THÈSE DE DOCTORAT

présentée à

L'UNIVERSITÉ PARIS VII

Spécialité : Informatique

par

Luc Maranget

Sujet de la thèse :

La stratégie paresseuse

Soutenue le 6 juillet 1992 devant la Commission d'examen composée de

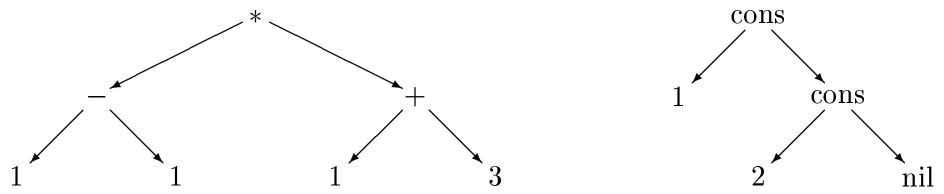
MM.	**Guy COUSINEAU**	**Président**
	Bruno COURCELLE	**Rapporteurs**
	Jan Willem KLOP	
	Jean-Pierre JOUANNAUD	**Examinateurs**
Mme	**Hélène KIRCHNER**	
M.	**Jean-Jacques LÉVY**	

Introduction

Argument de cette thèse

Il est maintenant bien connu que les systèmes de réécriture de termes en général et le λ-calcul en particulier constituent un bon modèle des langages de programmation. Le succès de cette approche s'explique largement par la simplicité de ce modèle, qui tient beaucoup au caractère purement syntaxique de ces formalismes.

Les objets manipulés par les programmes, leurs résultats et les programmes eux mêmes sont des assemblages de symboles que l'on peut dessiner sur une feuille de papier, inscrire dans la mémoire d'un ordinateur ou écrire de gauche à droite comme j'écris cette ligne de texte. Par exemple, on peut représenter les entiers naturels de façon assez intuitive mais peu efficace à l'aide de deux symboles 0 et S, l'entier n étant codé par la suite de symboles $SS\ldots S0$, où le symbole S est répété n fois. On peut plutôt considérer chaque entier comme un symbole distinct en se donnant une infinité dénombrable de symboles notés justement 0, 1, 2… Si l'infini fait peur, on se limite aux entiers entre zéro et soixante-cinq mille cinq cent trente-cinq par exemple. Pour traiter des objets plus structurés, on doit recourir à des conventions ou à des dessins. Ainsi, l'expression arithmétique $(1-1)*(1+3)$ ou la liste de deux entiers $[1;2]$ sont ils aussi les arbres de la figure suivante :



Le calcul opère selon des règles de réécriture qui indiquent comment transformer les termes qui ne sont pas encore des résultats. Par exemple, un ensemble de règles qui traite l'arithmétique entière contient les quatre opérations de base appliquées à tous les entiers.

$$0+0 \to 0, \quad 1+0 \to 1, \quad 0+1 \to 1, \quad 1+1 \to 2, \quad \ldots$$

Dans l'expression $(\underline{1-1})*(\underline{1+3})$, il y a deux sous-expressions réductibles (soulignées), que l'on appelle des *radicaux.* Un terme qui ne contient pas de radical (et donc n'est plus réductible) est un résultat ou une *forme normale.* Tant qu'on se limite aux quatre opérations, le calcul est déterministe, dans le sens que l'on a beau réduire les radicaux dans n'importe quel ordre, on obtient toujours le même résultat.

$$(\underline{1-1})*(1+3) \to 0*(\underline{1+3}) \to \underline{0*4} \to 0 \qquad (1-1)*(\underline{1+3}) \to (\underline{1-1})*4 \to \underline{0*4} \to 0$$

Et même un langage de programmation aussi rustique que le langage C ne spécifie pas lequel des deux arguments d'une multiplication doit être calculé le premier, essentiellement parce que cet ordre n'a aucune importance.

On se donne deux nouvelles constantes, les valeurs de vérité, ou booléens, `vrai` et `faux`. Puis, on définit l'opération égalité sur les entiers en énumérant toutes les possibilités.

$$0 = 0 \to \texttt{vrai}, \quad 1 = 0 \to \texttt{faux}, \quad 0 = 1 \to \texttt{faux}, \quad 1 = 1 \to \texttt{vrai}, \ldots$$

La fonction *if* à trois arguments permet d'utiliser les résultats des comparaisons. Sa définition comprend deux règles.

$$\mathit{if}(\texttt{vrai}, x, y) \to x \qquad\qquad \mathit{if}(\texttt{faux}, x, y) \to y$$

Les règles de la la fonction *if* comportent des variables x et y qui représentent n'importe quel terme. Le mécanisme général de l'appel de la fonction *if* est, d'abord de reconnaître si un terme est de l'une ou de l'autre des deux formes définies par les parties gauches des règles du *if*, puis de remplacer ce terme reconnu par la partie droite de la règle correspondante (le corps de la fonction). Si le membre droit sélectionné comprend des variables (ou paramètres formels), celles-ci sont remplacées par leurs valeurs au moment de l'appel (ou paramètres effectifs). Par exemple, on a les réécritures $\mathit{if}(\texttt{vrai}, 1, 2) \to 1$ et $\mathit{if}(\texttt{vrai}, (1 * 1), 2) \to 1 * 1$.

Le programmeur peut maintenant suivre l'exemple de la fonction *if* et définir de nouvelles fonctions. Par exemple, la fonction factorielle se définit en une seule règle.

$$\mathit{fac}(n) \to \mathit{if}(0 = n, 1, n * \mathit{fac}(n - 1))$$

La factorielle de un se calcule ensuite comme ceci :

$$\begin{array}{rcl}
\underline{\mathit{fac}(1)} & \to & \mathit{if}(\underline{0 = 1}, 1, 1 * \mathit{fac}(1 - 1)) \\
& \to & \underline{\mathit{if}(\texttt{faux}, 1, 1 * \mathit{fac}(1 - 1))} \\
& \to & 1 * \mathit{fac}(\underline{1 - 1}) \\
& \to & 1 * \underline{\mathit{fac}(0)} \\
& \to & 1 * \mathit{if}(\underline{0 = 0}, 1, \mathit{fac}(0 - 1)) \\
& \to & 1 * \underline{\mathit{if}(\texttt{vrai}, 1, \mathit{fac}(0 - 1))} \\
& \to & \underline{1 * 1} \to 1
\end{array}$$

Les règles d'un système de réécriture sont *a priori* arbitrairement complexes. Voici, par exemple, une fonction qui échange tous les éléments adjacents d'une liste.

$$\mathit{swap}(\mathit{nil}) \to \mathit{nil} \qquad\qquad \mathit{swap}(\mathit{cons}(x, \mathit{nil})) \to \mathit{cons}(x, \mathit{nil})$$
$$\mathit{swap}(\mathit{cons}(x, \mathit{cons}(y, l))) \to \mathit{cons}(y, \mathit{cons}(x, \mathit{swap}(l)))$$

Toutefois, les calculs doivent rester déterministes et les résultats uniques. Le moins que l'on puisse faire est d'interdire les couples de règles ambiguës du type $G(0, x) \to 0$, $G(x, 0) \to 1$, car, quel serait sinon le résultat de l'appel $G(0, 0)$? On prend l'option drastique d'exclure toutes les règles dont les membres gauches sont ambigus. En effet, si on considère par exemple les deux règles $H(0, x) \to 1$ et $H(x, 0) \to \mathit{fac}(x)$, il n'y a pas de moyen systématique de savoir que $\mathit{fac}(0)$ se réduira finalement en 1 et que l'appel $H(0, 0)$ n'est pas ambigu. Ce faisant, on se limite aux *systèmes orthogonaux* de réécriture de termes, une classe de systèmes de réécriture déjà très étudiée [Hue 79].

Les programmes ont donc au plus un résultat. Certains programmes n'ont pas de résultat du tout comme, par exemple, le membre gauche $\perp$ d'une règle $\perp \to \perp$. Se pose ensuite la question de calculer effectivement ce résultat quand il existe. Dans ce but, les compilateurs ou les interpréteurs utilisent une *stratégie de réduction*, c'est à dire une méthode de choix du prochain radical à réduire à chaque étape. La stratégie la plus couramment utilisée est la stratégie de l'appel par valeur qui exige que les arguments soient complètement réduits avant d'appeler une fonction. Cela revient à contracter les radicaux les plus intérieurs en premier. Par exemple, dans le terme $fac(1-1)$, la soustraction est d'abord effectuée avant d'appeler la fonction *fac*. Cette règle ne s'applique pas à la primitive *if*, pour laquelle on ne demande l'évaluation préalable que du seul premier argument. Le calcul de $fac(1)$ présenté plus haut suit la stratégie de l'appel par valeur.

Les avantages de l'appel par valeur sont nombreux. Le programmeur peut facilement prévoir l'ordre d'évaluation des expressions et a donc une idée précise du déroulement des programmes. On peut alors envisager des "effets de bords" tels que l'ouverture d'un fichier ou la modification de la valeur d'une variable. En outre, la stratégie de l'appel par valeur peut être mise en œuvre efficacement à l'aide d'une pile qui enregistre les appels en cours. L'inconvénient majeur de l'appel par valeur est qu'il est incorrect, dans le sens qu'il peut rater certains résultats. Soit la définition de fonction :

$$F(n,m) \to \mathit{if}\,(n=0,1,F(n-1,F(n,m+1)))$$

L'appel par valeur ne parvient pas à donner un résultat au terme $F(1,0)$. En effet, on a la suite infinie de réductions suivante :

$$\underline{F(1,0)} \to \ldots \to F(0,\underline{F(1,1)}) \to \ldots \to F(0,F(-1,\underline{F(1,2)})) \to \ldots$$

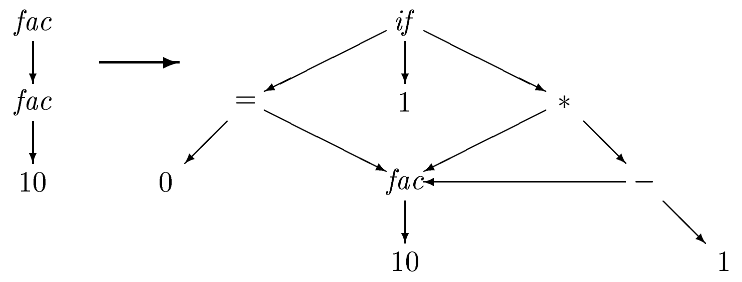
Et pourtant, en évitant le calcul du deuxième argument de la fonction F, il vient :

$$\begin{array}{lll}
\underline{F(1,0)} & \to & \mathit{if}\,(\underline{1=0},1,F(1-1,F(1,0+1))) \\
 & \to & \underline{\mathit{if}\,(\texttt{faux},1,F(1-1,F(1,0+1)))} \\
 & \to & \underline{F(1-1,F(1,0+1))} \\
 & \to & \mathit{if}\,(\underline{1-1}=0,1,F(1-1,F(1,0+1))) \\
 & \to & \mathit{if}\,(\underline{0=0},1,F(1-1,F(1,0+1))) \\
 & \to & \underline{\mathit{if}\,(\texttt{vrai},1,F(1-1,F(1,0+1)))} \\
 & \to & 1
\end{array}$$

Ce qui est recherché, c'est donc une stratégie de réduction *correcte*, qui calcule le résultat d'un programme dès que celui ci existe. Cette stratégie est connue, il s'agit de l'appel par nom qui, dans les cas simples, consiste à ne pas réduire les arguments avant d'appeler les fonctions. L'appel par nom revient donc à contracter les radicaux les plus extérieurs en premier. Intuitivement, en procédant ainsi, on ne risque pas de s'engager dans le calcul infini d'un argument qui n'est en fait pas utilisé, comme l'indiquent simplement la définition $K(x,y) \to x$ et la réduction $\underline{K(0,\perp)} \to 0$. Le cas général des systèmes orthogonaux, quoique plus complexe, est également connu [Hue 79]. Le radical extérieur à contracter dépend alors des règles de réécriture elles-mêmes. Nous connaissons déjà un exemple de l'appel par nom généralisé, avec la primitive *if*. Dans l'appel $\mathit{if}\,(F(0,1)=1,0,\perp)$, c'est le radical $F(0,1)$ qui doit être réduit en premier (en suivant, bien entendu, l'appel par nom), pour que l'on sache si c'est la règle $\mathit{if}\,(\texttt{vrai},x,y) \to x$ ou $\mathit{if}\,(\texttt{faux},x,y) \to y$ qui s'applique. En ce sens, le radical $F(0,1)$ est nécessaire pour pouvoir réduire la primitive *if* qui est elle-même l'appel de fonction en attente le plus extérieur. C'est pourquoi l'appel par nom est aussi parfois connu comme l'appel par nécessité, l'idée étant que ce sont les primitives qui commandent les réductions des radicaux dont elles ont besoin.

Bien qu'il évite certaines suites de réductions infiniment coûteuses, l'appel par nom est inefficace, car il conduit à dupliquer certains radicaux. Par exemple, le calcul de $fac(\overline{fac(10)})$ entraînera de nombreux appels de $fac(10)$, car, si on contracte d'abord le radical le plus extérieur (souligné), on fabriquera trois versions du radical intérieur (surligné) :

$$\underline{fac(\overline{fac(10)})} \to \mathit{if}\,(0=\overline{fac(10)},1,\overline{fac(10)} * fac(\overline{fac(10)}-1))$$



Pourtant, les trois instances (surlignées, à droite) du radical *fac*(10) sont “moralement” les mêmes puisqu’elles proviennent d’un même radical initial (surligné, à gauche). En particulier, si on sait que l’un des trois radicaux *fac*(10) se révélera nécessaire par la suite, on doit pouvoir contracter le radical *fac*(10) avant de le copier. On a donc l’idée d’identifier ces trois radicaux, cela revient à ne plus considérer des termes mais des graphes, comme l’illustre la figure ci-dessus.

Si une instance du radical *fac*(10) est contractée par la suite, alors toutes seront contractées simultanément et il n’y aura pas de risque de redites. On définit ainsi une nouvelle stratégie de réduction, l’appel paresseux, qui est l’appel par nom dans les graphes. Intuitivement, l’appel paresseux doit pouvoir calculer les résultats des programmes et ce pour un coût minimal, puisqu’il ne contracte pas de radicaux inutiles et qu’il évite les contractions multiples des radicaux utiles. L’intuition qui sous-tend les stratégies optimales est donc claire : contracter des radicaux nécessaires —des radicaux qui doivent être contractés de toute façon—, tout en évitant les duplications.

Dès que l’on quitte le domaine des intuitions pour celui des formalismes, les choses se compliquent et cette thèse répond à un certain nombre de questions restées ouvertes. Si on déroule l’appel *fac*(10), alors de nouveaux radicaux apparaissent tels que, par exemple, un appel *fac*(9). La question du partage des trois appels *fac*(9) lancés à partir des trois instances *fac*(10) se pose sur deux plans. Comment savoir que ces trois radicaux *fac*(9) sont “moralement” les mêmes ? Et seront ils effectivement les mêmes, si on calcule sur les graphes ? Enfin, une fois résolus ces deux problèmes, reste à vérifier que l’appel paresseux se conforme à l’intuition en étant optimal.

Plan de la thèse

Le premier chapitre met en place un formalisme rigoureux de la réduction de termes et de la réduction de graphes. La réduction de graphes est modélisée par la réduction de termes *simplement étiquetés*, chaque sous-terme d’un terme portant une étiquette et deux sous-termes effectivement partagés dans un graphe portant la même étiquette. Cette approche originale est simplificatrice car, en substituant les termes aux graphes, on se donne un objet d’étude beaucoup plus simple. L’idée des radicaux moralement identiques —ou des familles de radicaux, pour reprendre la terminologie de [Lév 78]— a en fait deux origines. La première relève d’une démarche naturelle d’amélioration de l’efficacité de la mise en œuvre de la réduction de termes par utilisation de la réduction de graphes. En effet, l’introduction d’un certain partage est traditionnelle quand on cherche à améliorer les performances d’un programme ou d’un algorithme, cela revient généralement à remplacer un objet par un pointeur sur cet objet. Par exemple, plutôt que de trier en place un tableau de gros objets, on créera un tableau de pointeurs sur ces objets, qui sera ensuite trié en considérant l’ordre sur les objets pointés. On vise ainsi à économiser le temps, puisque la copie d’un gros objet, dont la taille peut même être *a priori* inconnue, est remplacée par la copie d’un pointeur sur cet objet, opération qui a lieu en un temps constant et faible. Dans le cas de la réduction de termes, on vise aussi l'économie d'espace, car un pointeur vers un terme occupe généralement moins de mémoire machine que le terme pointé.

Le chapitre deux de la thèse traite de l'autre origine de l'idée des familles de radicaux qui est plutôt à rapprocher de l'exigence de déterminisme des calculs. Soient par exemple les deux règles de réécriture :

$$F(x) \to G(x,x) \qquad\qquad A \to B$$

Dans le terme $\overline{F(\widehat{A})}$, il y a deux radicaux que l'on peut contracter. On peut soit commencer par contracter le radical extérieur (surligné) ; soit commencer par contracter le radical intérieur (qui porte un chapeau). Dans les deux cas, on obtient un terme intermédiaire qui contient des radicaux qui proviennent du radical qui n'a pas été contracté à la première étape (dans la figure suivante, ces radicaux *résidus* sont surlignés ou portent un chapeau, comme le radical initial dont ils proviennent). Les calculs sont ici déterministes dans le sens précis que le résultat atteint $G(B,B)$ est indépendant de l'ordre dans lequel sont contractés les radicaux initiaux (et leurs résidus).

$$\begin{tikzcd}
 & & \overline{F(\widehat{A})} \arrow[dl] \arrow[dr] & \\
 & G(\widehat{A},\widehat{A}) \arrow[dl] \arrow[dr] & & \overline{F(B)} \arrow[ddl] \\
G(\widehat{A},B) \arrow[drr] & & G(B,\widehat{A}) \arrow[d] & \\
 & & G(B,B) &
\end{tikzcd}$$

Dans cet exemple, la contraction des deux radicaux initiaux est nécessaire pour produire le résultat $G(B,B)$, la contraction du radical initial extérieur $\overline{F(A)}$ produisant le symbole G et la contraction du radical initial intérieur $\widehat{A}$ produisant le symbole B. Si on s'impose de ne contracter qu'un radical à la fois, contracter le radical initial extérieur en premier conduit à un calcul —à gauche sur la figure ci-dessus— qui comprend une étape de réduction de plus que le calcul qui commence par contracter le radical intérieur. Pourtant, idéalement, le coût du calcul d'un certain résultat ne devrait pas dépendre de l'ordre dans lequel on contracte les radicaux nécessaires, comme ce résultat lui même n'en dépend pas.

$$\begin{tikzcd}
 & \overline{F(\widehat{A})} \arrow[dl] \arrow[dr] & \\
G(\widehat{A},\widehat{A}) \arrow[dr] & & \overline{F(B)} \arrow[dl] \\
 & G(B,B) &
\end{tikzcd}$$

Comme indiqué par la figure ci-dessus, la solution consiste à contracter ensemble les radicaux résidus d'un même radical initial, de sorte que le diagramme des réductions retrouve sa symétrie et que la permutation des contractions nécessaires n'influe pas sur le coût, à condition de compter pour un la contraction simultanée de tous les radicaux résidus. Ici, on remplace les deux calculs possibles $G(\underline{A}, A) \to G(B, \underline{A}) \to G(B, B)$ et $G(A, \underline{A}) \to G(\underline{A}, B) \to G(B, B)$ par une étape de réduction *parallèle* $G(\underline{A}, \underline{A}) \to G(B, B)$. Dans un premier temps, les familles sont donc les ensembles de radicaux résidus d'un même radical initial. En s'imposant de toujours contracter ensemble les membres d'une même famille, on compte annuler un effet qui dépend des règles de réécriture (ici, la variable x est présente deux fois dans le membre droit de la règle $F(x) \to G(x, x)$), pour se concentrer sur des concepts qui valent pour tous les systèmes orthogonaux.

La définition élémentaire des familles est insuffisante en raison de l'existence de radicaux qui ne sont résidus d'aucun radical initial. Par exemple, si on se donne la règle de réécriture supplémentaire $B \to C$, alors les calculs de la figure ci-dessus peuvent se prolonger par la réduction parallèle $G(\underline{B}, \underline{B}) \to G(C, C)$. On conçoit bien que les deux radicaux B en jeu sont de la même famille, en considérant le calcul permuté $F(\underline{A}) \to F(\underline{B}) \to \underline{F(C)} \to G(C, C)$, calcul qui ne contracte qu'un seul radical B. Pour distinguer les familles sans opérer de permutation de réductions, le chapitre deux introduit un système *d'étiquetage complet*, qui est tel que les famille sont des ensembles de radicaux qui portent les mêmes étiquettes. En effet, les radicaux de même familles sont des radicaux de même origine (voir le cas des deux radicaux B, créés par contraction de deux radicaux A de la même famille), et un système d'étiquetage sophistiqué permet de capturer l'historique des radicaux.

Le chapitre trois démontre deux résultats fondamentaux des systèmes orthogonaux. Le lemme des déplacements parallèle traduit le déterminisme des calculs tel que déjà évoqué plusieurs fois : les calculs ont au plus un résultat et l'ordre relatif des étapes de réduction utilisées pour atteindre ce résultat n'est pas important. Le théorème des développements finis est plus technique, la version très générale et nouvelle que je donne ici affirme qu'il est impossible de procéder à un calcul infini sans créer indéfiniment de nouvelles familles de radicaux.

Le chapitre quatre est l'étude de la structure des calculs commencés à partir d'un terme M donné, ou *dérivations* issues de M. Ce chapitre commence par le traitement rigoureux de la permutation des étapes de réductions. Ensuite sont définis les radicaux nécessaires, l'idée, déjà évoquée, est qu'un radical est nécessaire quand sa contraction est obligatoire pour parvenir à un résultat. Tous les radicaux ne sont pas nécessaires, du fait des règles de réécriture dont les membres droits ne contiennent pas toutes les variables des membres gauches. Soient par exemple les trois règles :

$$H(x, y) \to I(y) \qquad\qquad A \to B \qquad\qquad C \to D$$

On considère ensuite les dérivations issues du terme $H(A, C)$:

$$\begin{tikzcd}
& H(A,C) \arrow[dl] \arrow[d] \arrow[dr] & \\
I(C) \arrow[dr] & H(B,C) \arrow[l] \arrow[r] & H(A,D) \arrow[dl] \\
& I(D) &
\end{tikzcd}$$

Dans le terme $H(A, C)$, le radical initial A n'est pas nécessaire, car on peut calculer la forme normale $I(D)$ sans le contracter, comme le montrent la dérivation la plus à gauche et la dérivation la plus à droite de la figure.

Le chapitre quatre culmine par la définition de stratégies de réduction dans le cadre général des systèmes orthogonaux. Une distinction est faite entre les stratégies d'appel par nom et d'appel par nécessité. En simplifiant, l'appel par nom contracte des radicaux nécessaires en commençant par les plus extérieurs, tandis que l'appel par nécessité contracte des radicaux nécessaires sans autre contrainte. Il est montré que l'appel par nécessité atteint toujours le résultat final d'un calcul (i.e., la forme normale du terme initial M) quand celui-ci existe, tandis que l'appel par nom est plus précis en atteignant toutes sortes de résultats partiels dès que ceux-ci existent.

Dans la figure précédente, les deux dérivations $\underline{K(A,C)} \to I(\underline{C}) \to I(D)$ et $K(A,\underline{C}) \to \underline{K(A,D)} \to I(D)$ sont des appels par nécessité, la première dérivation étant un appel par nom. La différence entre appel par nécessité et appel par nom apparaît plus clairement, si on se donne la règle de réécriture supplémentaire $\perp \to \perp$ et que l'on considère les dérivations issues du terme $K(A,\perp)$. Ce terme n'a pas de forme normale et il n'est pas surprenant de voir que ni l'appel par nom $\underline{K(A,\perp)} \to I(\underline{\perp}) \to I(\underline{\perp}) \to \ldots$, ni l'appel par nécessité $K(A,\underline{\perp}) \to K(A,\underline{\perp}) \to \ldots$ ne terminent. Cependant, la première de ces dérivations parvient à atteindre la *forme normale de tête* $I(\perp)$, qui est un terme dont le symbole de tête I n'est plus modifiable. De ce fait, l'appel par nom donne une information sur les résultats partiels des calculs.

Le chapitre cinq précise l'étude du chapitre quatre en la restreignant aux dérivations qui contractent les familles de radicaux ou dérivations *complètes*. Il est montré que ces dérivations (dont la définition utilise les étiquettes complètes) peuvent être mises en œuvre à l'aide de la réduction de graphes (modélisée par la réduction de termes simplement étiquetés). En quelque sorte, il est possible d'oublier les étiquettes complètes, les familles de radicaux coïncidant avec les sous-termes partagés dans les graphes. Des stratégies produisant des dérivations complètes sont définies relativement aux stratégies du chapitre quatre. Ainsi, l'appel complet par nécessité contracte des familles de radicaux dont un membre est un radical nécessaire, et l'appel paresseux contracte des familles de radicaux dont un membre est susceptible d'être contracté par l'appel par nom. En raison du résultat sur la mise en œuvre des familles par les graphes, on peut voir chacune de ces nouvelles stratégies comme la transposition aux graphes de la stratégie correspondante définie sur les termes. À chaque propriété de correction des stratégies sur les termes correspond une propriété d'optimalité des stratégies complètes. Plus précisément, l'appel complet par nécessité est optimal pour le calcul des formes normales, tandis que l'appel paresseux est aussi optimal pour le calcul des résultat partiels. Tous ces résultats d'optimalité découlent d'un unique théorème d'optimalité dont le libellé reprend les termes du lemme des déplacements parallèles.

Une démarche logique, adoptée dans le chapitre six, est ensuite de réaliser concrètement la stratégie de l'appel paresseux dans le cadre d'un langage de programmation. Le langage choisi est ML et l'appel par filtrage de ce langage est traité sans aucune restriction. Cet appel correspond à des règles de réécriture formellement ambiguës, mais il est montré que ces règles ont en fait une sémantique par réécriture conforme à nos exigences. L'existence de la stratégie paresseuse n'entraîne malheureusement pas automatiquement la possibilité de la mettre en œuvre. En effet, d'une part le caractère nécessaire d'un radical n'est pas décidable en général et, d'autre part, on souhaite utiliser les ordinateurs les moins exotiques possibles, c'est à dire des machines séquentielles qui ne font qu'une seule chose à la fois. Heureusement, la stratégie paresseuse est réalisable pour une large classe de programmes. Le compilateur GAML reconnaît et compile de tels programmes, il constitue la première réalisation complète de ML paresseux.

Enfin, un article donné en annexe décrit le compilateur et l'environnement d'exécution de GAML. Les architectures hôtes de ce système sont variées, les programmes GAML pouvant, en particulier, s'exécuter sur machine parallèle. Afin de préserver l'unité du système GAML et de conserver une sémantique simple au langage, le parallélisme intervient uniquement pour accélérer l'exécution de programmes séquentiels. Les principaux aspects abordés sont d'ordre assez concret comme la gestion mémoire ou l'interaction entre les phases d'optimisation du compilateur et le parallélisme.

GAML n'est pas CAML

Deux algorithmes sont présentés en détail dans cette thèse, plutôt que des les décrire informellement ou d'utiliser un pseudo-PASCAL quelconque, j'ai choisi de les représenter en CAML, la version en appel par valeur du langage ML développée à l'INRIA [Wei 90]. Voici donc une description rapide des constructions CAML utilisées.

Les valeurs CAML prédéfinies, sont les booléens `true` et `false` (type `bool`) ; les entiers relatifs ..., `-1`, `0`, `1`, ... (type `int`) ; les paires et les listes (types α `&` β et α `list`). Les paires sont notées normalement, par exemple `(1,(true,2))` est une paire dont la deuxième composante est aussi une paire. Dans ce cas précis, on utilise aussi une représentation abrégée `(1,true,2)`. Les listes sont notées sous la forme `[1;2;3]` ou `1::2::3::[]`, le *cons* étant représenté par un `::` infixe et le *nil* par la constante `[]`. Il existe aussi des chaînes de caractères notées classiquement `"chaine"`. Ces objets de base sont exploités par les primitives usuelles, arithmétique, construction `if` e_c `then` e_1 `else` e_2...

Une caractéristique importante de CAML est son appel par filtrage dont la syntaxe est assez libre. Ainsi, les trois définitions suivantes définissent la même fonction :

```
let F (x,y) = (x+y,x-y)        let F = function (x,y) → (x+y,x-y)
          let F c = match c with (x,y) → (x+y,x-y)
```

Les définitions par filtrage peuvent avoir plusieurs cas ou *clauses*. Chaque clause est composée d'un *motif* et d'un *résultat* séparés par une flèche "→". Les diverses clauses d'un filtrage sont séparées par une barre verticale "|". Ainsi, la fonction `swap` déjà rencontrée s'écrit comme suit :

```
let rec swap = function
  []       → []
| [x]      → [x]
| x::y::l  → y::x::swap l
```

Lorsqu'un motif contient une variable qui n'est pas reprise dans le résultat correspondant, une simplification communément pratiquée est de remplacer cette variable par la notation "_". On peut ainsi écrire les fonctions `fst` et `snd` qui sélectionnent chaque composante d'un couple :

```
let fst (x,_) = x                let snd (_,x) = x
```

Les définitions par filtrage peuvent être formellement ambiguës. Cette ambiguïté est levée en essayant les différentes clauses dans l'ordre où elles sont écrites et en retenant la première qui convient. On pourra donc maintenant écrire :

```
let G (0,_) = 0
|   G (_,0) = 1
```

Ici, l'appel `G (0,0)` vaut `0` et l'appel `G (1,0)` vaut `1`. Le mécanisme de filtrage paresseux avec priorité de GAML est plus complexe, la principale difficulté étant que l'argument de la fonction `G` n'est plus nécessairement une forme normale.

En raison de la sémantique en appel par valeur de CAML, la liaison locale `let`...`in`... exprime l'enchaînement des calculs. Dans l'expression `let c = F (1,1) in fst c`, la fonction `F` est d'abord appelée une fois avec l'argument `(1,1)`, elle renvoie la paire `(2,0)` ; puis c'est la fonction `fst` qui est appelée avec l'argument `(2,0)` pour finalement renvoyer le résultat `2`.

Enfin, la syntaxe du filtrage s'applique aussi à la liaison locale, ce qui permet de récupérer directement les résultats des fonctions qui renvoient des valeurs composites. Ainsi, on pourra directement écrire `let (x,_) = F (1,1) in x`, au lieu de `let c = F (1,1) in fst c`.

