathrm{D}_1^n\lambda x s'\cdot(u_1,\ldots,u_n)$ *avec* $s\ \beta^1\ s'$ *ou* $u = \mathrm{D}_1^n\lambda x s\cdot({u'}_1,\ldots,{u'}_n)$ *avec* $u_j\ \beta^1\ u'_j$ *pour un* $j\in\{1,\ldots,n\}$ *et* $u_i = u'_i$ *pour tout* $i\neq j$ *ou*

  $$u = \mathrm{D}_1^{n-1}\lambda x\left(\frac{\partial s}{\partial x}\cdot u_j\right)\cdot(u_1,\ldots,u_{j-1},u_{j+1},\ldots,u_n)$$

  pour un certain $j\in\{1,\ldots,n\}$.
- $(s)\,t\ \beta^1\ u$ *si* $u = (s')\,t$ *avec* $s\ \beta^1\ s'$ *ou* $u = (s)\,t'$ *avec* $t\ \tilde{\beta}^1\ t'$ *ou* $u = v[t/x]$ *avec* $s = \lambda x v$ *ou* $u = \mu\alpha(\nu[t@\alpha])$ *avec* $s = \mu\alpha\nu$.

Définition 16 *On définit la relation β comme la clôture réflexive et transitive de $\tilde{\beta}^1$.*

Lemme 24 *La relation β est n'est pas contextuelle.*

Preuve: Considérons un terme simple s de la forme $s = (\mu\alpha\nu)\,v$ et $s' = \mu\alpha(\nu[v@\alpha])$. On a bien $s\ \beta\ s'$. Si u est un terme simple, on a toujours $u\ \beta\ u$. Mais on n'a pas en général $\mathrm{D}_i s\cdot u\ \beta\ \mathrm{D}_i s'\cdot u$. En effet, $\mathrm{D}_i s\cdot u = (\mathrm{D}_{i+1}\mu\alpha\nu\cdot u)\,v$ et $\mathrm{D}_i s'\cdot u = \mathrm{D}_i\mu\alpha(\nu[v@\alpha])\cdot u$, or aucune réduction du terme simple $(\mathrm{D}_{i+1}\mu\alpha\nu\cdot u)\,v$ ne mène au terme $\mathrm{D}_i\mu\alpha(\nu[v@\alpha])\cdot u$. □

3.3.2 Réduction parallèle

Définition 17 *On définit la réduction parallèle en un pas π des termes simples vers les termes par induction sur son premier argument :*

- $\mathrm{D}_{i_1,\ldots,i_n}x \cdot (u_1, \ldots, u_n)\ \pi\ u$ *si* $u = \mathrm{D}_{i_1,\ldots,i_n}x \cdot ({u'}_1, \ldots, {u'}_n)$ *avec* $u_j\ \pi\ u'_j$ *pour tout* $j \in \{1, \ldots, n\}$.
- $\mathrm{D}_{i_1,\ldots,i_n}\mu\alpha\nu \cdot (u_1, \ldots, u_n)\ \pi\ u$ *si*

$$u = \mathrm{D}_{i_{(I \setminus J)}}\mu\alpha\left(\frac{\delta_{i_J}\nu'}{\delta\alpha^p} \cdot u'_J\right) \cdot u'_{(I\setminus J)}$$

  avec $\nu\ \pi\ \nu'$, $u_j\ \pi\ u'_j$ *pour tout* $j \in \{1, \ldots, n\}$ *et* $J \subset \{1, \ldots, n\}$ *de cardinal* p.
- $\mathrm{D}_1^n\lambda xs \cdot (u_1, \ldots, u_n)\ \pi\ u$ *si*

$$u = \mathrm{D}_1^{n-p}\lambda x\left(\frac{\partial^p s'}{\partial x^p} \cdot u'_J\right) \cdot u'_{(I\setminus J)}$$