Chapitre 1

Réduction de termes

1.1 Termes sans étiquettes

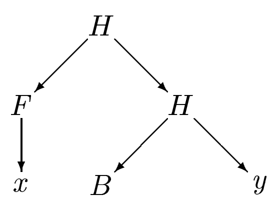
1.1.1 Ensemble de termes

Dans tout ce travail, on se donne un ensemble de symboles fonctionnels $\mathcal{F}$. Chaque symbole f de $\mathcal{F}$ possède une *arité* qui est un entier naturel représentant le nombre d'arguments du symbole f. On peut donc voir l'ensemble $\mathcal{F}$ comme la réunion d'ensembles de symboles $\mathcal{F}_a$, où $\mathcal{F}_a$ est un ensemble de symboles d'arité a. L'ensemble $\mathcal{F}_0$ regroupe les symboles fonctionnels *nullaires*, *zéroaires*, l'ensemble $\mathcal{F}_1$ les symboles *unaires*, l'ensemble $\mathcal{F}_2$ les symboles *binaires*... On se donne aussi un ensemble de variables $\mathcal{X}$, une variable étant généralement notée x. On considère ensuite l'ensemble $\mathcal{T}$ des termes formés à partir des symboles fonctionnels de $\mathcal{F}$ et des variables de $\mathcal{X}$:

Termes $\mathcal{T}$: $M ::= x \mid f(M_1, M_2, \ldots, M_a)$ où a est l'arité de f

Posons par exemple $\mathcal{X} = \{x, y, z\}$, $\mathcal{F}_0 = \{A, B\}$, $\mathcal{F}_1 = \{F, G\}$ et $\mathcal{F}_2 = \{H\}$. Un terme bien formé est $H(F(x), A())$, tandis que $F(F(x), A())$ n'est pas un terme car la règle de l'arité n'est pas respectée. Dans un terme de la forme $f(M_1, M_2, \ldots, M_a)$, le symbole f est le *symbole de tête* et les M_i sont les *fils*. On adopte une notation simplifiée en sous-entendant la paire de parenthèses qui suit les symboles d'arité nulle. C'est à dire que l'on écrit A au lieu de $A()$. Pour tout terme M, on note $\mathcal{V}(M)$ l'ensemble des variables de M. Par exemple, on a $\mathcal{V}(H(F(x), H(B, y))) = \{x, y\}$. Si $\mathcal{V}(M)$ est vide, alors M est un terme *clos*. Si la même variable n'apparaît pas plus d'une fois dans un terme, ce terme est dit *linéaire*. Par exemple, le terme $H(F(x), H(B, x))$ n'est pas linéaire car il contient deux fois la variable x.

Les termes de $\mathcal{T}$ peuvent être vus comme des arbres, ainsi on peut représenter le terme $H(F(x), H(B, y))$ par la figure :



Les *occurrences* sont des séquences d'entiers qui codent l'accès aux sous-termes.

Définition 1.1.1 (Occurrences dans un terme) *Soit un terme M. L'ensemble $\mathcal{O}(M)$ de ses* *occurrences est défini par :*

$$\begin{cases} \mathcal{O}(x) = \{\epsilon\} \\ \mathcal{O}(f(M_1,\ldots,M_a)) = \{\epsilon\} \cup \{\, 1.u \mid u \in \mathcal{O}(M_1)\,\} \cup \ldots \cup \{\, a.u \mid u \in \mathcal{O}(M_a)\,\} \end{cases}$$

On a par exemple, $\mathcal{O}(H(F(x), H(B,y))) = \{\epsilon, 1, 1.1, 2.1, 2.2\}$.

Soient u et v deux occurrences, si $u = v.w$, alors v est un *préfixe* de u et on note $v \preceq u$. Si l'occurrence w n'est pas l'occurrence vide ϵ, alors v est un préfixe *propre* de u (noté $v \prec u$). Si u n'est pas un préfixe de v ni v un préfixe de u, alors les occurrences u et v sont *disjointes* et on note $u \mid v$. Étant donné un ensemble fini d'occurrences U, ont note $\mathit{inf}_{\preceq}(U)$ l'ensemble des occurrences extérieures de l'ensemble U. Par définition, $\mathit{inf}_{\preceq}(U)$ est l'ensemble des point minimaux de U pour l'ordre préfixe sur les occurrences :

$$\mathit{inf}_{\preceq}(U) = \{\, u \in U \mid \forall v \in U,\ v \preceq u \Rightarrow v = u\,\}$$

Il est immédiat que $\mathit{inf}_{\preceq}(U)$ regroupe des occurrences deux à deux disjointes. De plus, l'ensemble U étant fini, on montre facilement l'implication $U \neq \emptyset \Rightarrow \mathit{inf}_{\preceq}(U) \neq \emptyset$ par induction sur le cardinal de U. L'ensemble $\mathit{inf}_{\preceq}(U)$ est donc vide, si et seulement si l'ensemble U est vide.

On étend l'ordre partiel $\preceq$ en en ordre total sur les occurrences. Si les occurrences u et v sont disjointes alors elles admettent un préfixe commun maximum w, tel que $u = w.n_u.u'$ et $v = w.n_v.v'$ et $n_u \neq n_v$. Dans ce dernier cas, si l'entier n_u est strictement plus petit que l'entier n_v, alors u est *à gauche* de v. L'ordre *extérieur-gauche* $\leq_{lo}$ sur les occurrence (leftmost outermost ordering) est un ordre total défini par : $u \leq_{lo} v$, si et seulement si u est un préfixe de v ou u est à gauche de v. Pour simplifier, on appelle order gauche l'ordre extérieur gauche. Dans l'ensemble $\{\epsilon, 1, 1.1, 2.1, 2.2\}$ du paragraphe précédent les occurrences sont classées selon l'ordre gauche croissant.

Notation : Soit u une occurrence. Soient U et V deux ensembles d'occurrences. On pose :

$$u.V = \{i.u \mid u \in U\} \quad U.V = \{u.v \mid u \in U,\ v \in V\}$$

On peut préciser la relation entre occurrences et sous-termes :

Définition 1.1.2 (Sous-termes) *Soit M un terme quelconque. On définit le sous-terme M/u situé à l'occurrence u dans le terme M de la manière suivante :*

- *Si u est l'occurrence vide, alors on pose $M/\epsilon = M$.*
- *Sinon, le terme M est de la forme $M = f(M_1, M_2, \ldots, M_a)$ et, pour tout entier i compris entre 1 et a, on pose $M/i.u = M_i/u$.*

Par exemple, on a $H(F(x), H(B,y))/1.1 = x$ et $H(F(x), H(B,y))/2 = H(B,y)$. Une occurrence u est une occurrence *dure* ou *intérieure* de M et l'on note $u \in \overline{\mathcal{O}}(M)$, quand le sous-terme M/u n'est pas une variable. Dans le cas contraire (i.e., si le sous-terme M/u est une variable), on parle d'occurrence molle. Par exemple, on a $\overline{\mathcal{O}}(H(F(x), H(B,y))) = \{\epsilon, 1, 2, 2.1\}$. Dans ce cas, on parlera aussi parfois de sous-terme dur. La figure 1.1 présente un terme sous forme d'un arbre dont chaque sous-arbre est décoré par son occurrence :

Remarque 1.1.1 *On confond souvent l'occurrence d'un sous-terme avec ce sous-terme lui même, afin d'alléger la terminologie.*

On définit aussi le *remplacement* d'un sous-terme par un autre.

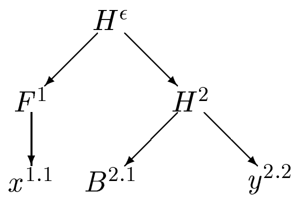
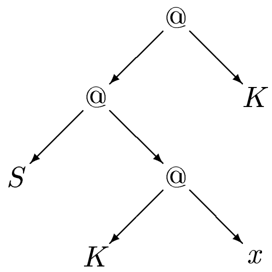


Figure 1.1 :

Définition 1.1.3 *Soient M et N deux termes. Soit encore u une occurrence de $\mathcal{O}(M)$. Le remplacement par N du sous-terme M/u, noté $M[u \leftarrow N]$, est défini récursivement par :*

- $M[\epsilon \leftarrow N] = N$.
- $f(M_1, M_2, \ldots, M_i, \ldots, M_a)[i.u \leftarrow N] = f(M_1, M_2, \ldots, M_i[u \leftarrow N], \ldots, M_a)$.

Sont maintenant décrits deux ensembles de termes qui vont servir d'exemples. Le premier de ces ensembles est composé des *termes applicatifs*. Soit un ensemble $\mathcal{F}_0$ de symboles zéroaires arbitraires que l'on appelle des *combinateurs* —$\mathcal{F}_0 = \{S, K\}$, par exemple—. On se donne aussi un unique symbole binaire @ que l'on appelle application —dans tous les cas, on pose $\mathcal{F}_2 = \{@\}$—. Les termes de $\mathcal{T}$ sont des applications de combinateurs à un nombre quelconque d'arguments. Par exemple, le terme $M = @(@(S, @(K, x)), K)$ est l'application du combinateur S à deux arguments codée à l'aide de deux nœuds d'application binaire (on parle parfois de *curryfication*). On utilise la notion simplifiée $M = S\ (K\ x)\ K$ qui laisse sous-entendues les applications @ et pose que cette application implicite associe à gauche. La représentation des termes sous forme d'arbre est la plus parlante, ainsi on peut écrire le terme M comme :

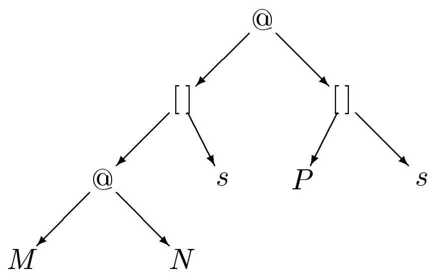


Un autre exemple significatif est celui de l'ensemble Λ-Σ des λ-termes avec substitutions explicites. Ces termes sont définis par :

λ-termes	Λ-Σ	$M ::= \mathtt{1} \mid \mathtt{2} \mid \ldots \mid M\ N \mid \lambda M \mid M[s] \mid X$
substitutions		$s ::= M.s \mid id \mid x$

La séquence infinie $\mathtt{1} \mid \mathtt{2} \mid \ldots$ représente tous les entiers strictement positifs. Ces entiers codent les variables du λ-calcul selon le mécanisme des indices de De Bruijn [Bru 72]. Chaque variable liée du λ-calcul classique est remplacée par un indice qui fait référence à l'abstraction qui la lie. Comme dans les indices de De Bruijn traditionnels, la valeur de l'indice correspond au nombre d'abstractions (i.e. de λ) entre l'occurrence de variable considérée et l'abstraction liante. Par exemple, les λ-termes $\lambda x.x$ and $\lambda x.\lambda y.x$ du λ-calcul classique deviennent $\lambda\mathtt{1}$ et $\lambda\lambda\mathtt{2}$ en notation de De Bruijn. Les termes de la forme $M[s]$ sont des *fermetures*, dont le fils gauche M est le *code* et le fils droit s l'*environnement*.

L'ensemble $\mathcal{F}$ de symboles fonctionnels est ici égal à la réunion des trois ensembles $\mathcal{F}_0 = \{id, \mathtt{1}, \mathtt{2}, \ldots\}$, $\mathcal{F}_1 = \{\lambda\}$ and $\mathcal{F}_2 = \{@, [\,], .\}$. L'application binaire @ est laissée implicite comme dans le cas des combinateurs. Il y a d'autres particularités syntaxiques : dans les fermetures de la forme $M[s]$ ou dans les *environnement composites* de la forme $M.s$. Ces particularités sont des commodités de notation et la structures des λ-termes est mieux comprise si on voit ceux ci comme des arbres. Par exemple, au terme $(M\ N)[s]\ P[s]$ correspond l'arbre :



Notons enfin que la définition de Λ-Σ est plus riche que la définition générale des termes, car elle utilise deux sortes de termes : les λ-termes et les substitutions qui se composent selon des règles précises propres à chaque symbole fonctionnel. Par exemple, le symbole de fermeture $[\,]$ définit un λ-terme, son fils gauche M est aussi un λ-terme, tandis que son fils droit s est une substitution. Je choisis de développer une théorie générale des réécritures de termes sans tenir compte de cet aspect de "typage" qui n'est pas très important. C'est à dire que, le cas échéant, seront implicitement seuls admis les termes "bien typés".

1.1.2 Systèmes de réécriture

Le réécriture de termes est une modélisation du calcul sur les termes. Elle est le fondement de la sémantique opérationnelle des langages de programmation. En ce qui nous concerne, l'introduction de la réécriture sur les termes à combinateurs ou les λ-termes avec substitutions explicites est un modèle des calculs possibles sur des formes compilées de ces langages (voir [Joh 87, Aug 87, Pey 87] pour le cas de ML et des combinateurs, voir [Ler 90] pour le cas des substitutions explicites).

Un *système de réécriture de termes* $\Sigma = \{\ldots, \alpha \to \beta, \ldots\}$ est la donnée d'un ensemble de *règles de réécriture*, qui sont des couples de termes. On note une règle de réécriture $\alpha \to \beta$, où le terme α est le *membre gauche* de la règle, tandis que le terme β est son *membre droit* . Les variables du membre droit β sont obligatoirement des variables du membre gauche α ($\mathcal{V}(\beta) \subseteq \mathcal{V}(\alpha)$). Si β est réduit à une simple variable, alors la règle $\alpha \to \beta$ est une *projection.* Si α (respectivement β) est un terme linéaire alors la règle $\alpha \to \beta$ est *linéaire à gauche* (respectivement *linéaire à droite*). Un système dont toutes les règles sont linéaires à gauche est un *système linéaire.*

Les substitutions sont les morphismes sur l'ensemble des termes.

Définition 1.1.4 (Substitutions) *Une substitution est une fonction partielle de $\mathcal{T}$ dans $\mathcal{T}$, telle que, pour tous les termes de la forme $f(M_1, M_2, \ldots, M_a)$, on a l'égalité :*

$$\sigma(f(M_1, M_2, \ldots, M_a)) = f(\sigma(M_1), \sigma(M_2), \ldots, \sigma(M_a))$$

Une substitution σ est donc uniquement déterminée par sa valeur sur un ensemble de variables qui est son support ou son domaine, noté $\mathcal{D}(\sigma)$. Étant données deux substitutions σ et ρ dont les supports sont disjoints, on note $\sigma \cup \rho$ la substitution union des substitutions σ et ρ définie par :

$$\forall x \in \mathcal{D}(\sigma),\ (\sigma \cup \rho)(x) = \sigma(x) \qquad\qquad \forall x \in \mathcal{D}(\rho),\ (\sigma \cup \rho)(x) = \rho(x)$$

Les substitutions de support fini sont souvent notées comme des environnements. C'est à dire que la substitution qui à la variable x_1 associe le terme M_1, à la variable x_2 le terme M_2, ... et a la variable x_n le terme M_n est notée $[x_1\backslash M_1, x_2\backslash M_2, \ldots, x_n\backslash M_n]$. L'application d'une telle substitution à un terme M quelconque est notée $M[x_1\backslash M_1, x_2\backslash M_2, \ldots, x_n\backslash M_n]$.

La relation de *réduction simple* est la relation $M \to M'$, définie sur l'ensemble des termes par les axiomes et règles d'inférence suivants :

$$\sigma(\alpha_i) \to \sigma(\beta_i) \ {\scriptstyle(\mathrm{red}_i)} \qquad \frac{M_i \to M'_i \text{ avec } 1 \leq i \leq a}{f(M_1, \ldots, M_i, \ldots, M_a) \to f(M_1, \ldots, M'_i, \ldots, M_a)}\ {\scriptstyle(\mathrm{rec}_i^f)}$$

Étant entendu, qu'il y a en fait un axiome (red_i) par règle de réécriture $\alpha_i \to \beta_i$ et par substitution, et une règle d'inférence (rec_i^f) par symbole fonctionnel f d'arité a et et par indice d'argument i compris entre 1 et a.

Chaque relation $A{:}M \to M'$ est donnée par un arbre de preuve, cet arbre n'a qu'une seule feuille qui correspond à l'application d'un axiome du type (red_i) à un sous-terme de M de la forme $\sigma(\alpha_i)$. Un tel sous-terme est un *radical.* L'occurrence u dans M de ce sous-terme est suffisante pour caractériser la réduction A. On écrit alors $A{:}M \xrightarrow{u} M'$ et on dit que A est la réduction qui *contracte* le radical $\sigma(\alpha)$ situé à l'occurrence u dans le terme M. Remarquons que l'on a l'égalité $M' = M[u \leftarrow \sigma(\beta)]$.

La relation de *dérivation*, notée $M \to^* N$, est la fermeture réflexive et transitive de la relation de réduction $\to$. La *longueur* $|A|$ d'une dérivation A est le nombre d'étapes de réduction qui la composent. La dérivation vide O est la dérivation de longueur zéro. Soient A: $M{\to^*}N$ et B: $N{\to^*}P$ deux dérivations, la *concaténation* AB de A et de B est la dérivation composée $M \to^* N \to^* P$.

Étant donné un terme M, on note $\mathcal{R}(M)$ l'ensemble des occurrences des radicaux du terme M. Parmi ces occurrences de radicaux, on distingue celles des radicaux *les plus extérieurs* ou simplement *extérieurs* qui forment un ensemble noté $\mathcal{E}xt(M)$. C'est à dire que l'on pose $\mathcal{E}xt(M) = inf_{\preceq}(\mathcal{R}(M))$. Un terme qui ne contient pas de radical ($\mathcal{R}(M) = \emptyset$)) est dit *en forme normale*, la dérivation vide est la seule dérivation issue d'une forme normale. Étant donné un terme M, si il existe une dérivation qui va du terme M à un terme N qui est en forme normale, alors on dit que N est une (la) forme normale de M.

Dans les systèmes à combinateurs, on associe une et une seule règle de réécriture à chaque symbole de combinateur et les membres gauches de ces règles ont une forme générale imposée. Étant donné un symbole de combinateur F, le membre gauche de la règle $\alpha_F \to \beta_F$ associée au combinateur F est toujours de la forme $\alpha_F = F\ x_1\ x_2 \ldots x_n$. Une règle de combinateur représente donc une définition de fonction. L'entier n est *l'arité* du combinateur F, à ne pas confondre avec l'arité du symbole F dans le système de réécriture sous-jacent qui est toujours nulle. Par exemple, on code la *logique combinatoire* [Hin 86] en se donnant les deux symboles de combinateurs S et K, ainsi que les deux règles :

$$\begin{array}{c} S\ f\ g\ x \to f\ x\ (g\ x) \\ K\ x\ y \to x \end{array}$$

L'arité du combinateur S est égale à trois, tandis que l'arité du combinateur K est égale à deux. Notons en outre que la règle du combinateur S n'est pas linéaire à droite et que la règle du combinateur K est une projection. La relation de réécriture simple s'énonce ensuite naturellement :

$$\sigma(S\ f\ g\ x) \to \sigma(f\ x\ (g\ x)) \qquad \sigma(K\ x\ y) \to \sigma(x)$$

$$\frac{M \to M'}{M\ N \to M'\ N} \qquad\qquad \frac{N \to N'}{M\ N \to M\ N'}$$

Si on considère le terme $S\ K\ K\ K$, on a la dérivation de longueur deux :

$$S\ K\ K\ K \overset{\epsilon}{\to} K\ K\ (K\ K) \overset{\epsilon}{\to} K$$

La logique combinatoire est une modélisation possible du λ-calcul faible [Tur 79, Hin 77]. La procédure de traduction des λ-termes en termes applicatifs de la logique combinatoire, ou *procédure d'abstraction*, n'est pas conforme à l'intuition et correspond à un codage assez arbitraire, par ailleurs la procédure d'abstraction peut produire des termes combinatoires dont la taille croît comme le carré de la taille du λ-terme traduit. Dans la section suivante, une représentation plus naturelle du λ-calcul faible est décrite. Cependant, la définition et la mise en œuvre du λ-calcul faible à l'aide d'une traduction vers un système à combinateurs reste une technique populaire. Elle est applicable en pratique grâce aux améliorations suivantes : des jeux de combinateurs plus larges que les seuls S et K peuvent être utilisés ; ou alors, les combinateurs peuvent dépendre étroitement du λ-terme à traduire (supercombinateurs de [Hug 84]). Le livre [Pey 87] passe en revue toutes ces techniques.

1.1.3 Réécriture conditionnelle

Je présente ici une extension simple et utile du modèle traditionnel de la réécriture en réécriture *conditionnelle*. Il est en effet des cas où la réécriture est interdite dans certains sous-termes. Un exemple significatif est la réécriture "faible" dans le λ-calcul, que l'on présente de façon informelle en disant que la réécriture est interdite "dans le corps des fonctions" ou "sous les λ". Cette restriction correspond à l'intuition que si le λ-calcul est le modèle du langage cible d'un compilateur fonctionnel, alors les fonctions sont en fait du code machine qui ne doit normalement pas être modifié.

Le λ-calcul en lui même se prête mal à des modélisation de calculs de bas niveau. En effet, le λ-calcul traditionnel englobe toutes les opérations qu'entraîne l'appel de fonction en une seule opération : la β-réduction. En particulier, la β-réduction n'a pas de coût bien fixé en termes de nombres d'instruction machines exécutées, puisque son coût est fonction de la taille du corps des fonctions. Il est plus réaliste et conforme aux réalisations en machine de séparer le processus de liaison des variables (ou paramètres formels) aux arguments (ou paramètres effectifs), de la substitution des uns par les autres dans le corps des fonctions. Cette substitution étant elle même découpée en opérations plus élémentaires dont chacune correspond *grosso-modo* à une instruction machine. Le λ-calcul avec substitutions explicites est un formalisme conforme à ces intuitions. Pour une étude complète du λ-calcul avec substitutions explicites, je renvoie à [Aba 90, Har 92], on y trouvera en particulier la description exacte des relations entre le λ-calcul avec substitutions explicites et le λ-calcul tout court. Pour décrire les réductions faibles, un système simplifié de quatre règles de réécriture est en fait suffisant :

$$\begin{array}{lrcl}
\text{(WeakBeta)} & (\lambda M)[s]\ N & \to & M[N.s] \\
\text{(App)} & (M\ N)[s] & \to & M[s]\ N[s] \\
\text{(FVar)} & \mathtt{1}[M.s] & \to & M \\
\text{(RVar)} & \mathtt{n+1}[M.s] & \to & \mathtt{n}[s]
\end{array}$$

La première règle (WeakBeta) réalise la liaison du paramètre formel au paramètre effectif, tandis que les trois autres sont les mécanismes élémentaires dont la composition réalise la substitution. La dernière règle (RVar) est à comprendre comme une infinité dénombrable de règles : $\mathtt{2}[M.s] \to \mathtt{1}[s]$, $\mathtt{3}[M.s] \to \mathtt{2}[s]$, ... La relation de réduction simple est engendrée par les axiomes et règles d'inférence suivants :

$$\sigma(\alpha_i) \to \sigma(\beta_i) \ \text{(Red)} \qquad \frac{s \to s'}{M[s] \to M[s']} \ \text{(AppSubst2)}$$

$$\frac{M \to M'}{M\ N \to M'\ N} \ \text{(App1)} \qquad \frac{N \to N'}{M\ N \to M\ N'} \ \text{(App2)}$$

$$\frac{M \to M'}{M.s \to M'.s} \ \text{(Cons1)} \qquad \frac{s \to s'}{M.s \to M.s'} \ \text{(Cons2)}$$

Par rapport à un système de réécriture non-conditionnel, on constate qu'il manque des règles d'inférence. En effet, il n'est pas possible de réduire sous un λ ou dans le code M d'une fermeture $M[s]$. Ces deux restrictions sont conformes à la définition informelle de la réduction faible et au fait que le code d'une fonction n'est pas modifiable. Ce type de restriction peut être décrit de façon compacte par une fonction de domaine dom(), qui, à chaque symbole fonctionnel, associe l'ensembles des indices *autorisés* à partir de ce symbole. Par exemple, on a ici $\mathrm{dom}(@) = \{1, 2\}$, $\mathrm{dom}([\,]) = \{2\}$, $\mathrm{dom}(.) = \{1, 2\}$ et $\mathrm{dom}(\lambda) = \emptyset$.

On appelle $\lambda\sigma$ faible le système de réécriture conditionnel, complètement décrit par les quatre règles de réécriture et la fonction de domaine ci-dessus. Un aspect annexe de $\lambda\sigma$ faible concerne les types, il va sans dire que les types sont conservés par réduction. C'est à dire qu'un λ-terme se réécrit en un λ-terme et une substitution en une autre substitution.

Le système $\lambda\sigma$ faible est surtout une présentation cohérente du λ-calcul faible. En effet, comme on va le voir, $\lambda\sigma$ faible est un système déterministe dans le sens qu'il possède la propriété de Church-Rosser. Mieux, comme [Ler 90], on peut montrer que diverses machines abstraites de réduction faible du λ-calcul calculent une dérivation particulière du système $\lambda\sigma$ faible. Ainsi, la machine de Krivine [Kri 85, Cré 91] calcule en réduisant le radical le plus extérieur et le plus à gauche dans le terme courant, jusqu'à atteindre une fermeture $(\lambda M)[s]$ ou un terme de la forme $M_1\ M_2\ \ldots\ M_n$, où M_1 est un terme irréductible de la forme $\mathtt{n}$, $\mathtt{n}[s]$ ou X.

Plus généralement, étant donné un ensemble de termes $\mathcal{T}$, un système de réécriture conditionnel Σ est complètement décrit par la donnée d'un ensemble de règles de réécriture $\{\ldots, \alpha_i \to \beta_i, \ldots\}$ et d'une fonction de domaine dom(). Dans le cas d'un système non-conditionnel, la fonction de domaine n'exclut aucun indice, c'est à dire que, pour tout symbole fonctionnel f, on a $\mathrm{dom}(f) = \{1, 2, \ldots a\}$, où l'entier a est l'arité de f.

La notion d'indice autorisé s'étend naturellement aux occurrences.

Définition 1.1.5 (Statut d'une occurrence) *Soient M un terme et $u = n_1.n_2\ldots n_d$ une occurrence dans M. L'occurrence u est autorisée, si et seulement si, pour tout entier i de l'intervalle $[1\ldots d]$, l'indice n_i est dans le domaine du symbole de tête du sous-terme $M/n_1.n_2\ldots n_{i-1}$. Dans le cas contraire, l'occurrence u est interdite.*

Par exemple, dans $\lambda\sigma$ faible, l'ensemble des occurrences autorisées du terme $((\lambda M)[s]\ N)[t]$ est $\{\epsilon, 2\}$, puisque $\mathrm{dom}([]) = \{2\}$. Toutes les autres occurrences de ce termes sont préfixées par 1 et sont donc interdites ($1 \notin \mathrm{dom}([])$). Les sous-termes sont autorisés ou interdits selon le statut de leur occurrence. Dans l'ensemble $\mathcal{V}(M)$ des variables d'un terme M, on distingue alors le sous-ensemble $\mathcal{V}_a(M)$ des variables autorisées de Met le sous-ensemble $\mathcal{V}_i(M)$ des variables interdites de M. On a toujours $\mathcal{V}(M) = \mathcal{V}_a(M) \cup \mathcal{V}_i(M)$. Toutefois, il est possible d'avoir $\mathcal{V}_a(M) \cap \mathcal{V}_i(M) \neq \emptyset$, quand le terme M n'est pas linéaire.

Un radical est toujours un sous-terme auquel on peut appliquer un axiome (red), il faut maintenant, en plus de sa forme, tenir compte de son statut.

Définition 1.1.6 (Radicaux dans les systèmes conditionnels) *Soient M un terme et R un sous-terme d'occurrence u dans M. Le sous-terme R est un radical, si et seulement si les deux conditions suivantes sont satisfaites :*

1. *Il existe un membre gauche de règle α et une substitution σ, avec $R = \sigma(\alpha)$.*
2. *L'occurrence u est autorisée.*

Dans un radical $R = \sigma(\alpha)$ le membre gauche en jeu α est le *nom* du radical R. Une occurrence v de la forme $v = u.w$, où l'occurrence w est dure dans le nom α aboutit *à l'intérieur* du nom du radical R. Conformément à la remarque 1.1.1, on confondra souvent un radical et son occurrence.

1.1.4 Résidus

Les radicaux et plus généralement les sous-termes peuvent être suivis par delà les réductions.

Définition 1.1.7 (Résidus par réduction) *Soit A: $M \xrightarrow{u} N$ une réduction selon la règle $\alpha \to \beta$. À toute occurrence v du terme M, on associe l'ensemble $v\backslash A$ de ses résidus dans le terme N par la réduction A. L'ensemble $v\backslash A$ est un ensemble d'occurrences du terme N défini de la façon suivante :*

1. *Si $v \prec u$ ou $u \mid v$, on pose $v\backslash A = \{v\}$.*
2. *Si $v = u.w$ avec $w \in \overline{\mathcal{O}}(\alpha)$, on pose $v\backslash A = \emptyset$.*
3. *Si $v = u.u_x.w$, où u_x est l'occurrence dans α d'une variable x, on pose*

$$v\backslash A = u.\{\, v_x \in \mathcal{O}(\beta) \mid \beta/v_x = x \,\}.w$$

Si v est l'occurrence d'un radical R du terme M, on parle de résidus de radical que l'on note $R\backslash A$. Une occurrence qui n'a pas de résidu parce qu'elle aboutit à l'intérieur du nom du radical contracté (cas *2*) est *contractée*. Une occurrence qui n'a pas de résidu parce qu'une variable de α n'apparaît pas dans β (cas *3*, avec $\{\, v_x \in \mathcal{O}(\beta) \mid \beta/v_x = x \,\} = \emptyset$) est *effacée*. Une occurrence qui a plusieurs résidus parce que β n'est pas linéaire (cas *3*) est *dupliquée*. Enfin, les occurrences du terme N qui ne sont pas résidu d'une occurrence du terme M sont *créées* par la réduction A. Les occurrences créées sont exactement les occurrences de la forme $u.w$, où w est une occurrence intérieure du membre droit β. Par conséquent, les règles de projection ne créent pas d'occurrences. À titre d'exemple, on se place dans $\lambda\sigma$ faible et on considère les trois réductions successives selon les règles (App), (WeakBeta) et (FVar) :

$$A\colon (\lambda\mathtt{1}\ \mathtt{1})[s] \xrightarrow{\epsilon} (\lambda\mathtt{1})[s]\ \mathtt{1}[s] \quad B\colon (\lambda\mathtt{1})[s]\ \mathtt{1}[s] \xrightarrow{\epsilon} \mathtt{1}[\mathtt{1}[s].s] \quad C\colon \mathtt{1}[\mathtt{1}[s].s] \xrightarrow{\epsilon} \mathtt{1}[s]$$

On suit les résidus de l'occurrence 2 du sous-terme s. On a $2\backslash A = \{1.2, 2.2\}$; puis $1.2\backslash B = \{2.2\}$ et $2.2\backslash B = \{2.1.2\}$; et enfin $2.2\backslash C = \emptyset$ et $2.1.2\backslash C = 2$. L'occurrence 2 est dupliquée par la réduction A, tandis que l'occurrence 2.2 est effacée par la réduction C.