  avec $s\ \pi\ s'$, $u_j\ \pi\ u'_j$ *pour tout* $j \in \{1, \ldots, n\}$ *et* $J \subset \{1, \ldots, n\}$ *de cardinal* p.
- $(s)\,t\ \pi\ u$ *dans chacun des cas suivants :*
  - $u = (s')\,t'$ *avec* $s\ \pi\ s'$ *et* $t\ \overline{\pi}\ t'$ *;*
  - $u = \frac{\partial^n v'}{\partial x^n} \cdot ({u'}_1, \ldots, {u'}_n)[t'/x]$ *avec* $s = \mathrm{D}_1^n\lambda xv \cdot (u_1, \ldots, u_n)$, $v\ \pi\ v'$, $u_j\ \pi\ u'_j$ *pour tout* $j \in \{1, \ldots, n\}$ *et* $t\ \overline{\pi}\ t'$ *;*
  - $u = \frac{\delta_{i_1,\ldots,i_n}\nu'}{\delta\alpha^n} \cdot ({u'}_1, \ldots, {u'}_n)[t'@\alpha]$ *avec* $s = \mathrm{D}_{i_1,\ldots,i_n}\mu\alpha\nu \cdot (u_1, \ldots, u_n)$, $\nu\ \pi\ \nu'$, $u_j\ \pi\ u'_j$ *pour tout* $j \in \{1, \ldots, n\}$ *et* $t\ \overline{\pi}\ t'$.

Lemme 25 *La relation $\overline{\pi}$ <u>n'est pas</u> contextuelle.*

Preuve: Comme pour β. □

Lemme 26 *$\beta^1 \subset \pi \subset \beta$. En particulier, β est la clôture réflexive et transitive de $\overline{\pi}$.*

Preuve: On peut se passer de la contextualité de β pour prouver la deuxième inclusion : on raisonne par induction sur le premier argument de la réduction, et les définitions de π et β permettent de faire passer l'hypothèse d'induction. □

3.4 Confluence ?

Les lemmes 24 et 25 sont de mauvais augure : ils laissent entrevoir que la preuve de confluence ne sera pas aussi simple que pour le λ-calcul différentiel. En effet, en adaptant le contre-exemple de la preuve du lemme 24, on établit les lemmes suivants :

Lemme 27 *On peut trouver des termes t, t', u, u' tels que $t\ \overline{\pi}\ t'$ et $u\ \overline{\pi}\ u'$, mais*

$$t[u/x]\ \ \not\overline{\pi}\ t'[u'/x].$$

Lemme 28 *On peut trouver des termes t, t', u, u' tels que $t\ \overline{\pi}\ t'$ et $u\ \overline{\pi}\ u'$, mais*

$$\frac{\partial t}{\partial x} \cdot u\ \ \not\overline{\pi}\ \frac{\partial t'}{\partial x} \cdot u'.$$

Lemme 29 *On peut trouver des termes t, t', u, u' tels que $t\ \overline{\pi}\ t'$ et $u\ \overline{\pi}\ u'$, mais*

$$t[u@\alpha]\ \ \not\overline{\pi}\ t'[u'@\alpha].$$

Lemme 30 *On peut trouver des termes t, t', u, u' tels que $t \mathrel{\overline{\pi}} t'$ et $u \mathrel{\overline{\pi}} u'$, mais*

$$\frac{\delta_i t}{\delta\alpha} \cdot u \;\not\mathrel{\overline{\pi}}\; \frac{\delta_i t'}{\delta\alpha} \cdot u'.$$

Il est à noter toutefois que ces résultats ne contredisent pas la confluence. En particulier, sur le contre-exemple du lemme 24, on montre facilement que :

Lemme 31 *Si $s = (\mu\alpha\nu)\, v$ et $s' = \mu\alpha(\nu[v@\alpha])$ alors $\mathrm{D}_i s \cdot u \;\beta\; s''$ et $\mathrm{D}_i s' \cdot u \;\beta\; s''$ avec*

$$s'' = \mu\alpha\Big(\frac{\delta_i s[v@\alpha]}{\delta\alpha} \cdot u\Big) = \mu\alpha\Big(\Big(\frac{\delta_{i+1} s}{\delta\alpha} \cdot u\Big)[v@\alpha]\Big).$$

On utilise pour cela la propriété suivante :

Lemme 32 *Pour tout $i \in \mathbf{N}^*$, tout nom α et tous termes s,u et v tels que $\alpha \notin u \cup v$, on a :*

$$\frac{\delta_i s[u@\alpha]}{\delta\alpha} \cdot v = \Big(\frac{\delta_{i+1} s}{\delta\alpha} \cdot v\Big)[u@\alpha].$$

Preuve: Facile, par induction sur s. □

Comme les lemmes 12 et 13 étaient centraux dans la preuve de confluence du λ-calcul différentiel, on ne peut raisonnablement pas l'adapter telle quelle au $\lambda\mu$-calcul différentiel. Il nous reste cependant deux voies pour prouver la confluence du $\lambda\mu$-calcul différentiel. La première est de tenter d'adapter la démonstration de la partie 1.3, avec des variantes des lemmes 12 et 13 d'une forme proche de celle du lemme 31. Le problème qui consiste à trouver une bonne formulation pour ces lemmes, qui les rende utilisable dans une preuve de confluence par induction sur les termes n'est pas encore résolu, mais la piste semble prometteuse.