Soient une réduction A:$M \to N$ et un ensemble $U = \{u_1, u_2, \ldots, u_n\}$ d'occurrences du terme M. Par définition, l'ensemble $U \setminus A$ des résidus de l'ensemble U par la réduction A est composé de tous les résidus des occurrences de U.

$$U \setminus A = u_1 \setminus A \cup u_2 \setminus A \cup \ldots \cup u_n \setminus A$$

Dans le cas où l'ensemble U n'a qu'un seul élément u, on continue à noter $u\backslash A$ pour $\{u\}\backslash A$. Soit maintenant une dérivation $A = A_1A_2$ issue d'un terme M et U un ensemble d'occurrence de M. On pose $U\backslash A_1A_2 = (U\backslash A_1)A_2$. Dans l'exemple du paragraphe précédent on a, $2\backslash ABC = \{2\}$.

1.1.5 Systèmes orthogonaux

Mon intention est de modéliser des calculs par la réécriture de termes. Pour être raisonnable, je pense qu'un tel modèle doit répondre à deux contraintes :

1. Le processus de calcul doit être local.
2. Les calculs doivent être non-ambigus.

Par la première de ces exigences, les radicaux doivent être reconnus en examinant leurs noms uniquement. Ceci impose aux membres gauche de règles d'être des termes linéaires. En effet, soit un terme linéaire α et un terme quelconque M. Pour savoir si il existe une substitution σ, telle que $\sigma(\alpha) = M$, il suffit d'appliquer la procédure suivante :

1. Pour chaque occurrence u dure de α, examiner si u est une occurrence de M et si oui, vérifier que les symboles de tête situés à l'occurrence u dans α et dans M sont les mêmes.
2. Ensuite, pour toute variable x d'occurrence u_x dans α, on pose $\sigma(x) = M/u_x$.

Bien entendu, la deuxième étape de la procédure ci-dessus ne serait pas suffisante si le terme α n'était pas linéaire, auquel cas il faudrait comparer certains sous-termes de M entre eux.

$$\begin{tikzcd} & M \arrow[dl, "A", "*"'] \arrow[dr, "B"', "*"] & \\ N \arrow[dr, "C", "*"'] & & P \arrow[dl, "D"', "*"] \\ & Q & \end{tikzcd}$$

Figure 1.2 : Propriété de Church-Rosser

Traditionnellement, la seconde de ces exigences se traduit par la propriété de confluence encore appelée propriété de Church-Rosser : pour tout terme M et tout couple de dérivations issues du terme M, A: $M \to^* N$ et B: $M \to^* P$, il existe un terme Q ainsi que deux dérivations C: $N \to^* Q$ et D: $P \to^* Q$ qui referment le "diagramme en diamant" issu de M (cf. figure 1.2). On dit alors que les dérivations A et B sont *confluentes*, tandis que les dérivations C et D (ou AC et BD) sont *convergentes* en Q. On voit bien comment la propriété de Church-Rosser traduit le déterminisme des calculs : on ne peut pas calculer de deux façons qui soient différentes au point d'aboutir nécessairement à des résultats différents. En particulier, dans un système qui possède la propriété de Church-Rosser, tout terme admet au plus une forme normale.

Je pose ici une condition encore plus restrictive en interdisant les ambiguïtés locales. Soit un terme M et deux radicaux, $R_1 = \sigma_1(\alpha_1)$ et $R_2 = \sigma_2(\alpha_2)$ d'occurrences respectives u_1 et u_2 dans M. On suppose en outre que les radicaux R_1 et R_2 sont distincts au sens que l'on a $u_1 \neq u_2$ ou $\alpha_1 \to \beta_1 \neq \alpha_2 \to \beta_2$. Par définition, les radicaux R_1 et R_2 se recouvrent, si et seulement si il existe une occurrence v_1, dure dans α_1, telle que l'on a $u_2 = u_1.v_1$, ou bien une occurrence v_2, dure dans α_2, telle que l'on a $u_1 = u_2.v_2$. C'est à dire que les radicaux R_1 et R_2 se recouvrent dans le terme M, si et seulement si il existe une occurrence de M qui aboutit à l'intérieur des deux noms α_1 et α_2 de R_1 et R_2. Dans cette thèse, seuls sont étudiés les systèmes pour lesquels il n'existe pas de recouvrements des radicaux dans les termes.

À la place de recouvrements de radicaux dans les termes, on parle aussi de paires critiques entre règles de réécriture. En effet, si il existe un recouvrement, alors il en existe un pour lequel on a $u_1 = \epsilon$. C'est à dire qu'il suffit de considérer les membres gauches de règles pour détecter d'éventuels recouvrements. Les domaines ont une influence sur les paires critiques. Considérons un système qui utiliserait les quatre règles de réécriture de $\lambda\sigma$ faible, sans en adopter les contraintes de domaine. Dans un tel système, les règles (WeakBeta) et (App) forment une paire critique ($u_1 = \epsilon$, $u_2 = 1$) et contredisent la propriété de Church-Rosser :

$$(\lambda M)[s][t]\ N[t] \leftarrow ((\lambda M)[s]\ N)[t] \rightarrow (M[N.s])[t]$$

Avec la contrainte de domaine $\mathrm{dom}([\,]) = \{2\}$, la réduction $\rightarrow$ devient impossible —il n'y a plus de radical à l'occurrence interdite $u_2 = 1$— et la paire critique disparaît.

$$\begin{tikzcd}
 & F(A) \arrow[dl, "\epsilon"'] \arrow[dr, "1"] & \\
G(A) & & F(B) \arrow[dl, "\epsilon"] \\
 & G(B) &
\end{tikzcd}$$

Figure 1.3 : Contre-exemple de Church-Rosser

Enfin, on souhaite éviter que les sous-termes changent de statut au cours des réductions en imposant deux conditions supplémentaires :

1. Les résidus des occurrences autorisées sont des occurrences autorisées.
2. Les résidus des occurrences interdites sont des occurrences interdites.

Du point de vue des règles de réécriture $\alpha \rightarrow \beta$, la première condition se traduit par "les variables interdites du membre droit β sont interdites dans le membre gauche α" et la seconde par "les variables autorisées du membre droit β sont autorisées dans le membre gauche α". On peut rapprocher ces deux contraintes de l'exigence de localité des calculs : la contraction d'un certain radical $\sigma(\alpha)$ n'a pas d'effet "à distance" sur les occurrences de ses sous-termes de la forme $\sigma(x)$. En outre, si la première de ces conditions n'est pas remplie, on peut facilement construire un contre-exemple de la propriété de Church-Rosser. Soient les symboles zéroaires A et B ainsi que les deux symboles unaires F et G. On pose $\mathrm{dom}(F) = \{1\}$, $\mathrm{dom}(G) = \emptyset$ et on se donne les deux règles $A \rightarrow B$ et $F(x) \rightarrow G(x)$. La règle $F(x) \rightarrow G(x)$ invalide la contrainte sur les variables interdites, puisque l'on a, $x \in \mathcal{V}_i(G(x))$ et $x \notin \mathcal{V}_i(F(x))$. Par ailleurs, les membres gauches des règles de ce système sont linéaires et il n'y a pas de paire critique. L'exemple de la figure 1.3 montre que le système considéré n'a pas la propriété de Church-Rosser. De toute évidence, on parviendra à refermer le diagramme en posant $\mathrm{dom}(G) = \{1\}$ et donc $\mathcal{V}_i(G(x)) \subseteq \mathcal{V}_i(F(x)) = \emptyset$.

Pour résumer, tout au long de ce travail on considère comme acquises les trois contraintes u conditions d'orthogonalité suivantes :

1. Les membres gauches α sont des termes linéaires.
2. Les radicaux ne se recouvrent pas dans les termes.

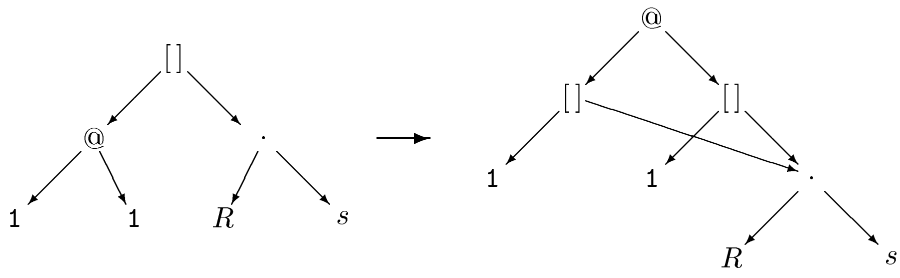


Figure 1.4 : Réduction de graphes

3. Pour toute règle de réécriture $\alpha \to \beta$, on a les deux inclusions :
   (a) $\mathcal{V}_a(\beta) \subseteq \mathcal{V}_a(\alpha)$
   (b) $\mathcal{V}_i(\beta) \subseteq \mathcal{V}_i(\alpha)$

Par la suite, nous verrons que les conditions d'orthogonalité (1), (2) et (3)-(b) entraînent une forme stricte de la propriété de Church-Rosser. La propriété (3)-(a) interviendra pour garantir que les nouveaux radicaux sont créés selon un processus local.

1.2 Étiquettes simples

1.2.1 Le problème du partage

Le formalisme des termes ne peut pas rendre compte du partage souhaitable de certains sous-termes. Ce partage peut être introduit par réécriture et n'est pas nécessairement présent dans le terme initial. En effet, quand on réduit selon une règle qui n'est pas linéaire à droite, une occurrence peut avoir plusieurs résidus. Intuitivement, il semble bien que ces résidus sont des chemins d'accès différents vers un même sous-terme. Pour préciser cette idée, plaçons nous dans le système $\lambda\sigma$ faible et considérons la réduction A:$(\mathtt{1}\ \mathtt{1})[\underline{R.s}] \xrightarrow{\epsilon} \mathtt{1}[\underline{R.s}]\ \mathtt{1}[\underline{R.s}]$. Le sous-terme $R.s$ (souligné) a deux résidus (soulignés également). Si le sous-terme R est un radical, il possède deux radicaux résidus qui devront être contractés séparément, alors qu'en contractant le radical R avant d'effectuer la réduction A, on aurait contracté qu'un seul radical. Une solution à ce problème est d'opérer sur des graphes, la figure 1.4 donne une représentation informelle de ce type de réduction.

Il existe un formalisme de la réécriture de graphes [Ehr 86, Cou 86]. Une approche possible, suivie dans [Bar 87], est de l'utiliser directement, des fonction de traduction étant ensuite fournies pour passer des termes aux graphes et réciproquement. Malheureusement, la réécriture de graphes modélise des transformations bien plus complexes que la simple transposition aux graphes de la réécriture de termes. Par ailleurs les graphes sont des objets mathématiques plus compliqués que les termes. Par exemple, il est difficile de raisonner par récurrence sur les graphes. Ceci me pousse à rejeter la réécriture de graphes en cherchant plutôt à étendre le formalisme de la réécriture de termes.

1.2.2 Termes étiquetés

Une première étape est de représenter les graphes par des termes. À chaque sous-terme est associée une *étiquette* que l'on note en exposant. Les étiquettes sont des séquences non-vides de symboles notés a et pris dans un ensemble d'étiquettes atomiques $\mathcal{A}$:

Étiquettes simples	$\mathcal{A}^+$:	$l ::= a \mid a.l$
Termes	$\mathcal{T}^g$:	$M ::= x \mid x^l \mid f^l(M_1, M_2, \ldots, M_a)$

On note $\mathcal{A}^*$ l'ensemble $\mathcal{A}^+$ augmenté de l'étiquette vide ϵ. Comme les occurrences, les étiquettes simples sont (partiellement) ordonnées par l'ordre préfixe défini par l'équivalence $l \preceq l' \Leftrightarrow \exists l'' \in \mathcal{A}^*,\ l = l'.l''$.

La fonction τ, définie de l'ensemble des termes étiquetés $\mathcal{T}^g$ dans l'ensemble des étiquettes A^+, est la fonction qui associe son *étiquette de tête* à chaque sous-terme.

$$\tau(x^l) = l \qquad \tau(f^l(M_1, M_2, \ldots, M_a)) = l$$

Les variables peuvent ne pas porter d'étiquette. Ceci n'est possible que dans les termes étiquetés qui n'entendent pas représenter des graphes concrets, mais sont plutôt des intermédiaires techniques, comme, par exemple, les membres gauches et droits des règles de réécriture.

Le graphe de droite de la figure 1.4 est maintenant une représentation parlante du terme étiqueté $((\mathtt{1}^e[(R^f.s^g)^d])^b\ (\mathtt{1}^h[(R^f.s^g)^d])^c)^a$. Les deux sous-termes de la forme $R.s$ portent la même étiquette d, ce qui indique qu'ils sont deux *instances* d'un sous-terme partagé. Par contre, les deux sous-termes dont le squelette est $\mathtt{1}[R.s]$ ne portent pas la même étiquette : ils sont des sous-termes distincts.

Il y a une analogie entre l'étiquette d'un sous-terme et son adresse dans la mise en œuvre dans la mémoire d'un ordinateur du graphe qui est le terme étiqueté M. Ainsi, toujours avec le même terme $((\mathtt{1}^e[(R^f.s^g)^d])^b\ (\mathtt{1}^h[(R^f.s^g)^d])^c)^a$ on a, en notant les adresses comme des entiers $a = \mathit{1}$, $b = \mathit{4}$, $c = \mathit{8}$ ...

@	*4*	*8*	[ ]	*7*	*11*	1	[ ]	*11*	*12*	1	·	*15*	*16*	R	s

Ceci justifie que l'on appelle *adresse* d'un sous-terme son étiquette de tête.

Définition 1.2.1 (Sous-termes partagés) *Soit M un terme étiqueté. Soit une étiquette l. Le sous-terme partagé d'adresse l, noté $\mathcal{S}(l)$, est l'ensemble des occurrences des sous-termes de M qui ont l pour adresse. Un sous-terme M/u, tel que l'occurrence u appartient au sous-terme partagé $\mathcal{S}(l)$, est une instance du sous-terme partagé d'adresse l.*

Tous les termes étiquetés ne correspondent pas forcément à un graphe valide. C'est le cas, par exemple, du terme $(\mathtt{1}^a\ \mathtt{1}^a)^a$. On souhaite également se placer dans le cas où l'on peut attribuer un statut aux sous-termes partagés, c'est à dire éviter qu'un sous-terme partagé ait à la fois des instances autorisées et des instances interdites.

On va donc caractériser un sous-ensemble de $\mathcal{T}^g$ comme regroupant des termes "raisonnables".

Définition 1.2.2 (Graphes acycliques (DAG)) *Soit M un terme étiqueté de $\mathcal{T}^g$. Le terme M est un graphe (DAG) si et seulement si, quels que soient les sous-termes M_1 et M_2 de M, on a l'implication :*

$$\tau(M_1) = \tau(M_2) \ \Rightarrow M_1 = M_2 \textit{ et les deux sous-termes } M_1 \textit{ et } M_2 \textit{ ont même statut.}$$

Dans un DAG, les sous-termes partagés sont conformes à l'intuition. En effet, non seulement toutes leurs instances ont la même structure, mais aussi tous leurs sous-termes sont étiquetés de la même façon. Autrement dit, le partage d'un sous-terme entraîne le partage de tous ses sous-termes. Notons aussi que, dans un DAG et grâce à la condition sur le statut des occurrences, si une des instances d'un sous-terme partagé est un radical, alors toutes les instances de ce sous-terme partagé sont des radicaux. Dans ce cas, on parle de radical partagé.

Définition 1.2.3 (Cycle) *Soit M un terme simplement étiqueté. Le terme M contient un cycle, si et seulement si il existe deux occurrences u et v du terme M, avec :*

$$u \prec v \quad et \quad \tau(M/u) = \tau(M/v)$$

On voit immédiatement, puisque l'on ne considère que des termes finis, qu'un DAG ne peut pas contenir de cycle. Ceci justifie l'emploi du mot anglais DAG, pour *Directed Acyclic Graph.*

1.2.3 Réductions parallèles

Les règles de réécriture vont maintenant porter sur des termes étiquetés. On construit donc un système de règles de réécriture étiquetées à partir de règles non-étiquetées. Pour ce faire, une définition technique préalable est utile :

Définition 1.2.4 (Diffusion d'une étiquette) *Soient $l \in \mathcal{A}^+$ et $M \in \mathcal{T}^g$. La diffusion $<l \circ M>$ de l'étiquette l sur le terme M est un terme simplement étiqueté défini par :*

$$\begin{cases} <l \circ x> = x \\ <l \circ f^m(M_1, M_2, \ldots, M_a)> = f^{lm}(<l \circ M_1>, <l \circ M_2>, \ldots, <l \circ M_a>) \end{cases}$$

La diffusion n'est pas définie sur les variables étiquetées. En effet, cette opération n'est qu'une étape technique de l'étiquetage des règles de réécriture dont les variables ne seront pas étiquetées.

Soient maintenant les règles $\{\ldots, \alpha_i \to \beta_i, \ldots\}$ d'un système non-étiqueté. On construit les règles d'un système étiqueté correspondant en deux étapes.

1. Tout d'abord, chaque membre droit β_i est étiqueté avec des étiquettes *atomiques* et en laissant les variables *sans étiquette*, ce qui nous donne le terme B_i.
2. Ensuite, chaque règle $\alpha_i \to \beta_i$ est remplacée par un ensemble de règles étiquetées, à raison d'une règle par étiquetage possible A_j de α_i, en laissant toujours les variables de α_i sans étiquette dans le terme étiqueté A_j. Cette nouvelle règle est $A_j \to <\tau(A_j) \circ B_i>$.

À titre de premier exemple, soient les règles de réécriture :

$$I(x) \to x \qquad\qquad A \to B \qquad\qquad F(x, B) \to A$$

Une version étiquetée possible de ces règles est :

$$I^l(x) \to x \qquad\qquad A^l \to B^{l.a} \qquad\qquad F^l(x, B^m) \to A^{l.b}$$

Chacune des trois règles étiquetées ci-dessus représente l'ensemble des règles qui sont les étiquetages d'une même règle du système initial. Autrement dit, il faut voir les étiquettes l et m, qui apparaissent dans les membres gauche des règles, comme des variables sur les étiquettes ; tandis que les étiquettes atomiques a et b, qui n'apparaissent que dans les membres droits de règles, sont fixées. Si on poursuit l'analogie entre les termes étiquetés et la mise en œuvre informatique des graphes, on voit que chaque occurrence dure des membres droits étiqueté correspond à un nouveau nœud dans le graphe. Un tel nœud doit être alloué, c'est à dire qu'il faut le créer en lui donnant une adresse qui soit nouvelle. Quand on applique la règle $F^l(x, B^m) \to A^{l.b}$ par exemple, il y a un nouveau nœud à allouer, pour lequel une nouvelle étiquette $l.b$ est explicitement construite.

Si on étiquette les règles de $\lambda\sigma$ faible de sorte que les termes B_i portent des étiquettes toutes distinctes, on obtient les règles :

$$
\begin{array}{lrcl}
\text{(WeakBeta)} & ((((\lambda M)^{l_3})[s])^{l_2}\ N)^{l_1} & \to & (M[(N.s)^{l_1.b}])^{l_1.a} \\
\text{(App)} & ((M\ N)^{l_2}[s])^{l_1} & \to & ((M[s])^{l_1.d}\ (N[s])^{l_1.e})^{l_1.c} \\
\text{(FVar)} & (\mathtt{1}^{l_2}[(M.s)^{l_3}])^{l_1} & \to & M \\
\text{(RVar)} & (\mathtt{n+1}^{l_2}[(M.s)^{l_3}])^{l_1} & \to & (\mathtt{n}^{l_1.g}[s])^{l_1.f}
\end{array}
$$

Certaines règles créent ici plusieurs occurrences, c'est à dire que l'on construit plusieurs nouvelles étiquettes en même temps. Par exemple, dans le cas de la règle (WeakBeta), $l_1.a$ et $l_1.b$ sont les nouvelles adresses des nouvelles occurrences. En imposant certaines contraintes de cohérence aux étiquettes du terme départ de la réduction ainsi qu'à celles des membres droits de règles, nous verrons que cette construction explicite des adresses des nœuds alloués ne provoque aucun "accident d'allocation" . C'est à dire que les nouvelles étiquettes ne sont pas des adresses déjà utilisées dans le graphe et que le fait d'ajouter de nouvelle occurrences à un terme étiqueté ne modifie pas sa cohérence en tant que représentation sous forme de terme d'un graphe acyclique.

La fonction de domaine ne subit aucune modification quand on étiquette un système. En revanche, la relation de réécriture simple n'est pas adaptée à la modélisation de la réécriture de graphes. En effet, on souhaite naturellement réduire toutes les instances d'un radical partagé en une seule étape. Du point de vue des termes, on va donc contracter plusieurs radicaux simultanément. On définit la relation de réduction parallèle par les axiomes et règles d'inférence suivants :

$$
x^l \to x^l \ \text{(id)} \qquad \frac{M_i \to N_i \text{ si } i \in \mathrm{dom}(f) \qquad M_i = N_i \text{ si } i \notin \mathrm{dom}(f)}{f^l(M_1, M_2, \ldots, M_a) \to f^l(N_1, N_2, \ldots, N_a)}\ \text{(rec)}
$$

$$
\frac{\sigma(x) \to \rho(x) \text{ si } x \in \mathcal{V}_a(\alpha) \qquad \sigma(x) = \rho(x) \text{ si } x \in \mathcal{V}_i(\alpha)}{\sigma(\alpha) \to \rho(\beta)}\ \text{(red)}
$$

L'axiome (id) de la figure ci-dessus est en fait un schéma d'axiomes et il existe un axiome du type (id) par variable et par étiquette possible. De même, les règles d'inférence (red) et (rec), s'entendent compte-tenu du choix d'une règle de réécriture ou d'un symbole de tête étiqueté.

Dans le cas du système de trois règles étiqueté en premier, on obtient l'axiome et les règles d'inférence suivants :

$$
x^l \to x^l \ \text{(id)}
$$

$$
\frac{M_1 \to N_1 \quad M_2 \to N_2}{F^l(M_1, M_2) \to F^l(N_1, N_2)}\ \text{(rec)} \qquad \frac{M_1 \to N_1}{I^l(M_1) \to I^l(N_1)}\ \text{(rec)} \qquad \frac{}{A^l \to A^l}\ \text{(rec)} \qquad \frac{}{B^l \to B^l}\ \text{(rec)}
$$

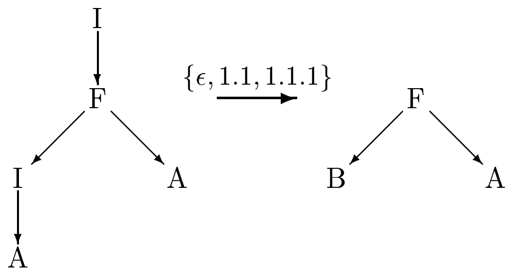
$$
\frac{M_1 \to N_1}{I^l(M_1) \to N_1}\ \text{(red)} \qquad \frac{}{A^l \to B^{l.a}}\ \text{(red)} \qquad \frac{M_1 \to N_1}{F^l(M_1, B^m) \to A^{l.b}}\ \text{(red)}
$$

On utilise ensuite ces règles d'inférence pour donner une représentation rigoureuse d'une réduction représentée informellement par un dessin (figure 1.5).

Dans le cadre de $\lambda\sigma$ faible, soit ensuite le terme :

$$
M = (\mathtt{1}^c[(\mathtt{1}^f[(\mathtt{1}^h.t^i)^g]^e.s^j)^d]^b\ \mathtt{1}^l[(\mathtt{1}^f[(\mathtt{1}^h.t^i)^g]^e.s^j)^d]^k)^a
$$

Le terme étiqueté M est un DAG (cf. figure 1.6) et il contient un radical partagé de type (FVar) et d'adresse c. La figure 1.7 définit la relation de réduction parallèle, tandis que la figure 1.8 donne un exemple de réduction parallèle qui contracte le radical partagé d'adresse c.



$$\dfrac{\dfrac{\dfrac{}{A^d \to B^{d.b}}\,(\mathrm{red})}{I^c(x)[x\backslash A^d] \to x[x\backslash B^{d.b}]}\,(\mathrm{red}) \qquad \dfrac{}{A^e \to A^e}\,(\mathrm{rec})}{\dfrac{F^b(I^c(A^d), A^e) \to F^b(B^{d.b}, A^e)}{I^a(x)[x\backslash F^b(I^c(A^d), A^e)] \to x[x\backslash F^b(B^{d.b}, A^e)]}\,(\mathrm{red})}\,(\mathrm{rec})$$

Figure 1.5 : Exemple de réduction parallèle

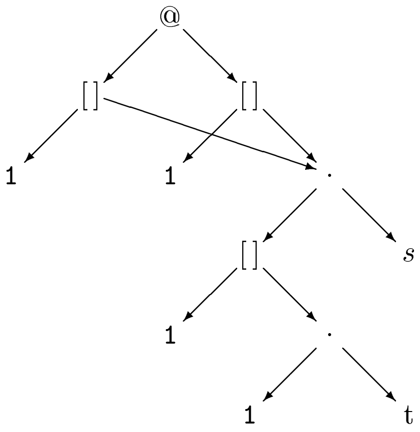


Figure 1.6 : Un terme de $\lambda\sigma$ faible étiqueté

Au cours d'une réduction simple, un radical est contracté. Au cours d'une réduction parallèle, ce sont plusieurs radicaux qui sont contractés.

Définition 1.2.5 (Radicaux contractés) *Soit une réduction $A\colon M \to N$. L'ensemble $\mathcal{R}(A)$ regroupe les (occurrences des) radicaux contractés par A et l'on note $A\colon M \stackrel{\mathcal{R}(A)}{\longrightarrow} N$. L'ensemble $\mathcal{R}(A)$ est défini comme suit :*

1. *Si A est du type (id), alors $\mathcal{R}(A)$ est l'ensemble vide.*
2. *Si A est du type (rec), alors on pose :*

$$\frac{A_i\colon M_i \stackrel{\mathcal{R}(A_i)}{\longrightarrow} N_i \ \textit{si}\ i \in \textit{dom}(f) \qquad A_i\colon M_i = N_i \ \textit{si}\ i \notin \textit{dom}(f)}{A\colon f^l(M_1, M_2, \ldots, M_a) \xrightarrow{(\bigcup_{i \in dom(f)} i.\mathcal{R}(A_i))} f^l(N_1, N_2, \ldots, N_a)}\,(\mathrm{rec})$$

3. *Si A est du type (red), alors le terme M est de la forme $\sigma(\alpha)$. Pour toute variable x du nom α, on note u_x l'occurrence de x dans α. On pose ensuite :*

$$\frac{A_x\colon \sigma(x) \stackrel{\mathcal{R}(A_x)}{\longrightarrow} \rho(x) \ \textit{si}\ x \in \mathcal{V}_a(\alpha) \qquad A_x\colon \sigma(x) = \rho(x) \ \textit{si}\ x \in \mathcal{V}_i(\alpha)}{A\colon \sigma(\alpha) \xrightarrow{(\bigcup_{x \in \mathcal{V}_a(\alpha)} u_x.\mathcal{R}(A_x)) \ \cup \ \{\epsilon\}} \rho(\beta)}\,(\mathrm{red})$$

$$X^l \to X^l \text{ (idTerms)} \qquad x^l \to x^l \text{ (idSubst)}$$

$$\frac{}{\mathtt{n}^l \to \mathtt{n}^l}\text{ (recVar)} \qquad \frac{M = M}{(\lambda M)^l \to (\lambda M)^l}\text{ (recLambda)}$$

$$\frac{M_1 \to N_1 \quad M_2 \to N_2}{(M_1\ M_2)^l \to (N_1\ N_2)^l}\text{ (recApp)} \qquad \frac{M_1 \to N_1 \quad s_2 \to t_2}{(M_1.s_2)^l \to (N_1.t_2)^l}\text{ (recCons)}$$

$$\frac{M_1 = M_2 \quad s_2 \to t_2}{(M_1[s_2])^l \to (M_2[t_2])^l}\text{ (recSubst)}$$

$$\frac{\sigma(N) \to \rho(N) \quad \sigma(s) \to \rho(s) \quad \sigma(M) = \rho(M)}{\sigma(((((\lambda M)^{l_3})[s])^{l_2}\ N)^{l_1}) \to \rho((M[(N.s)^{l_1.b}])^{l_1.a})}\text{ (redBeta)}$$

$$\frac{\sigma(s) \to \rho(s) \quad \sigma(M) = \rho(M) \quad \sigma(N) = \rho(N)}{\sigma(((M\ N)^{l_2}[s])^{l_1}) \to \rho(((M[s])^{l_1.d}\ (N[s])^{l_1.e})^{l_1.c})}\text{ (redApp)}$$

$$\frac{\sigma(M) \to \rho(M) \quad \sigma(s) \to \rho(s)}{\sigma((\mathtt{1}^{l_2}[(M.s)^{l_3}])^{l_1}) \to \rho(M)}\text{ (redFVar)} \qquad \frac{\sigma(M) \to \rho(M) \quad \sigma(s) \to \rho(s)}{\sigma((\mathtt{n+1}^{l_2}[(M.s)^{l_3}])^{l_1}) \to \rho((\mathtt{n}^{l_1.g}[s])^{l_1.f})}\text{ (redRVar)}$$

Figure 1.7 : Réduction parallèle dans $\lambda\sigma$ faible étiqueté

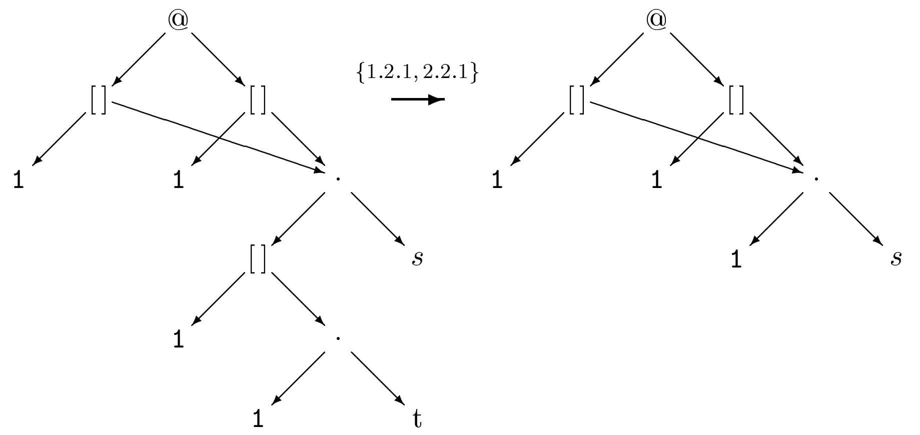
Par exemple, l'ensemble des occurrences des radicaux contractés est égal à $\{\epsilon, 1.1, 1.1.1\}$ dans la réduction de la la figure 1.5 et à $\{1.2.1, 2.2.1\}$, dans la réduction de la figure 1.8. Une réduction qui ne contracte aucun radical est *vide.*

Jusqu'ici, pour caractériser une réduction parallèle, il n'y a pas d'autre moyen que de donner tout son arbre de preuve. Heureusement, on peut reconstituer cet arbre de preuve à partir de données nettement moins encombrantes.