L'autre solution serait de changer la définition de l'égalité sur les termes pour y inclure l'équation :

$$\mathrm{D}_i \mu\alpha\nu \cdot u = \mu\alpha\Big(\frac{\delta_i \nu}{\delta\alpha} \cdot u\Big).$$

Alors la contextualité serait certainement vérifiée, mais d'une part la notion d'égalité des termes deviendrait bien plus complexe, et peu conforme à l'intuition, et d'autre part la μ-différenciation ne serait plus vraiment la version linéaire de la μ-application, qui elle resterait une réduction (ou non ?). Cette possibilité mérite toutefois d'être explorée.

3.5 $\lambda\mu$-calcul différentiel simplement typé

Comme pour le λ-calcul différentiel, on peut donner un système de types simples relativement naturel dans le $\lambda\mu$-calcul différentiel. Les jugments sont ceux du $\lambda\mu$-calcul et les règles de typage sont données en figure 6.

Le typage vérifie une propriété d'auto-réduction :

Théorème 33 *Si $t \;\beta\; t'$ et $\Gamma \vdash t : A$ alors $\Gamma \vdash t' : A$.*

Preuve: On prouve l'auto-réduction par induction sur les arbres de typage, à l'aide des résultats suivants :

$$\text{Si } \Gamma, x : A \vdash s : B|\Delta \text{ et } \Gamma \vdash u : A|\Delta \text{ alors } \Gamma \vdash s[u/x] : B|\Delta \text{ et } \Gamma, x : A \vdash \tfrac{\partial s}{\partial x} \cdot u : B|\Delta.$$

$$\text{Si } \Gamma \vdash s : C|\alpha : A \to B, \Delta \text{ et } \Gamma \vdash u : A|\Delta \text{ alors } \Gamma \vdash s[u@\alpha] : C|\alpha : B, \Delta.$$

$$\frac{}{\Gamma, x : A \vdash x : A|\Delta} \qquad \frac{\Gamma, x : A \vdash s : B|\Delta}{\Gamma \vdash \lambda xs : A \to B|\Delta} \qquad \frac{\Gamma \vdash s : A|\Delta \quad \Gamma \vdash t : A|\Delta}{\Gamma \vdash (s)\, t : B|\Delta}$$

$$\frac{\Gamma \vdash s : B|\alpha : A, \Delta \quad (\beta : B) \in \alpha : A, \Delta}{\Gamma \vdash \mu\alpha[\beta]s : A \to B|\Delta}$$

$$\frac{\Gamma \vdash s : A_1 \to \ldots \to A_i \to B|\Delta \quad \Gamma \vdash u : A_i|\Delta}{\Gamma \vdash \mathrm{D}_i s \cdot u : A_1 \to \ldots \to A_i \to B|\Delta}$$

$$\frac{}{\Gamma \vdash 0 : A|\Delta} \qquad \frac{\Gamma \vdash s : A \quad \Gamma \vdash t : A|\Delta}{\Gamma \vdash s + t : A|\Delta}$$

FIG. 6 – Règles de typage du $\lambda\mu$-calcul différentiel.

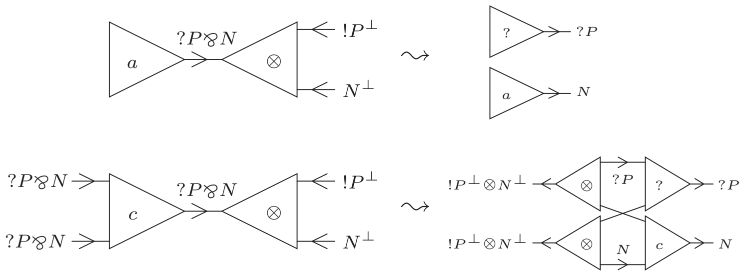


FIG. 7 – Paires rédex/réduit supplémentaires pour les réseaux polarisés.