Lemme 1.2.6 *La donnée d'un terme M et d'un ensemble U d'occurrences de radicaux à contracter dans M est suffisante pour caractériser une et une seule réduction A: $M \overset{U}{\to} N$.*

Démonstration: On va tout simplement reconstituer la réduction A: $M \overset{U}{\to} N$ (c'est à dire son arbre de preuve), à partir du couple (M, U). On distingue trois cas :

1. Si le terme M est une variable étiquetée, alors l'ensemble U est nécessairement vide et la réduction A: $x^l \to x^l$ est prouvée par un axiome (id).
2. Si le terme M est de la forme $M = f^l(M_1, \dots, M_a)$ et que l'occurrence à la racine ϵ n'est pas élément de U, alors on peut écrire l'ensemble U sous la forme $U = \bigcup_{i \in \mathrm{dom}(f)} i.U_i$, car les occurrences de radicaux sont toutes autorisées. Pour tout indice i autorisé à partir du symbole f, on construit ensuite inductivement l'arbre de preuve de la réduction A_i: $M_i \overset{U_i}{\to} N_i$. Pour tout indice interdit à partir du symbole f, on ne peut que poser A_i: $M_i = N_i$. Enfin, on termine la construction de l'arbre de preuve de la réduction A en utilisant une règle (rec),



$$
\dfrac{
\mathtt{1}^c = \mathtt{1}^c
\qquad
\dfrac{
\dfrac{
\dfrac{}{\mathtt{1}^h \to \mathtt{1}^h}\,\text{(rec)}
\qquad
t^i \to t^i\ \text{(id)}
}{
\mathtt{1}^f[(\mathtt{1}^h.t^i)^g]^e \to \mathtt{1}^h
}\,\text{(red)}
\qquad
s^j \to s^j\ \text{(id)}
}{
(\mathtt{1}^f[(\mathtt{1}^h.t^i)^g]^e.s^j)^d \to (\mathtt{1}^h.s^j)^d
}\,\text{(rec)}
}{
\mathtt{1}^c[(\mathtt{1}^f[(\mathtt{1}^h.t^i)^g]^e.s^j)^d]^b \to \mathtt{1}^c[(\mathtt{1}^h.s^j)^d]^b
}\,\text{(rec)}
\qquad
\dfrac{
\mathtt{1}^l = \mathtt{1}^l
\quad
\begin{array}{c}
\dfrac{}{\mathtt{1}^h \to \mathtt{1}^h}\,\text{(rec)} \qquad t^i \to t^i\ \text{(id)} \\
\vdots \\
(\mathtt{1}^f[(\mathtt{1}^h.t^i)^g]^e.s^j)^d \to (\mathtt{1}^h.s^j)^d
\end{array}
}{
\mathtt{1}^l[(\mathtt{1}^f[(\mathtt{1}^h.t^i)^g]^e.s^j)^d]^k \to \mathtt{1}^l[(\mathtt{1}^h.s^j)^d]^k
}\,\text{(rec)}
$$

$$
\overline{(\mathtt{1}^c[(\mathtt{1}^f[(\mathtt{1}^h.t^i)^g]^e.s^j)^d]^b\ \mathtt{1}^l[(\mathtt{1}^f[(\mathtt{1}^h.t^i)^g]^e.s^j)^d]^k)^a \to (\mathtt{1}^c[(\mathtt{1}^h.s^j)^d]^b\ \mathtt{1}^l[(\mathtt{1}^h.s^j)^d]^k)^a}\ \text{(rec)}
$$

Figure 1.8 : Une réduction parallèle dans $\lambda\sigma$ faible

qui est la seule possible ici :

$$\frac{A_i\colon M_i \to N_i \text{ si } i \in \mathrm{dom}(f) \qquad A_i\colon M_i = N_i \text{ si } i \notin \mathrm{dom}(f)}{A\colon f^l(M_1, M_2, \ldots, M_a) \to f^l(N_1, N_2, \ldots, N_a)}\ (\mathrm{rec})$$

3. Si le terme M est de la forme $M = f^l(M_1, \ldots, M_a)$ et que l'occurrence à la racine ϵ est un élément de U, alors le terme M admet un radical α à la racine, c'est à dire qu'il existe un membre gauche de règle α et une substitution σ, avec $M = \sigma(\alpha)$. D'après la condition de non-recouvrement des noms de radicaux, l'occurrence ϵ est suffisante pour retrouver d'abord le membre gauche α dans le terme M, puis le membre droit β de la règle de réécriture à utiliser ici. Une autre conséquence de cette condition d'orthogonalité est que l'on peut écrire l'ensemble U sous la forme $U = \{\epsilon\} \cup (\bigcup_{x \in \mathcal{V}_a(\alpha) u_x . U_x})$, où u_x est l'occurrence de la variable x dans le nom α et U_x est un ensemble d'occurrences de radicaux du sous-terme $M_x = \sigma(x)$. Pour chaque variable x autorisée dans α, on construit ensuite inductivement l'arbre de preuve de la réduction $A_x\colon M_x \xrightarrow{U_x} N_x$. Par ailleurs, pour chaque variable x interdite dans le nom α, on pose $A_x\colon M_x = N_x$. On définit ensuite une substitution ρ, par ses valeurs sur les variables de α : $\forall x \in \mathcal{V}(\alpha),\ \rho(x) = N_x$. D'après la condition de base $\mathcal{V}(\beta) \subseteq \mathcal{V}(\alpha)$, on peut considérer le terme $\rho(\beta)$. Il reste, pour compléter l'arbre de preuve de le réduction A, à composer les arbres de preuve des réductions A_x à l'aide d'une règle (red) qui est la seule possible ici :

$$\frac{A_x\colon \sigma(x) \to \rho(x) \text{ si } x \in \mathcal{V}_a(\alpha) \quad A_x\colon \sigma(x) = \rho(x) \text{ si } x \in \mathcal{V}_i(\alpha)}{A\colon \sigma(\alpha) \to \rho(\beta)}\ (\mathrm{red})$$

□

Comme cas particulier de ce lemme, il existe une et une seule réduction vide issue d'un terme M donné. On note cette réduction $\emptyset\colon M \to M$, ou même $\emptyset$. L'arbre de preuve de la réduction vide n'a pas de nœud du type (red) et sa structure suit exactement celle du terme M.

Dans le cas où l'ensemble U ne comprend qu'un seul élément u, il est clair que l'on peut identifier la réduction parallèle qui contracte l'ensemble de radicaux $\{u\}$ et la réduction simple qui contracte le radical d'occurrence u. Par conséquent, on ne considère plus que des réductions parallèles que l'on appelle tout simplement des réductions.

La relation de dérivation sur l'ensemble des termes est la fermeture réflexive et transitive de la relation de réduction. Cette notion et les notations qui l'accompagnent restent valables dans le cas des réductions parallèles. À proprement parler, une dérivation A de longueur n est caractérisée par un terme initial M_0 et une suite $(U_i)_{1 \leq i \leq n}$ d'ensembles d'occurrences de radicaux à contracter successivement. Quand on veut insister sur cette présentation de la dérivation A, on l'écrit sous forme d'étapes élémentaires :

$$A\colon M_0 \overset{U_1}{\to} M_1 \overset{U_2}{\to} \ldots \overset{U_n}{\to} M_n$$

Remarquons que, pour tout entier i de l'intervalle $[0 \ldots n[$, le terme M_{i+1} peut être calculé à l'aide du terme M_i et des radicaux de U_{i+1}. Par conséquent, les termes $(M_i)_{1 \leq i \leq n}$ ne figurent dans l'écriture de la dérivation A ci-dessus que pour des raisons de lisibilité.

Notation : Soit une dérivation $A\colon M_0 \overset{U_1}{\to} M_1 \overset{U_2}{\to} \ldots \overset{U_n}{\to} M_n$. Pour tout entier i compris entre 1 et n, la i-ème étape élémentaire de la dérivation A est la réduction :

$$A_i\colon M_{i-1} \overset{U_i}{\to} M_i$$

C'est à dire que l'on a $A = A_1 A_2 \dots A_n$. Soient maintenant deux entiers i et j, compris entre 1 et n et tels que l'on a $i \leq j$, la dérivation intervalle prise entre la i-ème et la j-ème étape de A est la dérivation $A[i,j] = A_i A_{i+1} \dots A_j$.

1.2.4 Résidus dans les réductions parallèles

La définition des résidus par une réductions parallèle est une extension de la définition dans le cas de la réécriture simple (définition 1.1.7). Cette extension n'est pas tout à fait immédiate.

Définition 1.2.7 (Résidus) *Soit une réduction $A\colon M \to N$. Pour toute occurrence u prise dans le terme initial M, on définit l'ensemble $u \setminus A$ des résidus de l'occurrence u par la réduction A dans le terme N en fonction de la nature de la réduction A :*

1. *Si A est du type (id), alors on a $M = N = x^l$ et on pose $\epsilon \setminus A = \{\epsilon\}$.*
2. *Si A est du type (rec), avec*

$$\frac{A_i\colon M_i \to N_i \ \mathit{si}\ i \in \mathit{dom}(f) \qquad A_i\colon M_i = N_i \ \mathit{si}\ i \notin \mathit{dom}(f)}{A\colon f^l(M_1, M_2, \dots, M_a) \to f^l(N_1, N_2, \dots, N_a)}\ (\mathrm{rec})$$

Alors, on pose :

$$\begin{cases} \epsilon \setminus A = \{\epsilon\} \\ \forall i \in [1 \dots a],\ \forall u \in \mathcal{O}(M_i),\ (i.u)\setminus A = i.(u \setminus A_i) \end{cases}$$

(Par convention, pour toute égalité $B\colon P = Q$ et toute occurrence u dans $\mathcal{O}(P)$, on a posé $u \setminus B = \{u\}$.)

3. *Enfin, si A est du type (red), alors on pose :*

$$\frac{A_x\colon \sigma(x) \to \rho(x) \ \mathit{si}\ x \in \mathcal{V}_a(\alpha) \qquad A_x\colon \sigma(x) = \rho(x) \ \mathit{si}\ x \in \mathcal{V}_i(\alpha)}{A\colon \sigma(\alpha) \to \rho(\beta)}\ (\mathrm{red})$$

Par définition, les occurrences de $M = \sigma(\alpha)$ qui sont aussi des occurrences intérieures du nom α sont contractées par la réduction A et n'ont pas de résidu :

$$\forall u \in \overline{\mathcal{O}}(\alpha),\ u \setminus A = \emptyset$$

En revanche, les autres occurrences du terme M peuvent avoir des résidus. Soit donc une variable x d'occurrence u_x dans le nom α. On pose :

$$\forall u \in \mathcal{O}(\sigma(x)),\ (u_x.u) \setminus A = \{\, v_x \in \mathcal{O}(\beta) \mid \beta/v_x = x \,\}.(u \setminus A_x)$$

Dans le cas où U est un singleton, cette définition est équivalente à la définition 1.1.7 des résidus par une réduction simple. Les notions d'occurrences contractées, effacées, dupliquées et créées s'étendent naturellement au cas des réductions parallèles.

Si l'on revient sur la réduction de la figure 1.8, on a par exemple : $1.2.1.2/A = \emptyset$ (occurrence contractée), $1.2.1.2/A = \emptyset$ (occurrence effacée) et $1.2.1.1/A = \{1.2.1\}$.

Certaines occurrences ne sont pas concernées par les réductions.

Définition 1.2.8 (Occurrences préservées) *Soit une réduction $A\colon M \xrightarrow{U} N$. Et soit u une occurrence du terme M. Par définition, l'occurrence u est préservée par la réduction A, si et seulement si on a :*

$$\forall v \in U,\ u \prec v \ \mathit{ou}\ u \mid v$$

Nous avons déjà rencontré les occurrences préservées dans le premier cas de la définition 1.1.7 des résidus, dans ce cas les occurrences initiales et les résidus étaient formellement les mêmes. Cette identité vaut aussi pour les symboles de tête.

Définition 1.2.9 *Deux termes M et N ont la même tête, si et seulement si on se trouve dans l'un des des cas suivants :*

$$\left\{\begin{array}{l} M = f^l(M_1, M_2, \ldots, M_a) \\ \quad et \\ N = f^l(N_1, N_2, \ldots, N_a) \end{array}\right. \qquad ou \qquad \left\{\begin{array}{l} M = x^l \\ \quad et \\ N = x^l \end{array}\right.$$

Lemme 1.2.10 *Soit une réduction $A\colon M \xrightarrow{U} N$. Soit une occurrence u préservée par la réduction A. Alors, on a l'égalité $u \setminus A = \{u\}$. En outre, les sous-termes M/u et N/u ont la même tête.*

Démonstration : Par induction sur la structure de A. Cette induction ne porte que sur des réductions de type (id) et de type (rec), car il n'y a pas d'occurrence préservée par les réductions du type (red). □

En particulier, si la réduction $A\colon M \xrightarrow{U} N$ est vide ($U = \emptyset$ ou $A = \emptyset$), alors toutes les occurrences de M sont préservées par A.

1.2.5 Descendants, héritiers

Les occurrences des radicaux contractés par la réduction A n'ont pas de résidu :

Lemme 1.2.11 *Soit une réduction $A\colon M \xrightarrow{U} N$. On a l'égalité : $U \setminus A = \emptyset$*

Démonstration : Facile, à partir du cas des réductions du type (red) :

$$\frac{A_x\colon \sigma(x) \to \rho(x) \text{ si } x \in \mathcal{V}_a(\alpha) \quad A_x\colon \sigma(x) = \rho(x) \text{ si } x \in \mathcal{V}_i(\alpha)}{A\colon \sigma(\alpha) \to \rho(\beta)} \text{ (red)}$$

L'occurrence à la racine ϵ est l'occurrence du radical à la racine contracté a ce niveau et n'a pas de résidu par A. □

De façon plus générale, ce sont toutes les occurrences dures du nom α d'un radical contracté $\sigma(\alpha)$ qui n'ont pas de résidu, on peut néanmoins leur associer la ou les occurrences du contractum $\rho(\beta)$ correspondant.

Définition 1.2.12 (Descendants) *Pour toute réduction $A\colon M \to N$ et toute occurrence u dans le terme M, on définit l'ensemble $u\setminus\setminus A$ des descendants de u par A en reprenant complètement la définition des résidus et en ajoutant le seul cas de base suivant :*

3. Si la réduction A est du type (red), avec

$$\frac{A_x\colon \sigma(x) \to \rho(x) \;\; si \; x \in \mathcal{V}_a(\alpha) \qquad A_x\colon \sigma(x) = \rho(x) \;\; si \; x \in \mathcal{V}_i(\alpha)}{A\colon \sigma(\alpha) \to \rho(\beta)} \text{ (red)}$$

Alors, pour toute occurrence u dure dans α, on pose $u\setminus\setminus A = \{\epsilon\}$.

Enfin, il importe parfois de distinguer parmi les occurrences qui sont des descendants celles qui ne sont pas des résidus.

Définition 1.2.13 (Héritiers) *Soit une réduction* $A\colon M \to N$, *Pour toute occurrence* u *dans* M, *on définit l'ensemble* $u\backslash\!\!\backslash\!\!\backslash A$ *des héritiers de* u *par* A *de la manière suivante :*

$$u\backslash\!\!\backslash\!\!\backslash A = u\backslash\!\!\backslash A - u \setminus A$$

La définition des descendants est directement inspirée de celle de Z. Khasidashvili [Kha 90]. L'intérêt des descendants est de permettre le suivi des occurrences de radicaux au delà de leur contraction. Du point de vue des graphes, les descendants n'ont pas réellement de signification intuitive et je pense qu'il vaut mieux les voir uniquement comme des outils techniques commodes.

Les occurrences les plus extérieures parmi les occurrences des radicaux contractés marquent la limite entre les occurrences préservées et celles qui ne le sont pas.

Définition 1.2.14 *Soit une réduction* $A\colon M \xrightarrow{U} N$. *Par définition, les radicaux extérieurs pour la réduction* A *sont ceux dont les occurrences sont les points minimaux de l'ensemble* U *pour l'ordre préfixe sur les occurrences. Soit, si on note* $\mathcal{E}xt(A)$ *l'ensemble des occurrences des radicaux extérieurs pour la réduction* A, *on pose :*

$$\mathcal{E}xt(A) = inf_{\preceq}(U)$$

Les occurrences des radicaux extérieurs sont leurs propres héritiers et n'ont pas de résidus.

Lemme 1.2.15 *Soit une réduction* $A\colon M \xrightarrow{U} N$. *Soit* u *l'occurrence d'un radical extérieur pour* A. *Alors, on a* $u\backslash\!\!\backslash A = u\backslash\!\!\backslash\!\!\backslash A = \{u\}$.

Démonstration : La preuve est facile, par induction sur la structure de A et à partir du cas où la réduction A est du type (red). Dans ce dernier cas, on a $\mathcal{E}xt(A) = \{\epsilon\}$, $\epsilon \setminus A = \emptyset$ et $\epsilon\backslash\!\!\backslash\!\!\backslash A = \{\epsilon\}$. □

1.2.6 Statut des résidus

La troisième condition d'orthogonalité a une importante conséquence sur le statut des résidus.

Lemme 1.2.16 *Soit* $A\colon M \to N$ *une réduction. Soient* u *une occurrence du terme* M *et* v *une occurrence résidu de* u *par* A *dans le terme* N. *Alors, les occurrences* u *et* v *ont le même statut.*

Démonstration: La preuve est par induction sur la définition des résidus dont elle reprend les notations :

1. Si A est du type (id), alors l'occurrence ϵ est autorisée dans M comme dans N.
2. Si A est du type (rec), on distingue trois sous-cas :
   (a) L'occurrence à la racine ϵ est autorisé dans M comme son unique résidu ϵ l'est dans N.
   (b) Soit i un indice interdit à partir du symbole de tête f de M. Alors, toute occurrence de $\mathcal{O}(M)$ qui est de la forme $i.u_i$ est interdite. De même, l'unique résidu $i.u_i$ de l'occurrence $i.u_i$ est interdit dans le terme N.
   (c) Soit i un indice du domaine de f. On considère les occurrences de la forme $u = i.u_i$. Les éventuels résidus de u sont de la forme $v = i.v_i$, avec $v_i \in u_i \setminus A_i$. Comme l'indice i est autorisé, les occurrences u et v ont même statut dans les terme M et N que les occurrences u_i et v_i dans les termes M_i et N_i respectivement. Il reste à appliquer l'hypothèse d'induction à la réduction $A_i\colon M_i \to N_i$ pour conclure.

3. Si A est du type (red), alors, par définition des résidus, l'occurrence u est de la forme $u_x.u'$, où u_x est l'occurrence d'une variable x dans α et l'occurrence u' est une occurrence du sous-terme $\sigma(x)$; tandis que l'occurrence v est de la forme $v_x.v'$, où l'occurrence v_x est une occurrence de la variable x dans β et $v' \in u' \setminus A_x$. On distingue ensuite deux cas :
   (a) Si la variable x est interdite dans α, alors l'occurrence u est interdite. Par la condition d'orthogonalité $\mathcal{V}_a(\beta) \subseteq \mathcal{V}_a(\alpha)$, l'occurrence v_x ne peut pas être autorisée dans β. Par suite, l'occurrence v est interdite.
   (b) Si la variable x est autorisée dans α, alors, l'occurrence $u = u_x.u'$ a même statut que l'occurrence u'. Par induction, l'occurrence v' a donc le même statut que l'occurrence u'. En outre, par la condition d'orthogonalité $\mathcal{V}_i(\beta) \subseteq \mathcal{V}_i(\alpha)$, l'occurrence v_x ne peut pas être interdite dans β. Et donc l'occurrence v a même statut que l'occurrence v'. Finalement, les occurrences u et v ont même statut. □

Si on examine attentivement les cas 3-(b) et 3-(c) de la preuve ci-dessus, on constate que l'hypothèse d'orthogonalité $\mathcal{V}_i(\beta) \subseteq \mathcal{V}_i(\alpha)$ permet de prouver que “les résidus des occurrences autorisées sont autorisés” ; tandis que l'autre condition $\mathcal{V}_a(\beta) \subseteq \mathcal{V}_a(\alpha)$ est suffisante pour prouver que “les résidus des occurrences interdites sont interdits”. De même, la condition $\mathcal{V}_i(\beta) \subseteq \mathcal{V}_i(\alpha)$ suffit pour prouver que les héritiers sont toujours des occurrences autorisées.

1.2.7 Résidus par les dérivations

Définition 1.2.17 (Résidus d'un ensemble d'occurrences) *Soit une réduction A: $M \to N$. Soit U un ensemble d'occurrences dans M. On définit les ensemble $U \setminus A$, $U \backslash\!\backslash A$ et $U \backslash\!\backslash\!\backslash A$ des résidus, descendants et héritiers de U par A :*

$$\begin{array}{rcl} U \setminus A & = & \{ v \in \mathcal{O}(N) \mid \exists u \in U,\ v \in u \setminus A \} \\ U \backslash\!\backslash A & = & \{ v \in \mathcal{O}(N) \mid \exists u \in U,\ v \in u \backslash\!\backslash A \} \\ U \backslash\!\backslash\!\backslash A & = & U \backslash\!\backslash A - U \setminus A \end{array}$$

Les résidus s'étendent aux dérivations par transitivité.

Définition 1.2.18 (Résidus par dérivation) *Soit une dérivation A: $M_0 \xrightarrow{U_1} M_1 \xrightarrow{U_2} \dots \xrightarrow{U_n} M_n$. Soit V un ensemble d'occurrences prises dans le terme M. On définit l'ensemble $V \setminus A$ comme le dernier terme de la suite $(V_i)_{0 \leq i \leq n}$, elle même définie comme suit :*

$$V_0 = V, \quad \forall i \in [0 \dots n-1],\ V_{i+1} = V_i \setminus A_i, \quad V \setminus A = V_n$$

De même, on définit l'ensemble des descendants de U par la dérivation A :

$$W_0 = V, \quad \forall i \in [0 \dots n-1],\ W_{i+1} = W_i \backslash\!\backslash A_i, \quad V \backslash\!\backslash A = W_n$$

Enfin, on définit également l'ensemble des héritiers de U par la dérivation A.

$$U \backslash\!\backslash\!\backslash A = U \backslash\!\backslash A - U \setminus A$$

Il est immédiat que pour toute concaténation de dérivations AB, on a les égalités :

$$U \setminus AB = (U \setminus A) \setminus B \quad U \backslash\!\backslash AB = (U \backslash\!\backslash A) \backslash\!\backslash B$$

En outre, on peut donner une caractérisation inductive des héritiers. Pour toute dérivation composite AB avec $A\colon M \to^* N$ et $B\colon N \to^* P$, ainsi que pour toutes occurrence u dans terme M et v dans le terme P, on a l'équivalence :

$$v \in u\backslash\!\!\backslash\!\!\backslash AB \Leftrightarrow \exists w \in \mathcal{O}(N), \begin{cases} w \in u\backslash\!\!\backslash A \text{ et } v \in w\backslash\!\!\backslash\!\!\backslash B \\ \quad \text{ou} \\ w \in u\backslash\!\!\backslash\!\!\backslash A \text{ et } v \in w\backslash\!\!\backslash B \end{cases}$$

1.3 Réécriture de graphes

Dans cette section on examine comment agit la réécriture parallèle sur les termes étiquetés qui sont effectivement des graphes.

1.3.1 Résidus des termes partagés

Les adresses des sous-termes se conservent par résidu.

Proposition 1.3.1 *Pour toute dérivation $A\colon M \to^* N$, on a les propriétés suivantes :*

1. $\forall u \in \mathcal{O}(M),\ \forall v \in \mathcal{O}(N),\ v \in u \setminus A \Rightarrow \tau(N/v) = \tau(M/u)$
2. $\forall v \in \mathcal{O}(N), \exists u \in \mathcal{O}(M),\ \tau(M/u) \preceq \tau(N/v)$

Démonstration : Dans le cas où A est une réduction, on démontre facilement les propositions *1* et *2* par induction sur la structure de A. On termine ensuite la preuve par induction sur la longueur de la dérivation A. □

La réciproque de la proposition (1) ci-dessus n'est pas toujours vraie, elle demande une condition de cohérence sur le terme initial M.

Définition 1.3.2 (Terme cohérent) *Le terme simplement étiqueté M est cohérent, si et seulement si on a la propriété suivante :*

$$\forall u \in \mathcal{O}(M)\ \forall v \in \mathcal{O}(M), \tau(M/u) \preceq \tau(N/u) \Rightarrow \tau(M/u) = \tau(N/u)$$

Autrement dit, dans un terme cohérent deux adresses distinctes ne sont pas préfixes l'une de l'autre. Quand le terme M est à la fois un terme cohérent et un graphe acyclique, les deux conditions des définition 1.2.2 et 1.3.2 se combinent en une seule.

Remarque 1.3.1 *Un terme simplement étiqueté M est un DAG cohérent, si et seulement si, pour tout couple (u_1, u_2) d'occurrences du terme M, on a l'implication*

$$\tau(M/u_1) \preceq \tau(M/u_2) \Rightarrow \begin{cases} M/u_1 = M/u_2 \\ \quad \textit{et} \\ \textit{les occurrences } u_1 \textit{ et } u_2 \textit{ ont même statut} \end{cases}$$

Proposition 1.3.3 *Soit une dérivation $A\colon M \to^* N$ telle que le terme M est cohérent. Soit une occurrence u dans le terme N. Si le sous-terme N/u porte en tête une étiquette qui est aussi l'étiquette de tête d'un sous-terme de M, alors u est le résidu d'une occurrence v de M avec :*

$$\tau(M/v) = \tau(N/u)$$

Démonstration: La preuve est par induction sur la longueur de A. Si A est la dérivation vide, alors $M = N$ et toute occurrence est son propre résidu par A. Sinon, la dérivation A est de la forme $A = BC$, ou B: $M \to^* P$ est une dérivation et C: $P \to N$ est une réduction. Selon la nature de l'occurrence u, on distingue deux cas.

- Si u est résidu par C d'une occurrence w de P, alors on a $\tau(N/u) = \tau(P/w)$ (proposition 1.3.1-(1)). Et donc, par induction, il existe une occurrence v dans M, telle que $w \in v \setminus B$. Soit finalement, $u \in v \setminus A$.
- Sinon, u n'est le résidu d'aucune occurrence de P (et donc d'aucune occurrence de M). C'est à dire que u est une occurrence créée dans N par la réduction C. Par conséquent, il existe un radical d'occurrence w, contracté par C, tel que $\tau(N/u) = \tau(P/w).a$, où a est une étiquette alphabétique. Par la proposition 1.3.1-2, il existe une occurrence v dans M, avec $\tau(M/v) \preceq \tau(P/w)$. Et donc, on a $\tau(M/v) \prec \tau(N/u)$. Soit, par cohérence de M, l'étiquette $\tau(N/u)$ ne peut pas être l'étiquette de tête d'un sous-terme de M. □

Corollaire 1.3.4 *Soit une dérivation A: $M \to^* N$ telle que M est cohérent. Quelle que soit le sous-terme partagé $\mathcal{S}(l)$ de M, on a l'égalité :*

$$\mathcal{S}(l) \setminus A = \{u \in \mathcal{O}(N) \mid \tau(N/v) = l\}.$$

Ce dernier résultat fait clairement apparaître la relation qui existe entre les étiquettes simples et les occurrences.

1.3.2 Cohérence de la réécriture de graphes acycliques

La représentation des graphes comme des termes étiquetés est suffisamment claire pour que je ne parle jamais de graphes en tant qu'objets mathématiques. Cette observation n'est pas gratuite, elle s'appuie notamment sur l'assimilation des étiquettes à des adresses machine dans la mise en œuvre informatique des graphes. Je me contenterai donc de l'exposition précise des contraintes à appliquer à la réécriture étiqueté pour en faire un modèle plausible de la réécriture de graphes (acycliques).

Définition 1.3.5 (Réécriture de graphes) *Un système simplement étiqueté Σ^g est un système de réécriture de graphes, si et seulement si il obéit aux deux conditions suivantes :*

1. *Les membres droits de règles sont des graphes acycliques.*
2. *Les seules réductions autorisées sont les réductions A: $M \xrightarrow{U} N$, telles que M est un graphe acyclique et U est une réunion de radicaux partagés.*

La condition *1* ci-dessus est facile à vérifier : il suffit de la contrôler lors de la première étape de l'étiquetages des membres droits, quand toutes les étiquettes qui décorent les membres droits sont atomiques. Remarquons aussi que les membres droits de règles sont toujours cohérents puisque toutes leurs étiquettes ont même longueur.

Il reste à prouver que l'on peut librement concaténer les réductions de graphes pour en faire des *dérivations de graphes.*

Lemme 1.3.6 *Soit une réduction de graphes A: $M \to N$, telle que le terme M est cohérent. Alors le terme N est un graphe acyclique cohérent.*

Démonstration: On utilise la caractérisation combinée des DAG cohérents telle que donnée par la remarque 1.3.1. Soient donc u_1 et u_2 les occurrences de deux sous-termes de N, N_1 et N_2, qui sont tels que l'on a $\tau(N_1) \preceq \tau(N_2)$. On distingue quatre cas selon la nature des occurrences u_1 et u_2.

1. Si u_1 et u_2 sont des résidus. C'est à dire qu'il existe deux occurrences v_1 et v_2 dans M, avec $u_1 \in v_1 \setminus A$ et $u_2 \in v_2 \setminus A$. Alors, puisque les étiquettes se conservent par résidu, on a $\tau(M/v_1) \preceq \tau(M/u_2)$ et donc $M/v_1 = M/v_2$. Par ailleurs, la réduction A contracte exactement les mêmes radicaux dans les deux instances M/v_1 et M/v_2, c'est à dire que l'on a finalement $N_1 = N_2$. Enfin, par le lemme 1.2.16, les occurrences u_1 et u_2 ont le même statut que les occurrences v_1 et v_2 respectivement, qui elles mêmes ont le même statut, puisque M est un DAG.

2. Si u_1 et u_2 sont toutes deux des occurrences créées, alors il existe dans le terme M deux radicaux $M/v_1 = \sigma_1(\alpha_1)$ et $M/v_2 = \sigma_2(\alpha_2)$ qui sont contractés par la réduction A. De plus, les occurrences u_1 et u_2 sont de la forme $v_1'.w_1$ et $v_2'.w_2$ respectivement, où v_1' est un héritier de v_1 et w_1 une occurrence *dure* du membre droit de la règle $\alpha_1 \to \beta_1$. Les occurrences v_2' et w_2 ayant la même signification relativement au radical M/v_2 et à la règle $\alpha_2 \to \beta_2$. Par étiquetage des règles, il existe deux étiquettes alphabétiques a_1 et a_2, avec $\tau(N/u_1) = \tau(M/v_1).a_1$ et $\tau(N/u_2) = \tau(M/v_2).a_2$. L'hypothèse $\tau(N_1) \preceq \tau(N_1)$ entraîne donc $\tau(M/v_1) \preceq \tau(M/v_2)$. Soit, puisque M est un DAG (cohérent) $M/u_1 = M/u_2$. Par l'absence de paires critiques, il vient $\alpha_1 = \alpha_2 = \alpha$ et $\beta_1 = \beta_2 = \beta$, soit encore $\sigma_1 = \sigma_2 = \sigma$. D'où il vient finalement $N/v_1' = N/v_2' = \rho(\beta)$ (comme dans le cas 1 ci-dessus, les deux instances M/v_1 et M/v_2 sont pareillement réduites par la réduction A qui contracte des radicaux caractérisés par leurs adresses). Or, le membre droit β est lui même un DAG (cohérent) et on a $\tau(\beta/w_1) \preceq \tau(\beta/w_2)$, par conséquent, il vient $\beta/w_1 = \beta/w_2$. Soit finalement $N_1 = \rho(\beta/w_1) = \rho(\beta/w_2) = N_2$. En ce qui concerne le statut des sous-termes N_1 et N_2, les héritiers v_1' et v_2' sont des occurrences autorisées. Dès lors les occurrences u_1 et u_2 ont même statut que les occurrences w_1 et w_2 respectivement, qui ont même statut, puisque le membre droit β est un DAG.

3. Supposons que u_1 est un résidu d'une occurrence v_1 de M, tandis que u_2 est créée par la contraction d'un radical d'occurrence v_2 dans M. Du point de vue des étiquettes, on a $\tau(N_1) = \tau(M/v_1)$ et $\tau(N_2) = \tau(M/v_2).a_2$, où a_2 est une étiquette alphabétique. On va voir que, dans ce cas, l'hypothèse $\tau(N_1) \preceq \tau(N_2)$ est impossible. Supposons tout d'abord $\tau(N_1) \prec \tau(N_2)$. On aurait alors $\tau(M/v_1) \preceq \tau(M/v_2)$. Or, le terme M est un DAG cohérent, on aurait donc nécessairement $M/v_1 = M/v_2$. Plus précisément, les sous-termes M/v_1 et M/v_2 seraient deux instance d'un radical partagé. Cette situation est impossible, car la réduction de graphes A doit soit contracter ces deux instances, soit n'en contracter aucune. Supposons ensuite $\tau(N_1) = \tau(N_2)$. Il viendrait alors $\tau(M/u_1) \prec \tau(M/u_2)$ qui est exclu par contradiction directe avec la cohérence de M.

4. Le cas où l'occurrence u_1 est créée (et alors il existe un radical M/v_1 contracté par A, tel que $\tau(N_1) = \tau(M/v_1).a_1$), tandis que l'occurrence u_2 est un résidu (et alors il existe une occurrence v_2 dans M, avec $\tau(N_2) = \tau(M/v_2)$), est également exclu. En effet l'hypothèse $\tau(N_1) \preceq \tau(N_2)$ entraîne ici $\tau(M/v_2) \prec \tau(M/v_1)$ et contredit directement la cohérence de M.

$\square$

La troisième condition d'orthogonalité, $\mathcal{V}_a(\beta) \subseteq \mathcal{V}_a(\alpha)$ et $\mathcal{V}_i(\beta) \subseteq \mathcal{V}_i(\alpha)$, est utilisée par la preuve du lemme ci-dessus. Cette condition est nécessaire pour pouvoir correctement combiner réduction conditionnelle et réduction de graphe. Sans elle une définition générale des dérivations de graphes deviendrait problématique, car les graphes acyclique ne seraient plus préservés par les réductions de radicaux partagés. Du point de vue des graphes, on peut dire qu'un sous-terme partagé ne change pas au cours des réductions, car il occupe toujours la même place dans le graphe (par la proposition 1.3.1) son adresse n'est pas modifiée). On peut donc voir les résidus des instances initiales comme autant d'instances du même sous-terme partagé et ces instances doivent toutes avoir le même statut qui est celui du sous-terme partagé.

Chapitre 2

Étiquettes complètes

2.1 Le problème de la participation

Les résidus ne traduisent qu'une partie de l'information qui passe du terme M dans le terme N au cours d'une réduction A: $M \to N$. Il existe dans le terme N des occurrences créées, qui ne sont des résidus d'aucune occurrence du terme initial M. On peut préciser le processus de création de ces occurrences et désigner un radical parmi ceux contractés par A qui en est plus particulièrement responsable. En effet, on montre facilement à partir du cas des réductions de type (red) que, pour toute occurrence u de N qui n'est pas un résidu, le symbole de tête du sous-terme N/u est un symbole d'un certain membre droit β d'une règle $\alpha \to \beta$ employée par la réduction A. De par l'étiquetage par diffusion des membres droits, le sous-terme N/u porte une étiquette de la forme $\tau(N/u) = \tau(\beta/w) = \tau(\alpha).a_w$. Il semble donc bien qu'un radical laisse sa marque (i.e., son adresse) sur les occurrences créées par sa contraction. Malheureusement, cette information de création ne subsiste pas quand on considère des dérivations à la place des réductions. Ce fait apparaîtra à l'aide de quelques exemples, soit donc un ensemble de termes simplement étiquetés $\mathcal{T}^g$:

Étiquettes atomiques	$\mathcal{A}$:	$s ::= a \mid b \mid a_1 \mid a_2 \mid \ldots$
Étiquettes simples	$\mathcal{A}^+$:	$l ::= s \mid s.l$
Termes	$\mathcal{T}^g$:	$M ::= x \mid x^l \mid A^l \mid B^l \mid I^l(M_1) \mid F^l(M_1, M_2)$

On se donne également les règles de réécriture simplement étiquetées :

$$
\begin{array}{lr}
(1) & I^l(x) \to x \\
(2) & A^l \to B^{l.a} \\
(3) & F^l(B^m, x) \to A^{l.b}
\end{array}
$$

Soient ensuite une dérivation quelconque A: $M \to^* N$ et un radical dans M ou radical initial. En première approximation, nous dirons qu'un radical initial participe à la dérivation A, si on peut lui attribuer la création d'une occurrence dans le terme final N. Posons, par exemple, la dérivation A égale à :

$$A{:}F^{a_1}(A^{a_2}, A^{a_3}) \xrightarrow{\{1,2\}} F^{a_1}(B^{a_2a}, B^{a_3a}) \xrightarrow{\{\epsilon\}} A^{a_1b}$$

Se pose ensuite la question de savoir lesquels, parmi les radicaux initiaux $M/1$ d'adresse a_2 et $M/2$ d'adresse a_3, sont responsables de la création de l'unique occurrence ϵ du terme final N. Il semble raisonnable de dire que le radical $M/1$ participe à la dérivation A. En effet, si le radical initial $M/1$ n'est pas contracté au cours de la première réduction, alors la deuxième réduction devient impossible tout comme la création de l'unique occurrence du terme N. Une autre façon de voir est de constater que la contraction du radical initial $M/1$ crée l'occurrence 1 dans le terme intermédiaire. Cette occurrence relais est dure dans nom $F^{a_1}(B^{a_2 a}, x)$ du radical dont la contraction par la deuxième étape crée l'occurrence ϵ dans N. En quelque sorte, la contraction du radical $M/1$ crée un radical dont la contraction ultérieure crée le terme N. Par contre, le radical $M/2$ ne participe pas à la dérivation A, comme le montre la dérivation A' suivante :

$$A' : F^{a_1}(A^{a_2}, A^{a_3}) \stackrel{\{1\}}{\to} F^{a_1}(B^{a_2 a}, A^{a_3}) \stackrel{\{\epsilon\}}{\to} A^{a_1 b}$$

Or, les adresses a_2 et a_3 des radicaux contractés à la première étape de la dérivation A disparaissent toutes les deux à la seconde. Par conséquent, il est impossible de distinguer un radical qui participe (a_2) d'un radical qui ne participe pas (a_3), en se fiant aux seules adresses. En ce sens, les étiquettes simples ne sont pas assez puissantes pour traduire la participation.

À cause des projections, la notion de participation ébauchée dans le paragraphe précédent ne peut pas se contenter de ne prendre en compte que les occurrences créées. Soit par exemple la réduction $B : M \to N$:

$$B : I^{a_1}(A^{a_2}) \stackrel{\{\epsilon\}}{\to} A^{a_2}$$

Il n'y a aucune occurrence créée dans le terme final $N = A^{a_2}$, pourtant il est évident que la contraction de l'unique radical initial M/ϵ est nécessaire pour aboutir en N. Ici encore, l'adresse a_1 du radical qui participe à la dérivation B n'apparaît pas dans le terme final.

2.2 Systèmes complètement étiquetés

Dans cette section est décrit un schéma d'étiquetage des systèmes de réécriture orthogonaux qui est tel que les informations de création et de participation se retrouvent codées dans les étiquettes.

2.2.1 Termes

Dans le cas des étiquettes simples, nous avons admis que la marque d'un radical $\sigma(\alpha)$ était son adresse. Le premier exemple de la section précédente montre que c'est insuffisant, cette marque devant contenir une information qui est dispersée dans tout le nom α du radical. Le deuxième exemple montre qu'il est nécessaire d'étiqueter les variables des membres droits, pour pouvoir traiter les règles de projection. Ces considérations me conduisent à proposer le système d'étiquettes suivant :

Étiquettes élémentaires	$\mathcal{E}$:	$e ::= a \mid \langle M,a \rangle$
Étiquettes complètes	$\mathcal{L}$:	$l ::= e \mid e.l$
Termes	$\mathcal{T}^l$:	$M ::= x \mid x^l \mid f^l(M_1, M_2, \ldots, M_a)$

Comme dans le cas des étiquettes simples, les étiquettes ne sont jamais vides. Par commodité, on considère parfois l'étiquette vide que l'on note ϵ. L'étiquette vide ϵ ne décore jamais aucun terme, mais elle peut intervenir comme composante d'une autre étiquette.

Les noms de radicaux α sont introduits dans les étiquettes des des membres droits β par l'opérateur de capture $\langle \alpha,a \rangle$. Cet opérateur est étendu par "diffusion" sur les termes dont tous les sous-termes portent une étiquette alphabétique. C'est à dire que l'on pose :

$$\langle \alpha,x^a \rangle = x^{\langle \alpha,a \rangle}$$
$$\langle \alpha,f^a(M_1, M_2, \ldots, M_n) \rangle = f^{\langle \alpha,a \rangle}(\langle \alpha,M_1 \rangle, \langle \alpha,M_2 \rangle, \ldots, \langle \alpha,M_n \rangle)$$

Cet opérateur de capture étendu permettra de laisser le nom d'un radical contracté dans les étiquettes de tête des sous-termes concernés.

La concaténation "." va maintenant servir a accumuler l'information de participation-création introduite dans les étiquettes par l'opérateur de capture. On étend la notation "." pour pouvoir ajouter une étiquette devant l'étiquette de tête d'un terme :

$$\begin{aligned} l.x &= x^l \\ l.x^m &= x^{l.m} \\ l.f^m(M_1, M_2, \ldots, M_a) &= f^{m.l}(M_1, M_2, \ldots, M_a) \end{aligned}$$

Enfin, l'étiquetage des variables des membres droits de règles induit une extension des substitutions qui utilise la concaténation étendue pour combiner les information contenues dans la variable substituée et dans le terme que l'on substitue. Une substitution σ sur $\mathcal{T}^l$ est une fonction qui satisfait les deux conditions suivantes :

$$\begin{aligned} \sigma(x^l) &= l.\sigma(x) \\ \sigma(f^l(M_1, M_2, \ldots, M_a)) &= f^l(\sigma(M_1), \sigma(M_2), \ldots, \sigma(M_a)) \end{aligned}$$

Une substitution sur $\mathcal{T}^l$ est donc déterminée uniquement par les valeurs qu'elle prend sur les variables sans étiquette.

2.2.2 Réécriture

On dispose maintenant de toutes les définitions techniques utiles pour étiqueter complètement les règles $\{\ldots, \alpha_i \to \beta_i, \ldots\}$ d'un système non-étiqueté. On procède en deux étapes :

1. On étiquette tous les sous-termes de chaque β_i, *y compris les variables*, avec des étiquettes alphabétiques. On obtient ainsi un terme étiqueté B_i.
2. Chaque règle $\alpha_i \to \beta_i$ est remplacée par un ensemble de règles étiquetées, à raison d'une règle par étiquetage *contraint* de α_i. Un étiquetage contraint est un terme étiqueté A_i qui satisfait deux restriction importantes :

   (a) Les variables de A_i ne portent pas d'étiquettes.

   (b) L'étiquette de de tête $\tau(A_i)$ est élémentaire.

   Enfin, pour chaque étiquetage contraint A_i, la règle étiquetée introduite est $A_i \to \langle A_i, B_i \rangle$.

La relation de réduction parallèle est ensuite définie par les axiomes et les règles d'inférence suivants :

$$x^l \to x^l \ \text{(id)} \qquad \frac{M_i \to N_i \text{ si } i \in \mathrm{dom}(f) \qquad M_i = N_i \text{ si } i \notin \mathrm{dom}(f)}{f^l(M_1, M_2, \ldots, M_n) \to f^l(N_1, N_2, \ldots, N_n)} \ \text{(rec)}$$

$$\frac{\sigma(x) \to \rho(x) \text{ si } x \in \mathcal{V}_a(\alpha) \qquad \sigma(x) = \rho(x) \text{ si } x \in \mathcal{V}_i(\alpha)}{l.\sigma(\alpha) \to l.\rho(\beta)} \ \text{(red)}$$

Cette définition diffère sensiblement de celle de celle des réduction parallèles simplement étiquetées par la règle d'inférence (red). En effet, un radical est maintenant de la forme générale $l.\sigma(\alpha)$ et non plus simplement $\sigma(\alpha)$. C'est à dire que le nom α d'un radical n'englobe plus la totalité de son étiquette de tête, mais seulement sa partie élémentaire la plus à droite. Cette subtilité est nécessaire pour que la réduction complètement étiquetée satisfasse la propriété de Church-Rosser, comme le suggère l'exemple qui va suivre.

$$\begin{tikzcd}
& I^a(I^b(B^c)) \arrow[dl, "\{\epsilon\}"'] \arrow[dr, "\{1\}"] \arrow[dd, "\{\epsilon,1\}"] & \\
I^{<I^a(x),a>.b}(B^c) \arrow[dr, "\{\epsilon\}"'] & & I^a(B^{<I^b(x),a>.c}) \arrow[dl, "\{\epsilon\}"] \\
& B^{<I^a(x),a>.<I^b(x),a>.c} &
\end{tikzcd}$$

Figure 2.1 : Réductions parallèles issues du terme $I^a(I^b(B^c))$

On étiquette la règle de l'identité en :

$$I^e(x) \to x^{<I^e(x),a>}$$

Puis on considère l'ensemble des dérivations issues d'un même terme donné dans la figure 2.1. La contrainte *2-(b)* permet de fermer le diagramme de Church-Rosser. En effet, les deux radicaux initiaux de noms $I^a(x)$ et $I^b(x)$ (et leurs résidus) restent parfaitement indépendants, puisque leurs noms ne se mélangent jamais. Par exemple, dans le terme le plus à gauche $I^{<I^a(x),a>.b}(B^c)$, le nom $I^a(x)$, n'intervient pas dans le nom $I^b(x)$ du radical qui reste à contracter. Ainsi, l'ordre relatif des deux noms de radicaux dans l'étiquette du terme final $B^{<I^a(x),a>.<I^b(x),a>.c}$ n'a rien à voir avec l'ordre dans lequel ces radicaux on été réduits, mais reflète plutôt leurs positions relatives dans le terme initial. Pour le cas général, on se reportera à la preuve du lemme 3.2.3 dans le chapitre suivant.

Revenons sur le système de la section précédente. Un étiquetage complet possible des trois règles de réécriture est :

$$\begin{array}{lrl}
(1) & I^e(x) & \to x^{<I^e(x),a>} \\
(2) & A^e & \to B^{<A^e,b>} \\
(3) & F^e(B^l,x) & \to A^{<F^e(B^l,x),c>}
\end{array}$$

Il faut voir chacune de ces règles comme un ensemble de règles, en faisant varier e sur l'ensemble des étiquettes élémentaires $\mathcal{E}$ et l sur l'ensemble des étiquettes complètes $\mathcal{L}$. Sous forme complètement étiquetée la dérivation A de la section précédente devient :

$$A{:}F^{a_1}(A^{a_2},A^{a_3}) \xrightarrow{\{1,2\}} F^{a_1}(B^{<A^{a_2},b>},B^{<A^{a_3},b>}) \xrightarrow{\{\epsilon\}} A^{<F^{a_1}(B^{<A^{a_2},b>},x),c>}$$

Nous pouvons maintenant constater que le nom A^{a_2} du radical initial d'occurrence 1 qui participe à la dérivation A est bien présent dans le terme final, tandis que le nom A^{a_3} du radical inutilement contracté n'y figure pas. L'information de participation se transmet donc bien par delà les dérivations dans les étiquettes complètes. Cette bonne propriété subsiste en présence de règles de projection, comme l'indiquent les dérivations de la figure 2.1.

2.2.3 Formation des étiquettes

On définit les radicaux contractés et les résidus d'occurrences comme dans le cas des étiquettes simples. Nous allons, une bonne fois pour toutes, examiner les combinaisons possibles des règles de preuve des dérivations et en déduire le processus de formation des étiquettes qui décorent les occurrences anciennes (les résidus), comme les occurrences nouvelles (les occurrences créées). Soit donc une réduction $A\colon M \to N$.

1. Si A est du type (id), alors $N = M = x^l$ et il n'y a aucune difficulté.
2. Si A est du type (rec) avec

$$\frac{A_i\colon M_i \to N_i \text{ si } i \in \mathrm{dom}(f) \quad A_i\colon M_i = N_i \text{ si } i \notin \mathrm{dom}(f)}{A\colon f^l(M_1, M_2, \ldots, M_a) \to f^l(N_1, N_2, \ldots, N_a)}\ (\mathrm{rec})$$

   Alors, on a $\tau(N/\epsilon) = \tau(N/\epsilon)$, tandis que l'étiquette d'un sous-terme N_i est déterminée par la réduction A_i uniquement.
3. Si A est du type (red) avec :

$$\frac{A_x\colon \sigma(x) \to \rho(x) \text{ si } x \in \mathcal{V}_a(\alpha) \quad A_x\colon \sigma(x) = \rho(x) \text{ si } x \in \mathcal{V}_i(\alpha)}{A\colon l.\sigma(\alpha) \to l.\rho(\beta)}\ (\mathrm{red})$$

   Occupons nous tout d'abord des occurrences de N qui ne sont pas des occurrences du membre droit β. Ces occurrences s'écrivent $u = u_x.v$, où u_x est l'occurrence d'une variable x de β et v est une occurrence *non-vide* du sous-terme $\rho(x)$. Dans ce cas, la réduction A n'intervient pas et on a $\tau(u_x.v) = \tau(\rho(x)/v)$.

   Le cas des occurrences de N qui sont aussi des occurrences de β est plus compliqué. Ces occurrences se décomposent en occurrences créées (les occurrences dures de β) et en éventuels résidus (les occurrences molles de β). Les occurrences créées n'existent que si β ne se réduit pas à une variable. Dans ce cas, l'occurrence à la racine de N est nécessairement une occurrence créée et son étiquette se forme de la façon suivante : on décompose d'abord l'étiquette de tête de M sous la forme $l.e$, où e est l'étiquette de tête (élémentaire) $\tau(\alpha)$ du nom α ; on a ensuite $\tau(N/\epsilon) = l.{<}\alpha{,}a_\epsilon{>}$, où a_ϵ est l'étiquette alphabétique choisie pour décorer la racine de β. Les autres occurrences créées dans N par cette règle (red) sont les occurrences intérieures non-vides de β. Pour une telle occurrence u, on a $\tau(N/u) = {<}\alpha{,}a_u{>}$, où a_u est une étiquette alphabétique utilisée pour décorer le sous-terme β/u. Quant aux occurrences u_x de variables x de β, c'est la réduction A_x qui détermine si u_x est un résidu ou bien une occurrence créée et forme l'étiquette de tête du terme $\rho(x)$ en conséquence. Par ailleurs, la variable x porte une étiquette de la forme ${<}\alpha{,}a_{u_x}{>}$. La réduction A intervient donc sur l'étiquette de tête du sous-terme $N/u_x = \rho(x^{<\alpha,a_{u_x}>})$, avec $\tau(N/u_x) = {<}\alpha{,}a_{u_x}{>}.\tau(\rho(x))$. Si la règle $\alpha \to \beta$ est une projection, alors la première et la troisième règle de formation des étiquettes se combinent et on a $\tau(N/\epsilon) = l.{<}\alpha{,}a_\epsilon{>}.\tau(\rho(x))$. Le tableau suivant récapitule les règles de formation de l'étiquette $\tau(N/u)$, où u est une occurrence du membre droit β.

	$\beta = x^{<\alpha,a_\epsilon>}$	$\beta = f^{<\alpha,a_\epsilon>}(\beta_1, \ldots, \beta_a)$
$u = \epsilon$	$l.{<}\alpha{,}a_\epsilon{>}.\tau(\rho(x))$	$l.{<}\alpha{,}a_\epsilon{>}$
$u \in \overline{\mathcal{O}}(\beta)$ et $u \succ \epsilon$	impossible	${<}\alpha{,}a_u{>}$
$\beta/u = x$ et $u \succ \epsilon$	${<}\alpha{,}a_u{>}.\tau(\rho(x))$	

Il est clair que les étiquettes ne se transmettent plus par résidu dans le cas général. Par exemple, avec le système de règles de la section précédente, on considère le terme $I^{a_1.a_2.a_3}(A^{a_4.a_5})$. Soient les réductions $A\colon M \to N$ et $B\colon M \to P$:

$$\begin{array}{lcl} A\colon I^{a_1.a_2.a_3}(A^{a_4.a_5}) & \xrightarrow{\{\epsilon\}} & A^{a_1.a_2.<I^{a_3}(x),a>.a_4.a_5} \\ B\colon I^{a_1.a_2.a_3}(A^{a_4.a_5}) & \xrightarrow{\{\epsilon,1\}} & B^{a_1.a_2.<I^{a_3}(x),a>.a_4.<A^{a_5},b>} \end{array}$$

Dans le premier cas (réduction A), l'étiquette de tête $a_4.a_5$ du sous-terme $M/1$ est un suffixe strict de l'étiquette de tête du sous-terme résidu N/ϵ. Dans le second cas (réduction B), l'unique occurrence ϵ du terme P porte le nom A^{a_5} du radical dont la contraction l'a créée, pris dans son étiquette élémentaire la plus à droite$<A^{a_5},b>$. Dans les deux cas, l'information de résidu ou de création est précédée d'un préfixe parasite qui peut être arbitrairement long.

Les lemmes suivants regroupent des résultats sur la formation des étiquettes complètes, ils se prouvent facilement en confrontant le tableau qui vient d'être dressé et les définitions inductives des résidus et des héritiers.

Lemme 2.2.1 (Étiquettes des résidus) *Soient une réduction $A\colon M \to N$, une occurrence u dans M et une occurrence v dans $u \setminus A$. Alors, il existe une étiquette l_1 éventuellement vide, telle que l'on a :*

$$\tau(N/v) = l_1.\tau(M/u)$$

Lemme 2.2.2 (Étiquettes des héritiers) *Soit une réduction $A\colon M \to N$. Soient une occurrence u dans M et une occurrence v dans $u\setminus\!\!\setminus\!\!\setminus A$. Alors on se trouve dans l'un des cas suivants :*

1. *L'occurrence u est de la forme $u = u'.w'$, où u' est l'occurrence dans M d'un radical de nom α et w' est une occurrence dure et non-vide de α. Alors, l'étiquette $\tau(M/u)$ toute entière est une étiquette du nom α, et il existe deux étiquettes éventuellement vides l_1 et l_2, telles que l'on a :*

$$\tau(N/v) = l_1.<\alpha,a_\epsilon>.l_2$$

2. *Le sous-terme M/u est un radical de nom α. Il existe donc une étiquette éventuellement vide l, telle que l'on a la décomposition $\tau(M/u) = l.\tau(\alpha)$. Il existe ensuite deux étiquettes éventuellement vides l_1 et l_2, avec :*

$$\tau(N/v) = l_1.l.<\alpha,a_\epsilon>.l_2$$

Dans le lemme ci-dessus l'étiquette alphabétique a_ϵ est celle qui sert à étiqueter la racine du membre droit β de la règle $\alpha \to \beta$. En outre, l'étiquette l_2 est vide dans le cas où la règle $\alpha \to \beta$ n'est pas une projection, tandis que l'étiquette l_1 est non-vide quand un radical est contracté juste au dessus de l'occurrence u.

Un autre façon de voir est d'analyser l'origine des étiquettes élémentaires dans les termes finaux des réductions.

Lemme 2.2.3 *Soit une réduction $A\colon M \to N$. Soit une occurrence u dans N, telle que l'étiquette qui décore le sous-terme N/u est de la forme $\tau(N/u) = l_1.<\alpha,a>.l_2$, où l_1 et l_2 sont des étiquettes éventuellement vides. Alors, l'une au moins des propriétés suivantes est vérifiée :*

1. *Il existe une règle de réécriture $\alpha \to \beta$ et un radical de nom α d'occurrence v dans le terme M qui est contracté par la réduction A. En outre, l'occurrence u est de la forme $u = v'.w$, avec $v' \in v\setminus\!\!\setminus\!\!\setminus A$, $w \in \mathcal{O}(\beta)$ et $\tau(\beta/w) = <\alpha,a>$.*

2. *Il existe une occurrence v dans M et deux étiquettes éventuellement vides m_1 et m_2, telles que l'on a $\tau(M/v) = m_1.\!<\!\alpha,a\!>\!.m_2$ et $u \in v\backslash\!\backslash A$.*

Remarquons que, dans le dernier cas ci-dessus, la condition générale $u \in v\backslash\!\backslash A$ correspond aux deux possibilités $u \in v \setminus A$ ou $u \in v\backslash\!\backslash\!\backslash A$, et que l'on a obligatoirement $m_2 \succ \epsilon$ si l'occurrence u est un héritier de de l'occurrence v.

2.3 Familles de radicaux

Nous avons déjà vu que le nom d'un radical constituait sa marque. Cette marque est insuffisante pour désigner un radical particulier, car deux radicaux d'occurrences distinctes peuvent porter le même nom. De tels radicaux sont regroupés par familles.

Définition 2.3.1 (Familles de radicaux) *Soit M un terme complètement étiqueté. Une famille $\mathcal{F}$ de radicaux de M est tout ensemble d'occurrences de radicaux de M qui ont le même nom. Si une famille regroupe tous les radicaux de M qui ont le même nom, elle est dite complète et on la note $\overline{\mathcal{F}}$.*

On peut donc traiter les radicaux individuellement, en considérant leurs occurrences, ou par familles complètes, en considérant leurs noms. Cette section étudie les radicaux résidus adoptant successivement chacun de ces deux points de vue. La section 2.4 suivante fait de même pour les radicaux crées.

2.3.1 Contextes et préfixes

Les occurrences ne peuvent pas rendre compte de toutes les situations. Par exemple, pour pouvoir désigner précisément un radical, il est utile de pouvoir considérer le nom du radical en question en plus de son occurrence.

Définition 2.3.2 (Contextes) *Soit M un terme. Un contexte dans M est une paire (u, γ), composée d'une occurrence u dans M et d'un terme γ, tel qu'il existe une étiquette éventuellement vide l et une substitution σ, avec $l.\sigma(\gamma) = M/u$.*

Étant donnés un terme M et un contexte (u, γ) de M, on dit qu'une occurrence v dans M *aboutit* dans le contexte (u, γ), si et seulement si il existe une occurrence w dans γ, telle que $v = u.w$. Si l'occurrence w est dure dans γ, alors v aboutit *à l'intérieur* du contexte (u, γ).

Les contextes rendent compte des membres gauches comme des membres droits de règles. À un radical d'occurrence u et de nom α d'un terme M, on peut associer le *contexte de radical* (u, α). La condition de non-recouvrement des noms de radicaux revient à dire qu'il n'existe pas d'occurrence qui aboutisse à l'intérieur de deux contextes de radicaux différents.

Les préfixes sont des contextes particuliers.

Définition 2.3.3 (Préfixes) *Soit M, un terme. Un préfixe de M est tout terme γ, tel qu'il existe une substitution σ, avec $\sigma(\gamma) = M$.*

Étant donné un terme M, un radical à la racine $M = l.\sigma(\alpha)$ dans un terme M, définit le préfixe $l.\alpha$ de M. Remarquons qu'à un contexte (u, γ) du le terme M $(M/u = l.\sigma(\gamma))$, correspond un préfixe $l.\gamma$ dans le sous-terme M/u $(\sigma(l.\gamma) = M/u)$ et ceci, même lorsque le terme γ est réduit 'a une variable étiquetée ou non.