Si $\Gamma \vdash s : C|\alpha : A_1 \to \ldots \to A_i \to B, \Delta$ *et* $\Gamma \vdash u : A_i|\Delta$ *alors*
$\Gamma \vdash \frac{\delta_i s}{\delta\alpha} \cdot u : C|\alpha : A_1 \to \ldots \to A_i \to B, \Delta.$

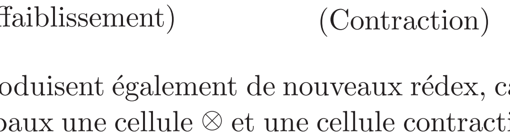
□

La question de l'éventuelle normalisabilité des termes typés n'a pas été abordée.

4 Réseaux différentiels polarisés

4.1 Réseaux polarisés

Dans [Lau03], Olivier LAURENT démontre que les réseaux de preuve polarisés permettent de traduire de façon correcte le $\lambda\mu$-calcul et sa réduction. Les réseaux polarisés sont obtenus à partir des réseaux de la logique linéaire en ajoutant deux nouvelles cellules, un affaiblissement et une contraction sur les types de sortie[7] :



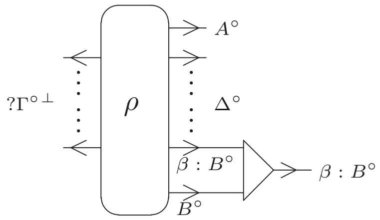
(Affaiblissement) (Contraction)

Ces nouvelles cellules introduisent également de nouveaux rédex, car on peut maintenant lier par leurs ports principaux une cellule $\otimes$ et une cellule contraction ou affaiblissement. Les paires rédex–réduit correspondantes sont données en figure 7.

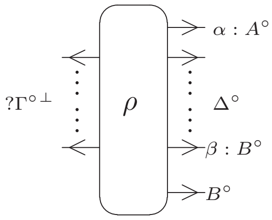
[7]Il y a aussi une nouvelle boîte, qui permet de fabriquer une cellule de type sortant $!N, N_1, \ldots, N_n, ?P_1, \ldots, ?P_p$ à partir d'un réseau de type $N, N_1, \ldots, N_n, ?P_1, \ldots, ?P_p$.

Le $\lambda\mu$-calcul simplement typé se traduit dans les réseaux différentiels polarisés de la même façon que le λ-calcul typé se traduit dans les réseaux de preuve de la logique linéaire. Les cas de la variable, l'abstraction et de l'application sont traités comme au 2.4. Le cas de la μ-abstraction se traite comme suit :

$$\frac{\Gamma \vdash s : B|\alpha : A, \Delta \quad (\beta : B) \in \alpha : A, \Delta}{\Gamma \vdash \mu\alpha[\beta]s : A \to B|\Delta}$$ est traduit par :



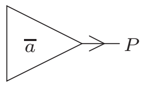
si $\Gamma \vdash s : B|\alpha : A, \Delta$ est traduit par :



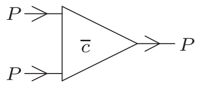
4.2 Une version différentielle ?

En ajoutant les cellules d'affaiblissement et de contraction sur les types de sortie et les règles de réduction de la figure 7 aux notions introduites dans la partie 2, on obtient les réseaux différentiels avec affaiblissement et contraction sur les types de sortie, et la traduction du $\lambda\mu$-calcul différentiel simplement typé obtenue en ajoutant le cas du paragraphe précédent semble[8] correcte. Ce n'est toutefois pas véritablement le concept qui nous intéresse ici. On se demande plutôt ce que devraient être les cellules duales de l'affaiblissement et de la contraction sur les types de sortie dans les réseaux différentiels, et plus particulièrement quelles doivent être les règles de réduction associées.