Si le terme d'un contexte ou d'un préfixe est linéaire, on parle de contexte ou de préfixe linéaires. Il est évident que les noms des variables d'un préfixe linéaire importent peu, en particulier deux termes linéaires qui sont mutuellement préfixes l'un de l'autre sont les mêmes termes aux variables près.

Remarque 2.3.1 *Étant donné un terme M, l'équivalence :*

$$(u, f^l(\gamma_1, \ldots, \gamma_a)) \textit{ est un contexte dans } M \Leftrightarrow \begin{cases} M/u = f^{m.l}(M_1, \ldots, M_a) \\ \forall i \in [1 \ldots a], (u.i, \gamma_i) \textit{ est un contexte dans } M \end{cases}$$

n'est vérifiée en général que par les termes $f^l(\gamma_1, \ldots, \gamma_a)$ qui sont linéaires.

Démonstration : Seul le sens réciproque $\Leftarrow$ utilise la linéarité, pour combiner les substitutions σ_i telles que $\sigma_i(\gamma_i) = M_i$ en une substitution $\sigma = \sigma_1 \cup \sigma_2 \cup \ldots \cup \sigma_a$, avec $\sigma(\gamma) = f^l(M_1, \ldots, M_a)$. □

2.3.2 Résidus des familles de radicaux

Dans cette section, il est d'abord montré que les noms de radicaux se conservent par résidu. Cette "évidence" nécessite l'exposition préalable de deux lemmes techniques.

Lemme 2.3.4 *Soit un terme complètement étiqueté M et une étiquette l. On a l'équivalence :*

$$A \colon M \xrightarrow{U} N \Leftrightarrow B \colon l.M \xrightarrow{U} l.N$$

Démonstration : Montrons tout d'abord le sens direct $A \Rightarrow B$. Le résultat est immédiat quand A est du type (id) ou (rec). Soit maintenant une réduction A du type (red), avec $M = m.\sigma(\alpha)$ et $N = m.\rho(\beta)$. Plus précisément, on pose :

$$\frac{A_x \colon \sigma(x) \to \rho(x) \text{ si } x \in \mathcal{V}_a(\alpha) \qquad A_x \colon \sigma(x) = \rho(x) \text{ si } x \in \mathcal{V}_i(\alpha)}{A \colon m.\sigma(\alpha) \to m.\rho(\beta)} \text{(red)}$$

De par l'associativité de "." on peut écrire $l.M = (l.m).\sigma(\alpha)$ et $l.N = (l.m).\rho(\beta)$. Les prémisses de la preuve de la réduction A sont suffisantes pour prouver B :

$$\frac{A_x \colon \sigma(x) \to \rho(x) \text{ si } x \in \mathcal{V}_a(\alpha) \qquad A_x \colon \sigma(x) = \rho(x) \text{ si } x \in \mathcal{V}_i(\alpha)}{B \colon (l.m).\sigma(\alpha) \to (l.m).\rho(\beta)} \text{(red)}$$

Enfin, puisque les arbres de preuve des réductions A et B sont les mêmes à une étiquette près, elles contractent toutes deux le même ensemble U de radicaux.

Réciproquement, supposons qu'il existe une réduction $B \colon l.M \xrightarrow{U} l.N$. Ici encore, seul le cas (red) pose une petite difficulté. Si B est de type (red) alors il existe un nom de radical α, une étiquette éventuellement vide m et une substitution σ avec $l.M = m.\sigma(\alpha)$, la réduction B étant prouvée par une règle (red) dont la conclusion est $B \colon m.\sigma(\alpha) \to m.\rho(\beta)$. Les membres gauches de règles ne peuvent pas se réduire à des variables, le terme $l.M$ s'écrit donc comme $l.M = f^{m.e}(M_1, M_2, \ldots, M_a)$. Le terme M est un terme étiqueté valide, il porte donc en tête une étiquette non-vide l'. De $l.M = m.\sigma(\alpha)$, on tire $l.l' = m.e$. L'étiquette l', qui n'est pas vide, s'écrit donc $l' = m'.e$, où m' peut être vide. Soit finalement, on peut écrire M sous la forme $M = m'.\sigma(\alpha)$ et prouver la réduction $A \colon m'.\sigma(\alpha) \to m'.\rho(\beta)$ par une règle d'inférence de type (red), dont les prémisses sont celles de la preuve de la réduction B. □

Ce lemme est moins gratuit qu'il n'y paraît, il rend rend compte du cas des variables étiquetées dans l'important lemme du préfixe préservé qui suit.

Définition 2.3.5 *Soit une réduction $A\colon M \xrightarrow{U} N$ et soit γ un préfixe* linéaire *de M. Par définition, le préfixe γ est préservé par la réduction A, si et seulement si il n'existe pas d'occurrence dans U qui est une occurrence dure de γ.*

Lemme 2.3.6 *Soit une réduction $A\colon M \to N$ et soit γ un préfixe préservé par la réduction A. Alors, le terme γ est un préfixe du terme N.*

Démonstration : On procède par induction sur le préfixe γ. Le cas de base $\gamma = x$ est immédiat, puisqu'une variable est préfixe de tout terme et n'a pas d'occurrence dure. Le cas de base $\gamma = x^l$, se résoud par le lemme 2.3.4. Sinon, le préfixe γ est de la forme $f^l(\gamma_1, \dots, \gamma_a)$, on a donc $M = f^l(M_1, \dots, M_a)$, avec, pour tout indice i compris entre 1 et a, l'égalité $M_i = \sigma(\gamma_i)$. L'occurrence à la racine ϵ est dure dans γ, elle ne peut donc pas appartenir à $\mathcal{R}(A)$ et A ne peut pas être du type (red). La réduction A est donc du type (rec), avec :

$$\frac{A_i\colon M_i \to N_i \text{ si } i \in \mathrm{dom}(f) \quad A_i\colon M_i = N_i \text{ si } i \notin \mathrm{dom}(f)}{A\colon f^l(M_1, M_2, \dots, M_a) \to f^l(N_1, N_2, \dots, N_a)}\ \text{(rec)}$$

Pour tout indice i autorisé à partir de f, le terme γ_i est un préfixe linéaire de M_i. Soit u une occurrence de $\mathcal{R}(A_i)$, l'occurrence $i.u$ appartient à $\mathcal{R}(A)$ et donc, par hypothèse, n'est pas une occurrence dure dans γ. Par conséquent, l'occurrence u n'est pas une occurrence dure de γ_i. On peut donc appliquer l'hypothèse d'induction et prouver que γ_i est un préfixe de N_i. Pour tout indice i interdit à partir de f, on a $N_i = M_i$, et donc γ_i est un préfixe de N_i là aussi. Puisque γ est linéaire, on peut conclure en disant que γ est un préfixe de N (cf. la remarque 2.3.1). $\square$

Des préfixes, nous allons passer aux contextes et plus particulièrement aux contextes de radicaux. Jusqu'ici nous avons allègrement confondu occurrences de radicaux et radicaux eux même. Le lemme suivant justifie cette confusion dans le cas des résidus. C'est à dire que l'on pourra désormais parler de résidus de radicaux.

Lemme 2.3.7 *Soit une réduction $A\colon M \to N$. Soit u une occurrence dans M et v une occurrence de $u \setminus A$. Si le sous-terme M/u est un radical de nom α, alors le sous-terme N/v est aussi un radical de nom α.*

Démonstration: La preuve est par induction sur la structure de A.

1. Si A est du type (id), alors le terme M se réduit à une variable et ne contient pas de radical.
2. Si A est du type (rec) avec

$$\frac{A_i\colon M_i \to N_i \text{ si } i \in \mathrm{dom}(f) \quad A_i\colon M_i = N_i \text{ si } i \notin \mathrm{dom}(f)}{A\colon f^{l'}(M_1, M_2, \dots, M_a) \to f^{l'}(N_1, N_2, \dots, N_a)}\ \text{(rec)}$$

   Il y a deux sous-cas, selon la valeur de u :

   (a) Cas $u = \epsilon$. Alors c'est le terme M qui est un radical de nom α. C'est à dire que M admet un préfixe *linéaire* de la forme $m.\alpha$ (avec $l = m.\tau(\alpha)$). Par la condition de non-recouvrement des noms de radicaux, il ne peut pas y avoir de radical contracté par A dont l'occurrence dans M est une occurrence dure du préfixe $m.\alpha$. On peut donc utiliser le lemme 2.3.6 pour prouver que $m.\alpha$ est un préfixe de N. Autrement dit, l'unique résidu ϵ de l'occurrence ϵ par A est l'occurrence d'un radical dont le nom est α.

(b) Cas $u = i.u'$, où i est un indice autorisé et u' une occurrence d'un radical de N_i. Par définition des résidus, pour toute occurrence v de $u \setminus A$, il existe une occurrence v' dans $u' \setminus A_i$ avec $v = i.v'$. Par hypothèse d'induction, N_i/v' est un radical de nom α. Enfin, $N/v = N_i/v'$ est aussi un radical de nom α, puisque l'occurrence $i.v'$ est autorisée comme l'est l'occurrence v'.

3. Si A est du type (red) avec :

$$\frac{A_x\colon \sigma(x) \to \rho(x) \text{ si } x \in \mathcal{V}_a(\alpha) \quad A_x\colon \sigma(x) = \rho(x) \text{ si } x \in \mathcal{V}_i(\alpha)}{A\colon l'.\sigma(\alpha') \to l'.\rho(\beta')}\,(\text{red})$$

Par définition des résidus, il existe une variable x, d'occurrence u_x dans α, et dont v_x est une occurrence dans β, telle que l'on a $u = u_x.u'$ et $v = v_x.v'$, avec $v' \in u' \setminus A_x$. Ici encore, on peut appliquer l'hypothèse d'induction à u' et à v', puis utiliser le fait que $v_x.v'$ est une occurrence autorisée dans N. Il faut en outre remarquer que, dans le cas $v' = \epsilon$, la modification de l'étiquette de tête $\tau(M/v_x) = <\alpha', a_{v_x}>.\tau(\rho(x))$ n'affecte pas le nom du radical à la racine dans $\rho(x)$ qui devient le radical d'occurrence v_x dans N. □

Ce lemme est important dans deux aspects. Tout d'abord, les noms des radicaux se conservent par résidu, c'est à dire que les étiquettes complètes apparaissent comme un moyen raisonnable de suivre les radicaux dans les réductions, alors que les étiquettes simples sont plutôt utiles pour suivre les occurrences. Ensuite, les résidus de radicaux sont des radicaux, une condition nécessaire pour pouvoir se réserver la possibilité de faire abstraction de l'ordre relatif de contraction des radicaux d'un même terme. En effet, étant donnés un terme M et deux ensembles de radicaux U et V dans M, c'est à dire deux réductions $A\colon M \xrightarrow{U} N$ et $B\colon M \xrightarrow{V} P$, il existe une réduction *résidu* de A par B (notée naturellement $A \setminus B$), qui est la réduction issue du terme P qui contracte les radicaux dont les occurrences sont celles de $U \setminus B$. De même, la réduction $B \setminus A$ est la réduction issue du terme N et qui contracte les radicaux résidus $V \setminus A$. On appelle *dérivations permutées* les deux dérivations $A(B \setminus A)$ et $B(A \setminus B)$. L'importance de cette nouvelle notion justifie une notation particulière :

Notation : Soient deux réductions $A\colon M \to N$ et $B\colon M \to P$, issues d'un même terme M. On note $A \sqcup B$ la dérivation composée de la réduction A suivie de la réduction résidu $B \setminus A$.

Une dernière remarque sur ce lemme concerne un point un peu technique de sa preuve. Pour prouver que les résidus de radicaux sont des radicaux, on utilise toutes les condition d'orthogonalité sauf une. L'absence de paires critiques et la linéarité des noms de radicaux sont directement invoquées. La condition $\mathcal{V}_i(\beta) \subseteq \mathcal{V}_i(\alpha)$ est utilisée sous sa forme “les résidus des occurrences autorisées sont des occurrences autorisées”. Il n'est pas fait usage de la dernière condition $\mathcal{V}_a(\beta) \subseteq \mathcal{V}_a(\alpha)$, dont nous verrons dans la section suivante qu'elle a plus à voir avec les radicaux qui ne sont pas des résidus.

Enfin, il n'est pas nécessaire de considérer les résidus d'occurrences pour définir les résidus de radicaux. Ceci reste valable même quand on considère les résidus par des dérivations. Plus précisément, la relation de résidu entre radicaux par delà les dérivations peut être introduite comme la fermeture transitive de la relation de résidu pour les réductions. Pour toute dérivation $A\colon M {\to^*} N$, telle que $A = BC$ avec $B\colon M {\to^*} P$ et $C\colon P {\to^*} N$, tout radical R dans M et tout radical S dans N, on pose :

$$S \in R \setminus A \Leftrightarrow \exists R' \in \mathcal{R}(P),\ R' \in R \setminus B \text{ et } S \in R' \setminus C$$

Le lemme 2.3.7 s'étend ensuite aux dérivations directement par transitivité.

Lemme 2.3.8 *Soit une dérivation A: $M \to^* N$. Soit u une occurrence dans M et v une occurrence de $u \setminus A$. Si le sous-terme M/u est un radical de nom α, alors le sous-terme N/v est aussi un radical de nom α.*

En termes de familles de radicaux, les résidus par une dérivation A d'une famille $\mathcal{F}$ de radicaux de nom α forment une famille de nom α que l'on note naturellement $\mathcal{F} \setminus A$.

2.4 Radicaux créés

Il est tentant de penser qu'une démarche analogue à celle de l'étude des résidus de radicaux peut être suivie dans le cas de la création de radicaux : définir et étudier, dans un premier temps, les radicaux crées par réduction à l'aide des occurrences crées ; puis, dans un second temps, étendre par transitivité cette définition aux dérivations'. Malheureusement, il est impossible de procéder aussi simplement et cette section se limite à une relation de création particulière.

2.4.1 Le cas des réductions

Étant donnée une réduction A: $M \xrightarrow{U} N$, rappelons que les occurrences créées par A dans le terme N sont les occurrences de N qui ne sont pas résidus d'une occurrence de N. La définition suivante du radical créateur d'une occurrence reprend l'intuition évoquée au tout début de ce chapitre.

Définition 2.4.1 (Créateur d'une occurrence) *Soit une réduction A: $M \xrightarrow{U} N$. Soit u une occurrence de radical prise dans U. On définit l'ensemble $u \mapsto A$ des occurrences créées dans N par la contraction du radical d'occurrence u au cours de la réduction A :*

1. *Si la réduction A est du type (id), alors il n'y a ni radicaux contractés ni occurrences créées.*
2. *Si la réduction A est du type (rec), avec :*

$$\frac{A_i\colon M_i \to N_i \ \ si\ i \in dom(f) \quad A_i\colon M_i = N_i \ \ si\ i \notin dom(f)}{A\colon f^l(M_1, M_2, \ldots, M_a) \to f^l(N_1, N_2, \ldots, N_a)}\ \text{(rec)}$$

   Alors, pour tout indice i du domaine du symbole f et toute occurrence u d'un radical contracté par la réduction A_i, on pose :

$$(i.u) \mapsto A = i.(u \mapsto A_i)$$

3. *Enfin, si la réduction A est du type (red), avec :*

$$\frac{A_x\colon \sigma(x) \to \rho(x) \ \ si\ x \in \mathcal{V}_a(\alpha) \quad A_x\colon \sigma(x) = \rho(x) \ \ si\ x \in \mathcal{V}_i(\alpha)}{A\colon l.\sigma(\alpha) \to l.\rho(\beta)}\ \text{(red)}$$

   Alors, les occurrences créées par la contraction du radical à la racine sont les occurrences dures du membre droit β :

$$\epsilon \mapsto A = \overline{\mathcal{O}}(\beta)$$

   Les autres occurrences créées le sont par les réductions A_x. C'est à dire que, pour toute variable x d'occurrence u_x autorisée dans le membre gauche α, on pose :

$$(u_x.u) \mapsto A = \{\, v_x \in \mathcal{O}(\beta) \mid \beta/v_x = x \,\}.(u \mapsto A_x)$$

Soit R le radical d'occurrence u contracté par la réduction A dans la définition ci-dessus. Soit le nom α du radical R. Les occurrences de $u \mapsto A$ sont de la forme $v.w$, où v est un héritier de u par A et w une occurrence dure du membre droit β associé au membre gauche α. Du point de vue des étiquettes, on a les égalités $\tau(\beta/w) = <\alpha, a_w>$ et $\tau(N/(v.w)) = l.<\alpha, a_w>$, où l est une étiquette qui peut ne pas être vide quand l'occurrence w est vide. Enfin, si R est un radical associé à une règle de projection, alors l'ensemble $u \mapsto A$ est toujours vide. Dans ce cas, les héritiers de u par A portent une étiquette de la forme générale $l_1.<\alpha, a_\epsilon>.l_2$, où l_2 est une étiquette non-vide (lemme 2.2.2).

Étant donnée une réduction A: $M \to N$, un radical R du terme N qui n'est pas résidu d'un radical de M est un radical créé par la réduction A. On peut, ici aussi, être plus précis :

Définition 2.4.2 (Radical créateur d'un radical) *Soit une réduction A: $M \to N$. Soit un radical R d'occurrence u et de nom α, créé par la réduction A dans son terme final N. Soit enfin un radical R' d'occurrence u' dans M, contracté par la réduction A. Par définition, le radical R' est un créateur du radical R, si et seulement si on se trouve dans l'un des deux cas suivants :*

1. *Une occurrence créée par R' aboutit à l'intérieur du contexte (u, α).*
2. *Un héritier de u' différent de u aboutit à l'intérieur du contexte (u, α).*

Autrement dit, on pose :

$$R' \textit{ est un créateur de } R \Leftrightarrow \begin{cases} \exists v \in \overline{\mathcal{O}}(\alpha),\ u.v \in u' \mapsto A \\ \quad ou \\ \exists v \in \overline{\mathcal{O}}(\alpha),\ v \succ \epsilon \textit{ et } u.v \in u' \backslash\!\backslash\!\backslash A \end{cases}$$

Un créateur est donc un radical dont la contraction intervient dans la formation du nom du radical créé. Du fait du cas *2* de la définition du radical créateur, le nom d'un radical créé ne contient pas toujours une occurrence qui est une occurrence créée. Ceci n'est vraiment apparent que dans le cas des projections. Par exemple, si l'on retrouve les trois règles de réécriture du début de ce chapitre :

$$I^e(x) \to x^{<I^e(x), a>} \qquad A^e \to B^{<A^e, b>} \qquad F^e(B^l, x) \to A^{<F^e(B^l, x), c>}$$

Et que l'on se donne la réduction A: $M \to N$ suivante :

$$A\colon F^{a_1}(I^{a_2}(B^{a_3}), A^{a_4}) \xrightarrow{\{1\}} F^{a_1}(B^{<I^{a_2}(x), a>.a_3}, A^{a_4})$$

Le terme final N contient un radical créé à la racine. Le nom $\alpha = F^{a_1}(B^{<I^{a_2}(x), a>.a_3}, x)$ de ce radical n'a aucune occurrence intérieure créée, puisque $\epsilon \in \epsilon \setminus A$ et $1 \in 1.1 \setminus A$. Par contre, l'occurrence 1 est l'unique héritier par A de l'occurrence du radical $M/1 = I^{a_2}(\ldots)$ qui est contracté par la réduction A. Le radical $M/1$ est donc un créateur du radical N/ϵ.

De façon plus générale, tout radical créé admet au moins un créateur.

Lemme 2.4.3 *Soit une réduction A: $M \to N$. Soit un radical R créé dans N par A. Alors, il existe un radical R' contracté par A qui est un créateur de R.*

Démonstration: On appelle α le nom du radical R et on procède par induction sur la réduction A.

1. Si A est du type (id), alors il n'y a pas de radical dans $N = x^l$.

2. Si A est du type (rec), avec :

$$\frac{A_i\colon M_i \to N_i \text{ si } i \in \mathrm{dom}(f) \quad A_i\colon M_i = N_i \text{ si } i \notin \mathrm{dom}(f)}{A\colon f^l(M_1, M_2, \ldots, M_a) \to f^l(N_1, N_2, \ldots, N_a)} \text{(rec)}$$

Si le radical R est d'occurrence ϵ dans le terme N, alors le préfixe $m.\alpha$ (en posant $l = m.\tau(\alpha)$) ne peut pas être un préfixe du terme M. En effet, si tel était le cas, le terme M admettrait un radical à la racine. Donc, par le lemme 2.3.7 et puisque l'on a ici $\epsilon \setminus A = \{\epsilon\}$, le radical R serait un résidu, ce qui est exclu par hypothèse. Il existe donc une occurrence u non-vide et dure dans α ($\alpha/u = f_1^{l_1}(\alpha_1, \ldots, \alpha_{a_1})$), telle que l'on a :

$$M/u = x^{l_2} \quad \text{ou} \quad M/u = f_2^{l_2}(P_1, \ldots, P_{a_2}), \text{ avec } f_1^{l_1} \neq f_2^{l_2}$$

C'est à dire que les sous-termes M/u et N/u n'ont pas la même tête. Dès lors, par le lemme 1.2.10 (qui reste valide dans les systèmes complètement étiquetés), l'occurrence u ne peut pas être préservée par la réduction A. Donc, il existe une occurrence v dans U, telle que l'on a $v \preceq u$. Il est clair que l'on peut choisir v de façon minimale (i.e., $v \in \mathit{inf}_{\preceq}(U) = \mathcal{E}xt(A)$). Dès lors, par le lemme 1.2.15, on a $v \backslash\!\backslash\!\backslash A = \{v\}$. Par ailleurs, en se replaçant dans le terme N il vient $v \in \overline{\mathcal{O}}(\alpha)$ de $v \preceq u$ et $u \in \overline{\mathcal{O}}(\alpha)$. C'est à dire que, finalement, le radical d'occurrence v dans M est un créateur du radical R par le cas *2* de la définition. Notons que, si la règle de réécriture associée au radical d'occurrence v dans M n'est pas une projection, on peut tout aussi bien invoquer le cas *1* de la définition du créateur puisque, dans ce cas, l'occurrence v est une occurrence créée dans N.

Si le radical R est d'occurrence $i.u$ dans le terme N, alors il y a un radical d'occurrence u dans le terme N_i. Par induction, ce dernier admet un créateur d'occurrence v dans le terme M_i, avec $v \in \mathcal{R}(A_i)$. On vérifie facilement, qu'il y a bien un radical contracté par A à l'occurrence $i.v$ dans M ($i.\mathcal{R}(A_i) \subseteq \mathcal{R}(A)$) et que ce radical est un créateur de R (voir les deux cas de la définition du créateur, puis les définitions inductives des occurrences créées et des héritiers).

3. Si A est de type (red), avec :

$$\frac{A_x\colon \sigma(x) \to \rho(x) \text{ si } x \in \mathcal{V}_a(\alpha') \quad A_x\colon \sigma(x) = \rho(x) \text{ si } x \in \mathcal{V}_i(\alpha')}{A\colon l'.\sigma(\alpha') \to l'.\rho(\beta')} \text{(red)}$$

Si l'occurrence du radical R est une occurrence dure de β' alors le radical à la racine dans M est un créateur de R par le cas *1* de la définition.

Sinon, l'occurrence de R est de la forme $u_x.u$, où u_x est l'occurrence *autorisée* d'une variable x de β', tandis que u est l'occurrence d'un radical dans le sous-terme $M/u_x = \rho(x)$. Soit v_x l'occurrence de la variable x dans α'. D'après la condition d'orthogonalité $\mathcal{V}_a(\beta') \subseteq \mathcal{V}_a(\alpha')$, l'occurrence v_x est autorisée dans α'. Il existe donc une réduction $A_x\colon \sigma(x) \to \rho(x)$, à laquelle on applique l'hypothèse d'induction pour prouver 'existence d'un radical R' d'occurrence v dans le sous-terme $\sigma(x)$ qui est est un créateur du radical d'occurrence u dans $\rho(x)$. On termine ensuite la preuve comme dans cas 2 ci-dessus (Si l'occurrence v est un créateur par A_x de l'occurrence w, alors l'occurrence $u_x.v$ est un créateur par A de l'occurrence $v_x.w$. Sinon, l'occurrence w est un héritier par A_x de de l'occurrence v, et alors l'occurrence $v_x.w$ est un héritier par A de l'occurrence $u_x.v$). □

Ce lemme utilise la condition d'orthogonalité "les variables autorisées des membres droits sont des variables autorisées du membre gauche correspondant". Cette condition se justifie donc parce qu'elle assure la localité de la création de radical : pour créer un radical, il faut intervenir localement au niveau de son nom. Si les résidus des occurrences interdites pouvaient être des occurrences autorisées, alors la contraction d'un certain radical pourrait créer un radical à distance en modifiant le statut d'un sous-terme qui est déjà de la forme $l.\sigma(\alpha)$, tout en étant situé à une occurrence interdite sous le radical "créateur".

À lui seul, ce lemme justifie la définition des créateurs. En outre, cette définition un peu compliquée donne lieu à une propriété très directe des noms de radicaux.

Définition 2.4.4 (Contribution directe) *Soit α et γ deux noms de radicaux. Par définition, le nom α contribue directement à γ, si et seulement si une des étiquettes de γ est de la forme $l_1.<\alpha,a>.l_2$, où l_1 et l_2 sont des étiquettes éventuellement vides et a est une étiquette atomique. On note $\alpha < \gamma$ la relation de contribution directe.*

La relation de contribution entre noms de radicaux rend compte de la création de radicaux au cours des réductions.

Lemme 2.4.5 *Soit une réduction A: $M \to N$. Soient un radical $R = l.\sigma(\alpha)$ d'occurrence u dans le terme final N et un radical $R' = l'.\sigma'(\alpha')$ d'occurrence u' dans le terme initial M. Si le radical R' est un créateur du radical R, alors le nom α' contribue directement au nom α.*

Démonstration: On revient sur les deux cas de la définition du radical créateur :

1. Si il existe une occurrence créée par la contraction de R' de la forme $u.w$, avec $w \in \overline{\mathcal{O}}(\alpha)$, alors on a $\tau(\alpha/u) = l_1.<\alpha',a'>$ (voir la remarque sur l'étiquetage des occurrences créées qui suit la définition 2.4.1).
2. Si il existe un héritier de u' de la forme $u.w$, avec $w \succ \epsilon$ et $w \in \overline{\mathcal{O}}(\alpha)$, alors son étiquette $\tau(N/u.w) = l_1.<\alpha',a'_\epsilon>.l_2$ (lemme 2.2.2, cas *2*) est toute entière une étiquette du nom α. □

Ce lemme renforce l'intérêt de la condition $\mathcal{V}_a(\beta) \subseteq \mathcal{V}_a(\alpha)$. Grâce à elle, les créateurs d'un radical donné laissent leur marque dans les étiquettes du nom du radical créé. Il donne aussi une nouvelle justification de l'étiquetage contraint des noms de radicaux (section 2.2). Soit la réduction :

$$I^{a_1}(F^{a_2}(I^{a_3}(A^{a_4}), I^{a_5}(B^{a_6}))) \overset{\{\epsilon,1,1,1,2\}}{\to} F^{<I^{a_1}(x),a>.a_2}(B^{<I^{a_3}(x),a>.<A^{a_4},b>}, B^{<I^{a_5}(x),a>.a_6})$$

Ici, le terme final de la réduction admet un radical créé à la racine. Par définition du radical créateur, les radicaux initiaux de noms $I^{a_1}(x)$ et $I^{a_5}(x)$ ne contribuent pas à la création de ce radical. Or, si le nom $F^{a_2}(B^{<I^{a_3}(x),a>.<A^{a_4},b>}, x)$ du nouveau radical ne contient pas les noms de ces deux radicaux qui lui sont indifférents, c'est bien grâce aux contraintes qui portent sur son étiquette à la racine (qui est élémentaire) et sur la variable x (qui n'est pas étiquetée).

Malheureusement, la notion de radical créateur se généralise mal au cas des dérivations. Considérons un exemple un peu différent :

$$A: F^{a_2}(I^{a_3}(A^{a_4}), B^{a_6}) \overset{\{1\}}{\to} F^{a_2}(A^{<I^{a_3}(x),a>.a_4}, B^{a_6}) \overset{\{1\}}{\to} F^{a_2}(B^{<I^{a_3}(x),a>.<A^{a_4},b>}, B^{a_6})$$

Appelons A_1 et A_2 les deux étapes de la dérivation A. La réduction A_2 crée un radical d'occurrence ϵ dans le terme final. Conformément à la définition de la création par réduction, le radical de nom A^{a_4} est un créateur de ce nouveau radical. Mais ce radical A^{a_4} contracté par la réduction A_2 n'est pas seul responsable de la création du nouveau radical $F^{a_2}(B^{<I^{a_3}(x),a>.<A^{a_4},b>}, x)$. Il semble bien qu'il faille également en créditer de radical $I^{a_3}(x)$ qui a été contracté par la première étape A_1. Pour s'en convaincre, on peut considérer la réduction A' qui contracte ensemble les radicaux contractés séparément par la dérivation A.

$$A'\colon F^{a_2}(I^{a_3}(A^{a_4}), B^{a_6}) \overset{\{1,2\}}{\longrightarrow} F^{a_2}(B^{<I^{a_3}(x),a>.<A^{a_4},b>}, B^{a_6})$$

Dans cette réduction, le radical de nom $I^{a_3}(x)$ est un créateur du radical créé par A' dans son terme final (deuxième cas de la définition 2.4.2). Pourtant, si l'on revient à la première étape A_1 de la dérivation A, on constate qu'il n'y a pas de radical créé par A_1 dans le terme intermédiaire entre les réductions A_1 et A_2. Par conséquent, l'information de création ne peut pas survivre entre A_1 et A_2 en ne faisant intervenir que des radicaux.

Autrement dit, on voit mal comment généraliser aux dérivations la relation de création entre radicaux, comme on l'a fait pour les résidus.

2.4.2 Le cas des dérivations

Pour éviter une étude compliquée du gel dans les occurrences de l'information de création entre radicaux, je limite la définition des radicaux créateurs par delà les dérivations à un cas particulier.