On ajoute donc deux nouvelles cellules, qui sont naturellement de nouvelles versions du co-affaiblissement et de la co-contraction :

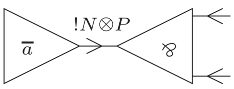


(Co-affaiblissement)

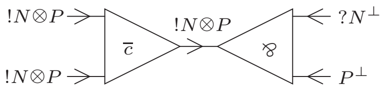


(Co-contraction)

Les deux nouveaux rédex possibles sont :



et



L'intuition donne deux manières de réduire ces rédex, qui ne sont pas équivalentes. La première consiste à faire fonctionner la cellule $\parr$ comme si elle était derrière une déréliction, et si le co-affaiblissement (ou la co-contraction) était celui des réseaux différentiels normaux. On aboutit aux paires rédex–réduit de la figure 8. Cette réduction

[8]Le résultat n'a pas été démontré formellement, mais la traduction est faite pour être correcte. En particulier, le problème qui se posait pour la confluence ne se pose pas ici, car on raisonne en termes d'équivalence de réseaux et non en termes de réduction en un pas.

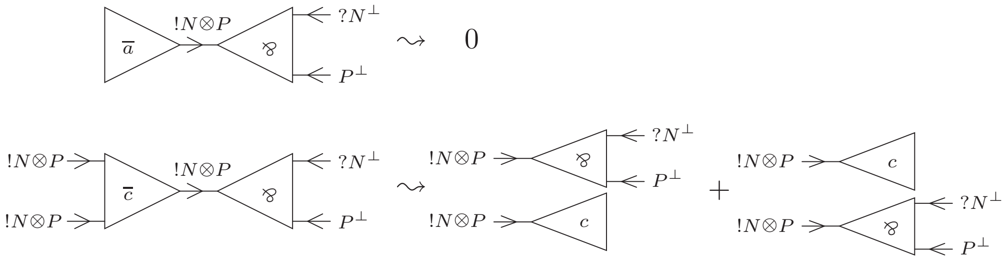


FIG. 8 – Paires rédex/réduit supplémentaires pour les réseaux différentiels polarisés : première version.

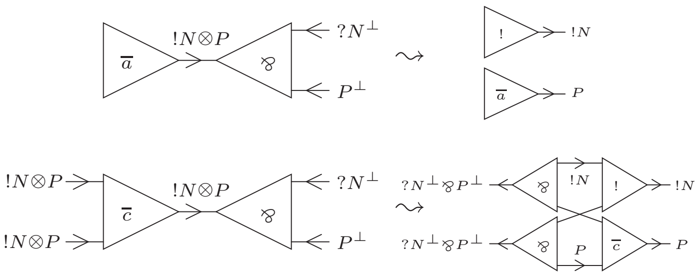


FIG. 9 – Paires rédex/réduit supplémentaires pour les réseaux différentiels polarisés : deuxième version.

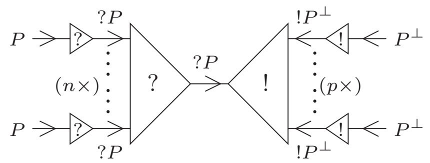
n'a cependant pas de fondement sémantique.

La deuxième de réduction possible est l'exact symétrique de la définition de la figure 7. On aboutit aux réductions de la figure 9. Cette réduction est préférée dans la suite car elle correspond à une possible sémantique dénotationnelle du $\lambda\mu$-calcul différentiel qui n'est toutefois pas développée ici.

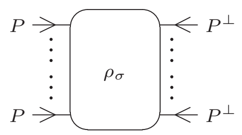
On appelle réseaux différentiels polarisés les réseaux obtenus à partir des cellules des réseaux différentiels (sans promotion) auxquelles on ajoute affaiblissement et contraction sur les formules négatives et co-affaiblissement et co-contraction sur les formules positives, et pour lesquels la réduction est celle définie par les couples rédex–réduit des figures 3, 7 et 9.

4.3 Premiers pas

L'étude des réseaux différentiels polarisés commence à peine, mais un problème assez intéressant se pose déjà. Dans les réseaux différentiels sans boîte [ER04], on pouvait généraliser la réduction en définissant des arbres maximaux de contractions et de co-contractions : l'associativité et la commutativité de la (co-)contraction, associée au fait que le (co-)affaiblissement en est l'élément neutre permettent de remplacer (co-)affaiblissement et (co-)contraction par des cellules avec un nombre nombre fini quelconque de ports auxiliaires. On les appelle (co-)contractions généralisées. Quand on remplace un arbre maximal de (co-)contractions par une contraction généralisée, ses ports auxiliaires sont forcément tous reliés à une (co-)déréliction (c'est forcé par le type). On obtient donc la forme générale de rédex :

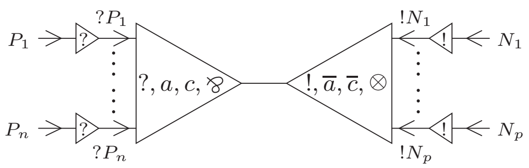


Si $n = p$, ce réseau se réduit en la somme sur toutes les permutations σ de $\{1, \ldots, n\}$ des :



où ρ_σ est le réseau formé de n fils, et qui branche le i-ème port de gauche sur le $\sigma(i)$-ème port de droite. Si $n \neq p$, il se réduit en le réseau nul.