Définition 2.4.6 (Créateur privilégié) *Soit une dérivation $A\colon M_0 \overset{U_1}{\to} M_1 \overset{U_2}{\to} \dots \overset{U_n}{\to} M_n$. Soient un radical R dans le terme N et un radical R' dans le terme M. Le radical R' est un créateur privilégié de R par la dérivation A, si et seulement si R n'est pas un résidu de R' et qu'il existe une suite de radicaux $(R_i)_{0\leq i\leq n}$, telle que :*

- *On a $R_0 = R'$ et $R_n = R$.*
- *Pour tout entier i de l'intervalle $[1 \dots n]$, le radical R_i est un radical de M_i, tel que R_i est un résidu de R_{i-1} ou bien R_{i-1} est un créateur de R_i.*

Soient $A\colon M{\to}^*N$ et $B\colon N{\to}^*P$ deux dérivations concaténables. Il est clair que, pour tout radical R créé par la dérivation AB dans le terme P, le radical R' du terme M est un créateur privilégié du radical R, si et seulement si il existe un radical R'' dans le terme N, tel que l'on se trouve dans une des trois situations suivantes :

- Le radical R' est un créateur privilégié du radical R'', qui est lui même un créateur privilégié du radical R.
- Le radical R'' est un résidu du radical R' et un créateur privilégié du radical R.
- Le radical R' est un créateur privilégié du radical R'', dont le radical R est un résidu.

La définition du créateur privilégié est suffisante pour pouvoir étendre aux dérivations le lemme d'existence du créateur.

Lemme 2.4.7 *Soit une dérivation $A\colon M{\to}^*N$ et soit R un radical dans le terme final N, tel que R n'est résidu d'aucun radical du terme initial M (i.e., le radical R est créé par la dérivation A). Alors il existe dans le terme M un radical R' qui est un créateur privilégié de R.*

Démonstration: La preuve est par induction sur la longueur de la dérivation A. Si la dérivation A est la dérivation vide, alors il n'y a pas de radical créé par A. Sinon, on décompose A sous la forme $A = BC$, où $B\colon M \to P$ est une réduction et $C\colon P{\to}^*N$. Deux cas sont possibles :

- Si il existe dans le terme P un radical R'', tel que l'on a $R \in R'' \setminus C$, alors, puisque le radical R est créé par la dérivation A, le radical R'' est nécessairement créé par la réduction B. Par le lemme 2.4.3, il existe dans le terme M un radical R qui est un créateur de R'' et donc un créateur privilégié de R.
- Sinon, par définition, le radical R est créé par la dérivation C. Par induction, il existe donc un créateur privilégié de R dans le terme P. Soit R'' ce radical créateur. Enfin, que R'' soit résidu par la réduction B, d'un radical R' de M ou que R'' soit créé par la réduction B (auquel cas il existe dans le terme M un créateur R' de R'' par la réduction B), il existe bien, dans le terme terme M, un radical R' qui est un créateur privilégié du radical R par la dérivation A. $\square$

La relation de création de radicaux a une traduction très directe en termes d'étiquettes complètes.

Définition 2.4.8 *La relation de contribution entre noms de radicaux (notée $<^+$) est la fermeture transitive de la relation de contribution directe $<$. On définit également la relation $<^*$, comme la fermeture transitive et réflexive de la relation de contribution directe $<$.*

Lemme 2.4.9 *Soit une dérivation A: $M \to^* N$. Soient un radical R de nom α dans le terme final N et un radical R' de nom α' dans le terme initial M, tels que R' est un créateur privilégié de R. Alors, on a la relation $\alpha' <^+ \alpha$.*

Démonstration : Ce lemme est la fermeture transitive du lemme 2.4.5. $\square$

Ce dernier résultat permet de préciser partiellement le mécanisme de formation des noms des familles de radicaux.

Lemme 2.4.10 *Soit une dérivation A: $M \to^* N$. Soit un radical R de nom α dans N. Alors il existe dans le terme M un radical R' de nom α', avec $\alpha' <^* \alpha$.*

Démonstration : Si il existe un radical R' dans M, tel que l'on a $R \in R' \setminus A$, alors on a $\alpha' = \alpha$ (lemme 2.3.8). Sinon, par le lemme 2.4.3, il existe un créateur privilégié R' de R par A et, par le lemme ci-dessus, on a $\alpha' <^+ \alpha$. $\square$

Le principe d'une définition complète des radicaux créateurs par delà les dérivations pourrait être la relation de contribution entre les noms de radicaux. Toutefois, les créateurs privilégiés suffisent aux preuves de la suite de cette thèse, je m'en tiendrai donc à eux.

2.5 Caractérisations à l'aide des étiquettes complètes

Les étiquettes complètes contiennent énormément d'informations. Il est en effet possible de reconnaître un radical résidu à son nom ou un héritier à son étiquette de tête.

2.5.1 Familles de radicaux résidus

On commence par généraliser les notations $<^+$ et $<^*$ au cas ou le second terme n'est pas nécessairement un nom de radical. Pour tout nom de radical α et tout terme étiqueté M on pose :

$\alpha <^+ M \quad \Leftrightarrow \quad \exists u \in \mathcal{O}(M),\ \tau(M/u) = l_1.<\gamma,a>.l_2,\ \text{avec } \alpha <^* \gamma$
$\alpha <^* M \quad \Leftrightarrow \quad \alpha <^+ M$ ou il existe un radical de nom α dans M

Dans le cas où l'on a $\alpha <^+ M$, on dit que le nom α contribue au terme M. Il faut noter ici que les relations étendues $<^+$ et $<^*$ n'ont plus rien de transitif ni de réflexif.

Compte tenu du mécanisme d'étiquetage des règles de réécriture, il est clair que la contraction d'un radical de nom α à une certaine étape d'une dérivation A: $M \rightarrow^* N$ peut entraîner une relation de contribution $\alpha <^+ N$. Le lemme suivant analyse le point de vue réciproque.

Lemme 2.5.1 *Soit une dérivation A: $M \rightarrow^* N$. Soit une occurrence u du terme N, telle qu'il existe une étiquette élémentaire $<\alpha,a>$, ainsi que deux étiquettes éventuellement vides l_1 et l_2, avec $\tau(N/u) = l_1.<\alpha,a>.l_2$. Alors, une des propositions suivantes au moins est vérifiée :*

1. *Il existe dans le terme M un radical de nom γ, tel que l'on a la relation $\gamma <^* \alpha$.*
2. *Il existe dans le terme M un sous-terme M/v, tel que l'on a l'égalité $\tau(M/v) = m_1.<\alpha,a>.m_2$.*

Démonstration: La preuve est par induction sur la longueur de la dérivation A. Si A est la dérivation vide O, alors le cas *2* s'applique toujours. Sinon, on décompose la dérivation A comme $A = BC$, avec B: $M \rightarrow^* P$ et C: $P \rightarrow N$. Par le lemme 2.2.3 appliqué à la réduction C, il y a deux possibilités.

1. Il existe dans le terme P un radical de nom α, alors, par le lemme 2.4.10, il existe dans le terme M un radical de nom γ, avec $\gamma <^* \alpha$.
2. Il existe dans le terme P une occurrence w, ainsi que deux étiquettes éventuellement vides n_1 et n_2, telles que l'on a $\tau(P/w) = n_1.<\alpha,a>.n_2$. Il reste à conclure par application de l'hypothèse d'induction à la dérivation B. □

Comme dans le cas des étiquettes simples, on impose quelques contraintes sur les étiquetages, afin d'éviter les “accidents syntaxiques” qui feraient perdre toute signification intuitive aux noms des radicaux. Pour commencer, on évite de mélanger dans un même terme radicaux et étiquettes qu'ils peuvent contribuer à former.

Définition 2.5.2 (Cohérence) *Un terme M est cohérent, si et seulement si il n'existe pas de radical de nom α dans M tel que le nom α contribue au terme M.*

La contrainte de cohérence permet d'identifier les radicaux qui sont des résidus à partir de leur noms uniquement.

Lemme 2.5.3 *Soit une dérivation A: $M \rightarrow^* N$, telle que le terme de départ M est cohérent. Alors, pour toute famille complète $\overline{\mathcal{F}}$ de nom α dans M, l'ensemble d'occurrences $\overline{\mathcal{F}} \setminus A$ est exactement la famille complète des radicaux de nom α dans N.*

Démonstration : Par le lemme 2.3.8. les sous-termes dont les occurrences sont dans $\overline{\mathcal{F}} \setminus A$ sont des radicaux de nom α. Réciproquement, soit un radical R de nom α dans N, alors le radical R ne peut pas être créé par la dérivation A. Si tel était le cas, alors, par le lemme 2.4.10, il existerait dans M un radical R' dont le nom α' contribuerait à α. Une telle situation est exclue car elle viole la cohérence du terme M (les étiquettes du nom α sont aussi des étiquettes du terme M). Le radical R est donc un résidu d'un radical R' de M. Le nom de R' ne peut être que α, c'est à dire que le radical R' est membre de la famille complète $\overline{\mathcal{F}}$. $\square$

La condition de cohérence est un peu forte pour obtenir cette caractérisation des radicaux résidus. De fait, la condition "il n'existe pas dans M deux radicaux de noms α_1 et α_2, tels que l'on ait $\alpha_1 <^+ \alpha_2$" serait suffisante. Mais, comme nous le verrons à la fin de ce chapitre, cette condition doit être renforcée pour pouvoir survivre aux dérivations.

2.5.2 Héritiers des familles de radicaux

Lemme 2.5.4 *Soit une dérivation A: $M \to^* N$. Soit un radical de nom α et d'occurrence u dans le terme M et soit une occurrence v dans $u \backslash\!\backslash\!\backslash A$. Alors, il existe une règle de réécriture $\gamma \to \delta$, ainsi que deux étiquettes éventuellement vides l_1 et l_2, telles que l'on a $\tau(N/v) = l_1.{<}\gamma{,}a_\epsilon{>}.l_2$, avec $\gamma <^* \alpha$ et $\tau(\delta) = {<}\gamma{,}a_\epsilon{>}$.*

Démonstration: Dans le cas où A est la dérivation vide, alors il n'y a pas d'héritiers par A. Sinon, on décompose la dérivation A en $A = BC$, avec B: $M \to^* P$ et C: $P \to N$. D'après la définition des héritiers, il existe une occurrence w dans le terme P, telle que l'on se trouve dans l'un des trois cas suivants :

1. On a $w \in u \setminus B$ et $v \in w \backslash\!\backslash\!\backslash C$. Alors, par le lemme 2.3.8, le sous-terme P/w est un radical de nom α. Ensuite, l'étiquette $\tau(N/v)$ est de la forme $\tau(N/v) = l_1.{<}\alpha{,}a_\epsilon{>}.l_2$, par le lemme 2.2.2 d'étiquetage des héritiers. On a donc ici $\gamma = \alpha$.
2. On a $w \in u \backslash\!\backslash\!\backslash B$ et $v \in w \setminus C$. Par induction, l'étiquette qui décore le sous-terme P/w est de la forme $\tau(P/w) = m_1.{<}\gamma{,}a_\epsilon{>}.m_2$, avec $\alpha <^* \gamma$. Ensuite, par le lemme 2.2.1 d'étiquetage des résidus, il existe une étiquette éventuellement vide n_1, avec $\tau(N/v) = n_1.\tau(P/w)$.
3. On a $w \in u \backslash\!\backslash\!\backslash B$ et $v \in w \backslash\!\backslash\!\backslash C$. Alors, comme dans le cas précédent, on a l'égalité $\tau(P/w) = m_1.{<}\gamma{,}a_\epsilon{>}.m_2$, avec $\alpha <^* \gamma$. Il suffit ensuite de reprendre les deux cas du lemme 2.5.4 d'étiquetage des héritiers.

    (a) Le sous-terme P/w n'est pas un radical mais l'occurrence w aboutit à l'intérieur d'un radical de nom α' et d'occurrence w' dans P, avec $w' \in \mathcal{R}(C)$ et $v \in w' \backslash\!\backslash\!\backslash C$. Alors, l'étiquette de tête du sous-terme N/v s'écrit $\tau(N/v) = l_1.{<}\alpha'{,}a'_\epsilon{>}.l_2$, avec $\alpha <^* \gamma < \alpha'$.

    (b) Le sous-terme P/w est lui même un radical contracté de nom α'. Alors, il existe une étiquette éventuellement vide l', telle que l'on a $\tau(P/w) = l'.\tau(\alpha)$ et deux étiquettes éventuellement vides n_1 et n_2, telles que l'on a $\tau(N/v) = n_1.l'.{<}\alpha'{,}a'_\epsilon{>}.n_2$. Il y a ensuite deux possibilités selon que l'étiquette m_2 est vide ou non. Dans le premier cas, on a $\tau(\alpha') = {<}\gamma{,}a_\epsilon{>}$, soit finalement, $\alpha <^* \gamma < \alpha'$. Dans le second cas, on a $l' = l'_1.{<}\gamma{,}a_\epsilon{>}.l'_2$, soit directement, $\tau(N/v) = l_1.{<}\gamma{,}a_\epsilon{>}.l_2$ en posant $l_1 = n_1.l'_1$ et $l_2 = l'_2.{<}\alpha'{,}a'_\epsilon{>}.n_2$. $\square$

Les contraintes à imposer pour obtenir la réciproque de ce lemme sont assez draconiennes

Définition 2.5.5 (Cohérence forte) *Soit Σ^l un système de réécriture complètement étiqueté. Par définition, le système Σ^l est fortement cohérent, si et seulement si les étiquettes atomiques employées pour étiqueter les membres droits des règles de Σ^l sont toutes distinctes. Une dérivation A: $M \to^* N$ dans un système fortement cohérent est fortement cohérente, si et seulement si toutes les étiquettes de son terme de départ M sont atomiques et deux à deux distinctes.*

Dans un système fortement cohérent, on notera habituellement a_α l'étiquette atomique qui décore la racine du membre droit β de la règle $\alpha \to \beta$ (i.e., on note $\tau(\beta) = <\alpha,a_\alpha>$). Cette notation fait bien apparaître que la connaissance de l'étiquette a_α suffit pour retrouver la règle de réécriture $\alpha \to \beta$. Par ailleurs, un terme M dont toutes les étiquettes sont atomiques est toujours cohérent au sens de la définition 2.5.2.

Du point de vue de l'intuition, les restrictions de cohérence forte correspondent aux conditions d'un vrai système de réécriture de termes : il n'y a aucun partage, ni dans le terme initial des dérivations, ni dans les membres droits de règles. Il est difficile d'imaginer plus strict tout en restant significatif.

Définition 2.5.6 (Contribution forte) *Soient deux noms de radicaux α et γ. Par définition, le nom α contribue fortement au nom γ et on note $\alpha \ll \gamma$, si et seulement si il existe dans le nom γ un sous-terme γ/u dont l'étiquette de tête est de la forme $\tau(\gamma/u) = l_1.<\alpha,a_\alpha>.l_2$, où l_1 et l_2 sont des étiquettes éventuellement vides. On note $\ll^*$ la fermeture réflexive et transitive de la relation $\ll$.*

On voit facilement que, dans le lemme 2.5.4, on peut remplacer la relation de contribution $<$ par la relation de contribution forte $\ll$. Cela revient à remplacer a_ϵ par le a_α adapté partout dans la démonstration. Reste à prouver le lemme réciproque.

Lemme 2.5.7 *Soit une dérivation fortement cohérente A: $M \to^* N$ et soit un sous-terme N/u, dont l'étiquette de tête est de la forme $\tau(N/u) = l_1.<\alpha,a_\alpha>.l_2$, où l_1 et l_2 sont des étiquettes éventuellement vide, tandis que le nom α est tel qu'il existe, dans le terme M, un radical d'occurrence v et de nom γ, avec $\gamma \ll^* \alpha$. Alors, l'occurrence u est un héritier de l'occurrence v par la dérivation A.*

Démonstration: La preuve est par induction sur la longueur de la dérivation A. Si la dérivation A est la dérivation vide O, alors le résultat est immédiat, car il n'y a alors, dans le terme final N, ni héritiers, ni étiquettes composites. Sinon, la dérivation A est de la forme $A = BC$, avec B: $M \to^* P$ et C: $P \to N$. D'après le lemme 2.2.3, il existe une occurrence w dans le terme P, telle que l'on se trouve dans l'un des deux cas suivants :

1. Le sous-terme P/w est un radical de nom α, il est contracté par la réduction C, et on a $u \in w\backslash\backslash\backslash C$. L'hypothèse $\gamma \ll^* \alpha$ donne lieu à deux possibilités :

   (a) Si les noms de radicaux α et γ sont égaux, alors, par le lemme 2.5.3, qui s'applique bien ici au terme M cohérent, on a $w \in v \setminus B$ (le radical M/v est ici seul dans sa famille). Soit finalement, $u \in v\backslash\backslash\backslash A$.

   (b) Sinon, par définition de $\ll^+$, il existe une occurrence w' dans $\overline{\mathcal{O}}(\alpha)$, avec $\tau(\alpha/w') = l'_1.<\alpha',a_{\alpha'}>.l'_2$ et $\gamma \ll^* \alpha'$. Les variables des noms de radicaux ne portant pas d'étiquette, l'occurrence w' est nécessairement dure dans α. C'est à dire qu'il existe dans le terme P, un sous-terme $P/w.w'$ avec $\tau(M/w.w') = l'_1.<\alpha',a_{\alpha'}>.l'_2$. Par hypothèse d'induction, on a $w.w' \in v\backslash\backslash\backslash B$. Par ailleurs, puisque l'occurrence w' est dure dans le nom α du radical P/w contracté par la réduction C, on a $u \in w.w'\backslash\backslash\backslash C$ par définition des héritiers. Soit finalement, $u \in v\backslash\backslash\backslash A$.

2. Le sous-terme P/w est décoré par une étiquette de la forme $\tau(P/w) = l'_1.<\alpha,a_\alpha>.l'_2$ et on a $u \in w\backslash\backslash A$. Alors, par induction, on a $w \in v\backslash\backslash\backslash B$ et donc, $u \in v\backslash\backslash\backslash A$. □

Ce lemme, allié à la version précisée pour la relation $\ll^*$ du lemme 2.5.4 donne une caractérisation des héritiers par les étiquettes. C'est à dire que, compte tenu de certaines contraintes sur les membres droits de règles et sur le terme initial d'une dérivation $A\colon M {\to^*} N$, on peut retrouver les héritiers dans le terme N par la dérivation A d'un radical donné du terme M, à l'aide des étiquettes de N uniquement. En particulier, si on considère deux dérivations fortement cohérentes $A\colon M {\to^*} N$ et $B\colon M {\to^*} N$ qui ont même point de départ et même point d'arrivée, alors, pour tout radical R d'occurrence u dans le terme M, on a l'égalité $u \backslash\!\!\backslash\!\!\backslash A = u \backslash\!\!\backslash\!\!\backslash B$.

2.6 Dérivations complètes

Quand on identifie les radicaux par leurs noms, on ne peut plus considérer les radicaux individuellement mais seulement les familles complètes de radicaux. Les dérivations complètes sont les dérivations naturellement issues de cette restriction.

Définition 2.6.1 *Soit une réduction $A\colon M \xrightarrow{U} N$. Par définition, la réduction A est complète, si et seulement si l'ensemble U ne contient que des familles complètes de radicaux. Autrement dit, on peut donner la partition de l'ensemble U par familles complètes :*

$$U = \overline{\mathcal{F}_1} \cup \overline{\mathcal{F}_2} \cup \ldots \cup \overline{\mathcal{F}_n}$$

Une dérivation est complète, si et seulement si toutes ses étapes élémentaires sont des réductions complètes. Dérivations complètes et termes cohérents sont compatibles :

Lemme 2.6.2 *Si $A\colon M {\to^*} N$ est une dérivation complète telle que le terme M est cohérent, alors le terme N est cohérent.*

Démonstration: Il suffit de montrer ce lemme pour les réductions. Supposons qu'il existe dans le terme N, un radical R de nom α qui contribue au terme N. Par définition, il existe alors un sous-terme N/u, dont l'étiquette de tête est de la forme $\tau(N/u) = l_1.<\gamma,a>.l_2$, avec $\alpha <^* \gamma$. On se pose ensuite la question de l'origine de l'élément $<\gamma,a>$. Selon le lemme 2.2.3, il n'y a que deux possibilités :

1. L'élément $<\gamma,a>$ est introduit dans l'étiquette $\tau(N/v)$ par une règle (red), parce qu'un radical de nom γ est contracté par la réduction A. A fortiori, il existe dans le terme M, un radical dont le nom est γ. Il y a ensuite deux sous-cas, selon la nature du radical R :
   (a) Si le radical R est un résidu, alors il existe un radical de nom α dans le terme M qui n'est pas contracté par la réduction A. Or, l'égalité $\gamma = \alpha$ est exclue, puisque la réduction A est complète. Par ailleurs, la relation $\alpha <^+ \gamma$ est elle aussi impossible de par la cohérence du terme M.
   (b) Si le radical R est créé par la réduction A, alors, par les lemme 2.4.3 et 2.4.5, il existe dans le terme M un radical de nom α', avec $\alpha' < \alpha$. On a donc la relation $\alpha' <^+ \gamma$ entre deux noms de radicaux du terme M, ce qui contredit directement la cohérence de M.
2. L'élément $<\gamma,a>$ est déjà présent dans une étiquette du terme M, c'est à dire qu'il existe une occurrence u' dans M, avec $\tau(M/u') = l'_1.<\gamma,a>.l'_2$. On distingue ensuite les mêmes deux sous-cas que dans 1. ci-dessus :
   (a) Si le radical R est un résidu, alors il existe un radical de nom α dans M. Dès lors, la relation $\alpha <^* \gamma$ et l'égalité $\tau(M/u') = l'_1.<\gamma,a>.l'_2$ contredisent la cohérence de M.
   (b) Si le radical R est créé par la réduction A, alors il existe un radical R' de nom α' dans M, avec $\alpha' < \alpha <^* \gamma$. Soit, une dernière contradiction avec la cohérence de M. □

Les dérivations complètes issues d'un terme cohérent ont un très bon comportement. Les noms des différentes familles de radicaux qu'elles contractent à chaque étape élémentaire sont tous deux à deux distincts. Plus précisément, si on considère une telle dérivation :

$$A\colon M_0 \xrightarrow{U_1} M_1 \xrightarrow{U_2} \ldots \xrightarrow{U_n} M_n$$

Soit ensuite le nom α d'une famille de radicaux contractée à la i-ème étape A_i de la dérivation A. Alors, l'ensemble de radicaux contractés U_i est le seul à contenir une famille de nom α. Du point de vue des termes intermédiaires, il existe un entier j compris entre 0 et $i-1$, qui marque la première apparition d'une famille de nom α dans le terme intermédiaire M_j. Enfin, quelque soit l'entier k de l'intervalle $[0 \ldots n]$, il existe au moins un radical de nom α dans le terme M_k, si et seulement si on a les inégalités $j \leq k < i$.

Chapitre 3

Théorèmes fondamentaux des systèmes orthogonaux

Ce chapitre explique en détail des deux théorèmes fondamentaux des systèmes orthogonaux. À savoir, le lemme des déplacements parallèles, qui donne la confluence des dérivations, et le théorème des développements finis, qui permet de raisonner par induction sur les dérivations. L'emploi des étiquettes va permettre des preuves particulièrement précises. La technique de base de ces preuves est le réétiquetage : pour prouver une certaine propriété, dont le libellé ne dépend pas d'un étiquetage particulier, on choisit un étiquetage spécialement étudié pour en faciliter la preuve. Un fois la preuve faite, on peut oublier cet étiquetage purement technique.

3.1 Réétiquetage des dérivations

Tous les concepts liés aux occurrences, comme les résidus ou les radicaux contractés ne dépendent pas des étiquettes. C'est à dire que si un résultat est acquis en termes d'occurrences ou de résidus, alors ce résultat est indépendant du choix d'un étiquetage ou d'un autre et s'applique même à la réécriture non-étiquetée.

3.1.1 Effacement

Pour transformer un terme M de $\mathcal{T}^g$ (termes simplement étiquetés) en un terme de $\mathcal{T}$ (termes sans étiquettes), il suffit d'effacer les étiquettes de M à l'aide de la *fonction d'oubli* F :

$$F(x^l) = x \qquad F(f^l(M_1, M_2, \ldots, M_a)) = f(F(M_1), F(M_2), \ldots, F(M_a))$$

Soit un terme M et un terme simplement étiqueté M'. Si on a $F(M') = M$, on dit que M' est un *étiquetage simple* de M. Il est immédiat que les ensemble des occurrences de M et M' sont identiques.

Soit ensuite un ensemble de règles de réécriture non-étiquetées $\Sigma = \{\ldots, \alpha \to \beta, \ldots\}$. Soit Σ^g un ensemble de règles simplement étiquetées construit a partir de Σ (section 1.2.3). Les réductions parallèles ont été introduites dans le cadre du système étiqueté Σ^g, afin de modéliser les réductions de graphes. Mais il est clair qu'on peut définir directement la relation de réduction parallèle engendrée par les règles de Σ. Pour ce faire, il suffit d'enlever toutes les étiquettes dans les définitions de la section 1.2.3. Ainsi, on définit la relation de réécriture parallèle sur $\mathcal{T}$ à l'aide de l'axiome et des règles d'inférence suivants :

$$x \to x \text{ (id)} \qquad \frac{M_i \to N_i \text{ si } i \in \mathrm{dom}(f) \quad M_i = N_i \text{ si } i \notin \mathrm{dom}(f)}{f(M_1, M_2, \ldots, M_a) \to f(N_1, N_2, \ldots, N_a)} \text{ (rec}f\text{)}$$

$$\frac{\sigma(x) \to \rho(x) \text{ si } x \in \mathcal{V}_a(\alpha) \quad \sigma(x) = \rho(x) \text{ si } x \in \mathcal{V}_i(\alpha)}{\sigma(\alpha) \to \rho(\beta)} \text{ (red)}$$

Soit donc une réduction parallèle non-étiquetée A: $M \to N$. On définit sans problème les résidus (définition 1.2.7) , les descendants (définition 1.2.12) et les héritiers (définition 1.2.13) par la réduction A. La définition 1.2.5 de l'ensemble $\mathcal{R}(A)$ des radicaux contractés par A s'applique également ici. De fait, les réductions parallèles selon les règles de Σ et Σ^g se correspondent très exactement.

Lemme 3.1.1 *Soient M un terme non-étiqueté, et M' un étiquetage simple de M.*

1. *Quelle que soit l'occurrence u dans M (et dans M'), le sous-terme M/u est un radical pour le système Σ si et seulement si le sous-terme M'/u est un radical pour le système simplement étiqueté Σ^g.*
2. *Quel que soit l'ensemble U d'occurrences de radicaux dans M (et donc dans M'), il existe deux réductions A: $M \xrightarrow{U} N$ et A': $M' \xrightarrow{U} N'$ et N' est un étiquetage de N.*

$$\begin{tikzcd} M' \arrow[r, "F"] \arrow[d, "U"'] & M \arrow[d, "U"] \\ N' \arrow[r, "F"] & N \end{tikzcd}$$

3. *En outre, quelle que soit l'occurrence v de M (et de M') on a les égalités suivantes :*

$$v \setminus A = v \setminus A' \quad v \backslash\!\backslash A = v \backslash\!\backslash A' \quad v \backslash\!\backslash\!\backslash A = v \backslash\!\backslash\!\backslash A'$$

Démonstration : La proposition *1* ci-dessus est une conséquence immédiate des définition et du procédé d'étiquetage de Σ en Σ^g qui décore les membres gauches de règles avec *toutes* les étiquettes possibles. Plus précisément, on veut prouver que tout sous-terme M'/u est de la forme $M'/u = \sigma'(\alpha')$, où α' est le membre gauche d'une règle étiquetée, si et seulement si le sous-terme $M/u = F(M'/u)$ est de la forme $M/u = \sigma(\alpha)$, ou α est le membre gauche d'une règle non-étiquetée, avec $F(\alpha') = \alpha$, tandis que les substitutions σ et σ' satisfont la condition $\forall x \in \mathcal{V}(\alpha),\ F(\sigma'(x)) = \sigma(x)$. L'équivalence proposée se démontre facilement par induction sur le terme α, il faut tout même noter qu'elle n'est valable que dans le cas de membres gauches linéaires. Les deux proposition restantes *2* et *3* sont immédiates, une fois acquise la proposition *1*. □

3.1.2 Simplification

On peut étiqueter les termes de $\mathcal{T}$ de façon simple, pour obtenir l'ensemble $\mathcal{T}^g$, ou complète, pour obtenir l'ensemble $\mathcal{T}^l$. On définit le morphisme de *simplification* S, comme une fonction de $\mathcal{T}^l$ vers $\mathcal{T}^g$.

$$S(x^l) = x^{s(l)} \qquad S(f^l(M_1, M_2, \ldots, M_a)) = f^{s(l)}(S(M_1), S(M_2), \ldots, S(M_a))$$

Où s est la fonction de simplification des étiquettes, définie par :

$$\left\{\begin{array}{rcl} s(a) & = & a \\ s(l.e) & = & s(e) \\ s(<\alpha,a>) & = & s(\tau(\alpha)).a \end{array}\right.$$

Si on a deux termes étiquetés M' et M'' avec $M' = S(M'')$, on dira que M' est un *réétiquetage simple* de M' et M'' un *réétiquetage complet* de M'.

Considérons maintenant un système de réécriture non-étiqueté Σ. On peut étiqueter Σ d'abord avec des étiquettes simples, puis avec des étiquettes complètes, pour obtenir deux systèmes étiquetés Σ^g et Σ^l. On a bien entendu un lemme d'isomorphisme des réductions dans Σ^g et Σ^l, en tout point semblable à celui qui porte sur Σ et Σ^g.