Si on cherche un équivalent de cette généralisation pour dans les réseaux différentiels polarisés, on obtient des structures plus complexes (et donc plus intéressantes). Dans ces réseaux, les cellules dont le port principal a pour type sortant une formule négative sont le $\parr$, l'affaiblissement, la déréliction et la contraction. Les cellules dont le port principal a pour type sortant une formule positive sont le $\otimes$, le co-affaiblissement, la co-déréliction et la co-contraction. La forme générale d'un rédex est :



Le point intéressant est que des $\parr$ peuvent apparaître dans les arbres négatifs, et des $\otimes$ dans les arbres positifs. Ces cellules déterminent des contraintes sur les appariements possibles entre les cellules déréliction et co-déréliction aux feuilles des arbres. Le typage impose une structure commune aux deux arbres : il existe un entier h tel que tout chemin d'une déréliction à une co-déréliction passe par exactement h cellules $\parr$ et h cellules $\otimes$. Les arbres négatifs (resp. positifs) sont donc organisés en strates par les cellules $\parr$ (resp. $\otimes$).

On prouve facilement que si $n \neq p$, alors la réduction aboutit au réseau nul. Mais si $n = p$, on n'obtient pas forcément tous les réseaux de permutations vus plus haut. En fait, on peut montrer que la réduction aboutit soit à 0, soit à la somme sur tous les $\sigma \in \mathcal{G}$ des ρ_σ, où $\mathcal{G}$ est un sous-groupe du groupe de permutation de $\{1, \ldots, n\}$. De plus il existe une partition $\mathcal{P}$ de $\{1, \ldots, n\}$ tel que $\mathcal{G}$ est isomorphe au produit des groupes de permutation des éléments de $\mathcal{P}$.

Est-ce que tous les sous-groupes de cette forme sont atteints ? Sinon, lesquels le sont ? Peut-on caractériser simplement les couples d'arbres qui ne se réduisent pas en 0 ? Ce sont quelques unes des questions auxquelles il faudra répondre avant de pousser l'étude des réseaux différentiels polarisés plus loin.

Références

[Ehr01] Thomas Ehrhard. On koethe sequence spaces and linear logic. *Mathematical Structures in Computer Science*, 12 :579–623, 2001. `http://iml.univ-mrs.fr/%7Eehrhard/pub.html`.

[Ehr04] Thomas Ehrhard. Finiteness spaces. `http://iml.univ-mrs.fr/%7Eehrhard/pub.html`, 2004.

[ER03] Thomas Ehrhard and Laurent Régnier. The differential lambda-calculus. *Theoretical Computer Science*, 309 :1–41, 2003. `http://iml.univ-mrs.fr/%7Eehrhard/pub.html`.

[ER04] Thomas Ehrhard and Laurent Régnier. Differential interaction nets. `http://iml.univ-mrs.fr/%7Eehrhard/pub.html`, 2004.

[Laf95] Y. Lafont. From proof nets to interaction nets. In J.-Y. Girard, Y. Lafont, and L. Regnier, editors, *Advances in Linear Logic*, pages 225–247. Cambridge University Press, 1995.

[Lau03] Olivier Laurent. Polarized proof-nets and $\lambda\mu$-calculus. *Theoretical Computer Science*, 290(1) :161–188, January 2003. `http://www.pps.jussieu.fr/~laurent/`.

[Par92] M. Parigot. $\lambda\mu$-calculus : An algorithmic interpretation of classical natural deduction. In A. Voronkov, editor, *Logic Programming and Automated Reasoning : Proc. of the International Conference LPAR'92*, pages 190–201. Springer, Berlin, Heidelberg, 1992.

[Ré92] Laurent Régnier. *λ-calcul et réseaux.* PhD thesis, Université Paris 7, 1992.