Lemme 3.1.2 *Soient M' un terme simplement étiqueté, et M'' un terme complètement étiqueté, tels que l'on a $S(M'') = M'$.*

1. *Quelle que soit l'occurrence u dans M' (et dans M''), le sous-terme M/u est un radical pour le système simplement étiqueté Σ^g, si et seulement si le sous-terme M'/u est un radical pour le système complètement étiqueté Σ^l.*
2. *Quel que soit l'ensemble U d'occurrences de radicaux dans M' (et donc dans M''), il existe deux réductions A': $M' \xrightarrow{U} N'$ et A'': $M'' \xrightarrow{U} N''$ avec :*

$$\begin{tikzcd} M'' \arrow[r, "S"] \arrow[d, "U"'] & M' \arrow[d, "U"'] \\ N'' \arrow[r, "S"] & N' \end{tikzcd}$$

3. *En outre, quelle que soit l'occurrence v de M' (et de M''), on a les égalités suivantes :*

$$v \setminus A' = v \setminus A'' \quad v \backslash\!\backslash A' = v \backslash\!\backslash A'' \quad v \backslash\!\backslash\!\backslash A' = v \backslash\!\backslash\!\backslash A''$$

Démonstration : Le plus simple est de considérer un morphisme d'effacement qui opère sur les termes complètement étiquetés. Ce morphisme est la composition $\Phi = F \circ S$. Par la complétude de l'étiquetage des règles avec des étiquettes complètes et puisque les noms de radicaux sont des termes linéaires, on montre facilement que le sous-terme M''/u est de la forme $M''/u = l.\sigma''(\alpha'')$, ou l est une étiquette complète éventuellement vide et α'' est le membre gauche d'une règle complètement étiquetée, si et seulement si le sous-terme $M/u = \Phi(M''/u)$ est de la forme $M/u = \sigma(\alpha)$, où $\alpha = \Phi(\alpha'')$ est le membre gauche d'une règle non-étiquetée. Or, on a $\Phi(M''/u) = F(M'/u) = M/u$ et on sait déjà (lemme 3.1.1) que le terme non-étiqueté M/u est un radical, si et seulement si le terme simplement étiqueté M'/u est un radical. □

3.1.3 Autres isomorphismes de dérivations

Comme on vient de le voir, on peut effacer les étiquettes complètes en composant les morphismes F et S :

$$\begin{tikzcd} M'' \arrow[r, "S"] \arrow[d, "U"'] & M' \arrow[r, "F"] \arrow[d, "U"'] & M \arrow[d, "U"'] \\ N'' \arrow[r, "S"] & N' \arrow[r, "F"] & N \end{tikzcd}$$

On peut aussi changer les étiquettes simples ou complètes, on parle alors de réétiquetage. Voici par exemple un réétiquetage dans les réductions simplement étiquetées :

$$\begin{tikzcd}
M_1' \arrow[r, "F"] \arrow[d, "U"'] & M \arrow[d, "U"'] & M_2' \arrow[l, "F"'] \arrow[d, "U"'] \\
N_1' \arrow[r, "F"] & N & N_2' \arrow[l, "F"']
\end{tikzcd}$$

Ainsi qu'un réétiquetage dans les réductions complètement étiquetées :

$$\begin{tikzcd}
M_1'' \arrow[r, "S"] \arrow[d, "U"'] & M_1' \arrow[r, "F"] \arrow[d, "U"'] & M \arrow[d, "U"'] & M_2' \arrow[l, "F"'] \arrow[d, "U"'] & M_2'' \arrow[l, "S"'] \arrow[d, "U"'] \\
N_1'' \arrow[r, "S"] & N_1' \arrow[r, "F"] & N & N_2' \arrow[l, "F"'] & N_2'' \arrow[l, "S"']
\end{tikzcd}$$

Il va également de soi que tous les résultats obtenus sur les réductions s'étendent aux dérivations. Par exemple, on peut effacer les étiquettes simples d'une dérivation étiquetée (ou étiqueter simplement une dérivation non-étiquetée).

$$\begin{tikzcd}
M' \arrow[r, "F"] \arrow[d, "U"'] & M \arrow[d, "U"'] \\
N' \arrow[r, "F"] \arrow[d, "*"] & N \arrow[d, "*"] \\
P' \arrow[r, "F"] & P
\end{tikzcd}$$

Il faut tout de même être prudent avec les procédés de réétiquetage. Par exemple, la confluence des systèmes orthogonaux non-étiquetés (qui est un résultat classique de la théorie de la réécriture [Klo 80, Hue 80]), n'entraîne pas la confluence des systèmes étiquetés. En effet, étant données deux dérivations (simplement) étiquetées A': $M' \rightarrow^* N'$ et B': $M' \rightarrow^* P'$, alors par le lemme 3.1.1, il existe deux dérivations non-étiquetées A: $M \rightarrow^* N$ et B: $M \rightarrow^* P$ qui leurs sont isomorphes par F. On suppose prouvée la confluence du système non-étiqueté, il existe donc deux dérivations C: $N \rightarrow^* Q$ et D: $P \rightarrow^* Q$. Soient ensuite les deux dérivations C': $N' \rightarrow^* Q_1'$ et D': $P' \rightarrow^* Q_2'$ qui sont isomorphes par F aux dérivations C et D respectivement. Il est clair que l'on a pas nécessairement $Q_1' = Q_2'$, mais seulement $F(Q_1') = F(Q_2')$.

3.2 Déplacements parallèles

Le lemme des déplacement parallèle est la propriété fondamentale des systèmes orthogonaux. Il traduit le déterminisme des calculs qui vient de l'indépendance de la contraction de radicaux différents.

3.2.1 Étude fine de la réduction résidu

Le lemme de substitution est une étape importante de la preuve de la confluence par la méthode des réductions parallèles ou méthode de Tait et Martin-Löf (voir par exemple [Hin 86, page 315] pour le cas plus complexe du λ-calcul)

Lemme 3.2.1 (Lemme de substitution) *Soit un terme γ. Soient σ et ρ deux substitutions. On a l'implication*

$$\left\{\begin{array}{l} \forall x \in \mathcal{V}_a(\gamma),\ A_x\colon \sigma(x) \to \rho(x) \\ \forall x \in \mathcal{V}_i(\gamma),\ A_x\colon \sigma(x) = \rho(x) \end{array}\right. \quad \Rightarrow \quad A\colon \sigma(\gamma) \to \rho(\gamma)$$

En outre, on a l'égalité :

$$\mathcal{R}(A) = (\bigcup_{x\in\mathcal{V}_a(\gamma)} \{\, u_x \mid \gamma/u_x = x\,\}.\mathcal{R}(A_x))$$

Tandis que les résidus par A des occurrences de M se calculent par :

$$\left\{\begin{array}{rrcl} \forall u \in \overline{\mathcal{O}}(\gamma), & u \setminus A & = & \{u\} \\ \forall x \in \mathcal{V}(\gamma)\ \forall u_x \ \textit{avec}\ \gamma/u_x = x\ \forall u \in \mathcal{O}(\sigma(x)), & (u_x.u)\setminus A & = & u_x.(u \setminus A_x) \end{array}\right.$$

Démonstration: Par induction sur la taille de γ. Si γ se réduit à une variable sans étiquette, alors le résultat est immédiat. Si γ est une variable étiquetée x^l, alors on utilise la définition $\sigma(x^l) = l.\sigma(x)$ et le lemme 2.3.4. Sinon, γ est de la forme $f^l(\gamma_1, \gamma_2, \ldots, \gamma_a)$ et on pose $M = \sigma(\gamma)$ et $N = \rho(\gamma)$. Soit maintenant un indice i du domaine de f. Par définition des variables autorisées et interdites, on a $\mathcal{V}_a(\gamma_i) \subseteq \mathcal{V}_a(\gamma)$ et $\mathcal{V}_i(\gamma_i) \subseteq \mathcal{V}_i(\gamma)$. On peut donc appliquer le raisonnement par induction et prouver l'existence d'une réduction $A_i\colon M_i \to N_i$. Soit maintenant i un indice interdit à partir du symbole f. Par définition, toutes les variables de M_i sont interdites. Par conséquent, on a $M_i = N_i$, à partir des hypothèses et par définition de la substitution. On dispose maintenant des prémisses suffisantes pour prouver A par une règle de type (rec). Par ailleurs on a :

$$\begin{array}{rcll} \mathcal{R}(A) & = & (\bigcup_{i\in \mathrm{dom}(f)} i.\mathcal{R}(A_i)) & \text{définition de } \mathcal{R}(A) \\ & = & (\bigcup_{i\in \mathrm{dom}(f)} i.(\bigcup_{x\in\mathcal{V}_a(\gamma_i)}\{\, u_x \mid \gamma_i/u_x = x\,\}.\mathcal{R}(A_x)) & \text{induction} \\ & = & (\bigcup_{x\in\mathcal{V}_a(\gamma)}\{\, u_x \mid \gamma/u_x = x\,\}.\mathcal{R}(A_x)) & \text{car, } \mathcal{V}_a(\gamma) = (\bigcup_{i\in\mathrm{dom}(f)} \mathcal{V}_a(\gamma_i)) \end{array}$$

Soit ensuite u, une occurrences quelconque dans M. On distingue deux cas :

1. Si l'occurrence u aboutit à l'intérieur du préfixe γ, alors on a encore deux sous-cas :
   (a) Si $u = \epsilon$, alors, puisque A est de type (rec), on a $\epsilon \setminus A = \{\epsilon\}$.
   (b) Sinon, il existe un indice i, avec $u = i.u'$, où u' est une occurrence dure de γ_i. Par définition des résidus et hypothèse d'induction, on a les égalités successives :

$$u \setminus A = i.(u_i \setminus A_i) = i.\{u_i\} = \{u\}$$

2. Enfin, si u est de la forme $u_x.u'$, où u_x est l'occurrence d'une variable x de γ et u' une occurrence de $\sigma(x)$, alors il existe un indice i, tel que l'on a $u_x = i.u'_x$, où u'_x est une occurrence de x dans γ_i. On termine ensuite la preuve à l'aide de l'hypothèse d'induction :

$$u \setminus A = i.(u'_x.u' \setminus A_i) = i.(u'_x.(u' \setminus A_x)) = u_x.(u' \setminus A_x)$$

□

Dans le cas du λ-calcul, on utilise en fait une version plus générale du lemme de substitution, qui correspondrait ici à :

$$A\colon \gamma \to \delta \quad \text{and} \quad \begin{cases} \forall x \in \mathcal{V}_a(\gamma),\ A_x\colon \sigma(x) \to \rho(x) \\ \forall x \in \mathcal{V}_i(\gamma),\ A_x\colon \sigma(x) = \rho(x) \end{cases} \Rightarrow \ A'\colon \sigma(\gamma) \to \rho(\delta)$$

Cette version étendue du lemme 3.2.1 est inutile ici, je ne la prouverai donc pas. La réciproque du lemme 3.2.1 vaut pour les préfixes linéaires et préservés (par la définition 2.3.5, un préfixe γ d'un terme M est préservé par la réduction $A\colon M \to N$ quand aucune des occurrences de $\mathcal{R}(A)$ n'est une occurrence dure de γ)

Lemme 3.2.2 *Soit une réduction $A\colon M \to N$, et soit γ un préfixe linéaire de M, c'est à dire qu'il existe une substitution σ avec $M = \sigma(\gamma)$. Si le préfixe γ est préservé par la réduction A, alors on a les trois résultats suivants :*

1. *Il existe une substitution ρ, telle que l'on a $N = \rho(\gamma)$.*
2. *Pour toute variable x, autorisée dans γ, il existe une réduction $A_x\colon \sigma(x) \to \rho(x)$.*
3. *Pour toute variable x, interdite dans γ, on a l'identité $A_x\colon \sigma(x) = \rho(x)$*

En outre, on a les égalités :

$$\mathcal{R}(A) = (\bigcup_{x \in \mathcal{V}_a(\gamma)} \{\, u_x \mid \gamma/u_x = x \,\}.\mathcal{R}(A_x))$$

$$\begin{cases} \qquad\qquad \forall u \in \overline{\mathcal{O}}(\gamma), & u \setminus A & = & \{u\} \\ \forall x \in \mathcal{V}(\gamma)\ \forall u_x \text{ avec } \gamma/u_x = x\ \forall u \in \mathcal{O}(\sigma(x)), & (u_x.u) \setminus A & = & u_x.(u \setminus A_x) \end{cases}$$

Démonstration : Ce lemme est une version plus explicite du lemme 2.3.6, dont la preuve peut être facilement étendue pour prouver *1*, *2* et *3* en plus du fait que γ est un préfixe de N. □

L'arbre de preuve de la réduction résidu $A \setminus B$ se construit inductivement à partir des arbres de preuve des réductions A et B. Soient donc $A\colon M \to N$ et $B\colon M \to P$, deux réductions issues d'un même terme M. On a les cinq possibilités suivantes :

- (id) $\leftrightarrow$ (id) La dérivation $B \setminus A$ ne peut être que du type (id).
- (rec) $\leftrightarrow$ (rec) Soient les arbres de preuve des réductions A et B :

$$\frac{A_i\colon M_i \to N_i \text{ si } i \in \mathrm{dom}(f) \qquad A_i\colon M_i = N_i \text{ si } i \notin \mathrm{dom}(f)}{A\colon f^l(M_1, \ldots, M_a) \to f^l(N_1, \ldots, N_a)}\,\text{(rec)}$$

$$\frac{B_i\colon M_i \to P_i \text{ si } i \in \mathrm{dom}(f) \qquad B_i\colon M_i = P_i \text{ si } i \notin \mathrm{dom}(f)}{B\colon f^l(M_1, \ldots, M_a) \to f^l(P_1, \ldots, P_a)}\,\text{(rec)}$$

On construit inductivement les arbres de preuve des a réductions $B_1 \setminus A_1, B_2 \setminus A_2, \ldots, B_a \setminus A_a$. —Dans le cas d'un indice interdit, $i \notin \mathrm{dom}(f)$, cette construction se ramène au simple renommage d'une égalité.— Puis on fabrique l'arbre de preuve de $B \setminus A$ en ajoutant une règle (rec) à la racine :

$$\frac{B_i \setminus A_i\colon N_i \to Q_i \text{ si } i \in \mathrm{dom}(f) \qquad B_i \setminus A_i\colon N_i = Q_i \text{ si } i \notin \mathrm{dom}(f)}{B \setminus A\colon f^l(N_1, \ldots, N_a) \to f^l(Q_1, \ldots, Q_a)}\ \text{(rec)}$$

Soit $\mathcal{R}(B \setminus A)$ l'ensemble des radicaux contractés par la réduction ci-dessus —notée $B \setminus A$ par anticipation—. On prouve que $\mathcal{R}(B \setminus A)$ est bien l'ensemble des résidus de $\mathcal{R}(B)$ par la réduction A.

$$\begin{array}{rll} \mathcal{R}(B \setminus A) & = \bigcup_{i \in \mathrm{dom}(f)} i.\mathcal{R}(B_i \setminus A_i) & \text{définition de } \mathcal{R}(B \setminus A) \\ & = \bigcup_{i \in \mathrm{dom}(f)} i.(\mathcal{R}(B_i) \setminus A_i) & \text{induction} \\ & = (\bigcup_{i \in \mathrm{dom}(f)} (i.\mathcal{R}(B_i)) \setminus A) & \text{définition des résidus par } A \\ & = (\bigcup_{i \in \mathrm{dom}(f)} i.\mathcal{R}(B_i)) \setminus A & \text{définition 1.2.17} \\ & = \mathcal{R}(B) \setminus A & \text{définition de } \mathcal{R}(B) \end{array}$$

- (rec) $\leftrightarrow$ (red) Soient les arbres de preuve des réductions A et B :

$$\frac{A_i\colon M_i \to N_i \text{ si } i \in \mathrm{dom}(f) \qquad A_i\colon M_i = N_i \text{ si } i \notin \mathrm{dom}(f)}{A\colon f^{l.e}(M_1, \ldots, M_a) \to f^{l.e}(N_1, \ldots, N_a)}\ \text{(rec)}$$

$$\frac{B_x\colon \sigma(x) \to \rho(x) \text{ si } x \in \mathcal{V}_a(\alpha) \qquad B_x\colon \sigma(x) = \rho(x) \text{ si } x \in \mathcal{V}_i(\alpha)}{B\colon l.\sigma(\alpha) \to l.\rho(\beta)}\ \text{(red)}$$

Ici, e est l'étiquette de tête (élémentaire) du nom α et on a $M = f^{l.e}(M_1, \ldots, M_a) = l.\sigma(\alpha)$. C'est à dire que $l.\alpha$ est un préfixe linéaire de M. Par la condition de non-recouvrement des noms de de radicaux, le préfixe linéaire $l.\alpha$ est préservé par la réduction A. Par conséquent, le lemme 3.2.2 s'applique et la réduction A est de la forme $A\colon l.\sigma(\alpha) \to l.\phi(\alpha)$. En outre, pour toute variable x interdite dans α, on a l'identité $A_x\colon \sigma(x) = \phi(x)$; tandis que, pour toute variable autorisée x, on peut construire la réduction $A_x\colon \sigma(x) \to \phi(x)$. Pour chaque sous-terme autorisé $\phi(x)$ de $\phi(\alpha)$ et à partir des arbres des deux réductions A_x et B_x, on peut construire l'arbre de preuve de la réduction $B_x \setminus A_x\colon \phi(x) \to Q_x$. On se donne maintenant une nouvelle substitution ψ définie sur les variables de α par :

$$\forall x \in \mathcal{V}_a(\alpha),\ \psi(x) = Q_x \qquad \forall x \in \mathcal{V}_i(\alpha),\ \psi(x) = \phi(x).$$

On dispose maintenant des prémisses d'une nouvelle règle d'inférence (red) :

$$\frac{B_x \setminus A_x\colon \phi(x) \to \psi(x) \text{ si } x \in \mathcal{V}_a(\alpha) \qquad B_x \setminus A_x\colon \phi(x) = \psi(x) \text{ si } x \in \mathcal{V}_i(\alpha)}{B \setminus A\colon l.\phi(\alpha) \to l.\psi(\beta)}\ \text{(red)}$$

Reste à vérifier que $B \setminus A$ contracte bien les résidus de $\mathcal{R}(B)$ par A. Pour toute variable x de α, soit u_x son occurrence. Il vient :

$$\begin{array}{rll} \mathcal{R}(B \setminus A) & = (\bigcup_{x \in \mathcal{V}_a(\alpha)} u_x.\mathcal{R}(B_x \setminus A_x)) \cup \{\epsilon\} & \text{définition de } \mathcal{R}(B \setminus A) \\ & = (\bigcup_{x \in \mathcal{V}_a(\alpha)} u_x.(\mathcal{R}(B_x) \setminus A_x)) \cup \{\epsilon\} & \text{induction} \\ & = (\bigcup_{x \in \mathcal{V}_a(\alpha)} u_x.\mathcal{R}(B_x) \setminus A) \cup \{\epsilon\} \setminus A & \text{lemme 3.2.2 (réduction } A \\ & = ((\bigcup_{x \in \mathcal{V}_a(\alpha)} u_x.\mathcal{R}(B_x)) \cup \{\epsilon\}) \setminus A & \text{définition 1.2.17} \\ & = \mathcal{R}(B) \setminus A & \text{définition de } \mathcal{R}(B) \end{array}$$

- (red) $\leftrightarrow$ (rec) On a les arbres de preuve :

$$\dfrac{A_x\colon \sigma(x) \to \rho(x) \text{ si } x \in \mathcal{V}_a(\alpha) \qquad A_x\colon \sigma(x) = \rho(x) \text{ si } x \in \mathcal{V}_i(\alpha)}{A\colon l.\sigma(\alpha) \to l.\rho(\beta)}\ \text{(red)}$$

$$\dfrac{B_i\colon M_i \to P_i \text{ si } i \in \mathrm{dom}(f) \qquad B_i\colon M_i = P_i \text{ si } i \notin \mathrm{dom}(f)}{B\colon f^{l.e}(M_1, \dots, M_a) \to f^{l.e}(P_1, \dots, P_a)}\ \text{(rec)}$$

Ici encore, $l.\alpha$ est un préfixe linéaire de M et la réduction $B\colon M \to P$ préserve ce ce préfixe. On utilise donc le lemme 3.2.2 pour extraire les réductions et les égalités B_x de la réduction B.

$$\forall x \in \mathcal{V}_a(\alpha),\ B_x\colon \sigma(x) \to \phi(x) \quad \forall x \in \mathcal{V}_i(\alpha),\ B_x\colon \sigma(x) = \phi(x)$$

On invoque ensuite l'hypothèse d'induction pour prouver les réductions $B_x \setminus A_x$ et construire la substitution ψ, avec :

$$\forall x \in \mathcal{V}_a(\alpha),\ B_x \setminus A_x\colon \rho(x) \to \psi(x) \qquad \forall x \in \mathcal{V}_i(\alpha),\ B_x \setminus A_x\colon \rho(x) = \psi(x)$$

De par les condition d'orthogonalité $\mathcal{V}_a(\beta) \subseteq \mathcal{V}_a(\alpha)$ et $\mathcal{V}_i(\beta) \subseteq \mathcal{V}_i(\alpha)$, on dispose maintenant des hypothèses suffisantes pour appliquer les lemme 3.2.1 puis 2.3.4, afin de construire l'arbre de preuve de la réduction $B \setminus A\colon l.\rho(\beta) \to l.\psi(\beta)$. —À proprement parler, seule la condition $\mathcal{V}_i(\beta) \subseteq \mathcal{V}_i(\alpha)$ est nécessaire ici. Pour toute variable x autorisée dans β, on peut se contenter de la condition de base $x \in \mathcal{V}(\alpha)$, puisque l'égalité $B_x \setminus A_x\colon \psi(x) = \rho(x)$ est aussi une réduction vide.— On vérifie ensuite que $B \setminus A$ est bien la réduction résidu de B par A.

$$\begin{array}{rll}
\mathcal{R}(B \setminus A) & = \bigcup_{x \in \mathcal{V}_a(\beta)} \{\, u_x \mid \beta/u_x = x \,\}.\mathcal{R}(B_x \setminus A_x) & \text{lemme 3.2.1 (réduction } B \setminus A) \\
& = \bigcup_{x \in \mathcal{V}_a(\beta)} \{\, u_x \mid \beta/u_x = x \,\}.(\mathcal{R}(B_x) \setminus A_x) & \text{induction} \\
& = \bigcup_{x \in \mathcal{V}_a(\alpha)} \{\, u_x \mid \alpha/u_x = x \,\}.\mathcal{R}(B_x) \setminus A & \text{définition des résidus par } A \\
& = (\bigcup_{x \in \mathcal{V}_a(\alpha)} \{\, u_x \mid \alpha/u_x = x \,\}.\mathcal{R}(B_x)) \setminus A & \text{définition 1.2.17} \\
& = \mathcal{R}(B) \setminus A & \text{lemme 3.2.2 (réduction } B)
\end{array}$$

- (red) $\leftrightarrow$ (red) Comme deux radicaux distincts ne peuvent se recouvrir à la racine, les réductions A et B réduisent toutes deux le même radical $l.\sigma(\alpha)$. On pose les arbres de preuve :

$$\dfrac{A_x\colon \sigma(x) \to \rho(x) \text{ si } x \in \mathcal{V}_a(\alpha) \qquad A_x\colon \sigma(x) = \rho(x) \text{ si } x \in \mathcal{V}_i(\alpha)}{A\colon l.\sigma(\alpha) \to l.\rho(\beta)}\ \text{(red)}$$

$$\dfrac{B_x\colon \sigma(x) \to \phi(x) \text{ si } x \in \mathcal{V}_a(\alpha) \qquad B_x\colon \sigma(x) = \phi(x) \text{ si } x \in \mathcal{V}_i(\alpha)}{B\colon l.\sigma(\alpha) \to l.\phi(\beta)}\ \text{(red)}$$

Par induction, pour toute variable x autorisée dans α, on peut construire l'arbre de preuve de la réduction $B_x \setminus A_x\colon \rho(x) \to Q_x$. On définit ensuite une nouvelle substitution ψ sur les variables de α :

$$\forall x \in \mathcal{V}_a(\alpha),\ \psi(x) = Q_x \quad \forall x \in \mathcal{V}_i(\alpha),\ \psi(x) = \rho(x)$$

Les variables autorisées ou interdites dans β sont des variables de α de même nature. C'est à dire que, par les lemme 3.2.1 puis 2.3.4, on peut construire l'arbre de preuve de la réduction

$B \setminus A$: $l.\rho(\beta) \to l.\psi(\beta)$. —Notons qu'ici encore, on peut se contenter de la condition sur les variables interdites—. Enfin, on justifie l'appellation $B \setminus A$.

$$\begin{array}{rll}
\mathcal{R}(B \setminus A) & = \bigcup_{x \in \mathcal{V}_a(\beta)} \{ u_x \mid \beta/u_x = x \}.\mathcal{R}(B_x \setminus A_x) & \text{lemme 3.2.1 (réduction } B \setminus A) \\
& = \bigcup_{x \in \mathcal{V}_a(\beta)} \{ u_x \mid \beta/u_x = x \}.(\mathcal{R}(B_x) \setminus A_x) & \text{induction} \\
& = (\bigcup_{x \in \mathcal{V}_a(\alpha)} (\{ u_x \mid \alpha/u_x = x \}.\mathcal{R}(B_x)) \setminus A & \text{définition des résidus par } A \\
& = ((\bigcup_{x \in \mathcal{V}_a(\alpha)} \{ u_x \mid \alpha/u_x = x \}.\mathcal{R}(B_x)) \cup \{\epsilon\}) \setminus A & A \text{ contracte } \epsilon \\
& = \mathcal{R}(B) \setminus A & \text{définition de } \mathcal{R}(B)
\end{array}$$

3.2.2 Confluence forte des réductions parallèles

Pour toutes réductions A et B issues du même terme, le lemme des déplacements parallèles traduit la très forte similarité des deux dérivations permutées $A \sqcup B$ et $B \sqcup A$.

Lemme 3.2.3 (Déplacements parallèles) *Soit M un terme. Soient A: $M \to N$ et B: $M \to P$ deux réductions issues de M. Alors, on a les deux résultats suivants :*

1. *Les réductions $A \sqcup B$ et $B \sqcup A$ aboutissent au même terme Q.*
2. *Quel que soit l'ensemble U d'occurrences de M, on a $U \setminus A \sqcup B = U \setminus B \sqcup A$.*

Démonstration : La preuve de la proposition *1* est par induction sur les réductions A et B. En éliminant les symétries, on a les cas suivants :

- (id) $\leftrightarrow$ (id) Ce cas est facile. Toutes les réductions en jeu sont la même réduction de type (id), issue du terme x^l et aboutissant au terme x^l.
- (rec) $\leftrightarrow$ (rec) Ce cas se résout par une simple induction.
- (rec) $\leftrightarrow$ (red) Posons :

$$\frac{A_i\colon M_i \to N_i \text{ si } i \in \mathrm{dom}(f) \qquad A_i\colon M_i = N_i \text{ si } i \notin \mathrm{dom}(f)}{A\colon f^{l.e}(M_1, \ldots, M_a) \to f^{l.e}(N_1, \ldots, N_a)} \text{(rec)}$$

$$\frac{B_x\colon \sigma(x) \to \rho(x) \text{ si } x \in \mathcal{V}_a(\alpha) \qquad B_x\colon \sigma(x) = \rho(x) \text{ si } x \in \mathcal{V}_i(\alpha)}{B\colon l.\sigma(\alpha) \to l.\rho(\beta)} \text{(red)}$$

Comme on l'a déjà vu, il existe une substitution ϕ , telle que $N = l.\phi(\alpha)$. Il existe également (lemme 3.2.2) les réductions et les identités A_x, avec :

$$\forall x \in \mathcal{V}_a(\alpha),\ A_x\colon \sigma(x) \to \phi(x) \quad \forall x \in \mathcal{V}_i(\alpha),\ A_x\colon \sigma(x) = \phi(x)$$

La réduction $B \setminus A$ est une réduction de type (red) qui est issue du terme $N = l.\phi(\alpha)$:

$$\frac{B_x \setminus A_x\colon \phi(x) \to \psi(x) \text{ si } x \in \mathcal{V}_a(\alpha) \qquad B_x \setminus A_x\colon \phi(x) = \psi(x) \text{ si } x \in \mathcal{V}_i(\alpha)}{B \setminus A\colon l.\phi(\alpha) \to l.\psi(\beta)} \text{(red)}$$

Tandis que la réduction $A \setminus B$: $l.\rho(\beta) \to l.\psi'(\beta)$ est construite grâce au lemme 3.2.1, à partir des hypothèses :

$$\forall x \in \mathcal{V}_a(\beta),\ A_x \setminus B_x\colon \rho(x) \to \psi'(x) \quad \forall x \in \mathcal{V}_i(\beta),\ A_x \setminus B_x\colon \rho(x) = \psi'(x)$$

Pour toute variable x de $\mathcal{V}_a(\beta)$, on applique l'hypothèse d'induction aux dérivations $A_x \sqcup B_x$: $\sigma(x) \to^* \psi(x)$ et $B_x \sqcup A_x$: $\sigma(x) \to^* \psi'(x)$. Il vient $\psi(x) = \psi'(x)$. Pour toute variable x de $\mathcal{V}_i(\beta)$, on a aussi $\psi(x) = \psi'(x)$, par transitivité de l'égalité, à partir des hypothèses A_x: $\sigma(x) = \phi(x)$, $B_x \setminus A_x$: $\phi(x) = \psi(x)$, B_x: $\sigma(x) = \rho(x)$ et $A_x \setminus B_x$: $\rho(x) = \psi'(x)$. On a donc finalement $l.\psi(\beta) = l.\psi'(\beta)$.

- (red) $\leftrightarrow$ (red) Dans ce cas, les réductions $B \setminus A$ et $A \setminus B$ sont prouvées par le lemme 3.2.1. C'est à dire qu'il existe deux substitutions ψ et ψ', telles que :

$$\begin{array}{ll} \forall x \in \mathcal{V}_a(\alpha),\ B_x \setminus A_x\colon \rho(x) \to \psi(x) & \forall x \in \mathcal{V}_i(\alpha),\ B_x \setminus A_x\colon \rho(x) = \psi(x) \\ \forall x \in \mathcal{V}_a(\alpha),\ A_x \setminus B_x\colon \phi(x) \to \psi'(x) & \forall x \in \mathcal{V}_i(\alpha),\ A_x \setminus B_x\colon \phi(x) = \psi'(x) \end{array}$$

  Et que l'on a alors les hypothèses suffisantes pour appliquer les lemme 3.2.1 puis 2.3.4 :

$$B \setminus A\colon l.\rho(\beta) \to l.\psi(\beta) \quad A \setminus B\colon l.\phi(\beta) \to l.\psi'(\beta)$$

  On termine ensuite la preuve comme dans le cas précédent, en utilisant l'induction pour les variables autorisées et la transitivité de "=" pour les variables interdites.

Pour démontrer la proposition *2*, il suffit de réétiqueter M en un terme simplement étiqueté M', tel que M' est cohérent et que U représente un sous-terme partagé d'adresse l dans M'. Appelons A' et B' les réductions simplement étiquetés isomorphes à A et à B respectivement. Par le résultat de confluence *1* ci-dessus et le lemme 3.1.2, les dérivations $A' \sqcup B'$ et $B' \sqcup A'$ aboutissent au même terme Q'. D'après la proposition 1.3.4 les résidus de U par $A' \sqcup B'$ et $B' \sqcup A'$ sont exactement les sous-termes d'adresse l dans Q'. On a donc $U \setminus (A' \sqcup B') = U \setminus (B' \sqcup A')$. Soit, par le lemme 3.1.2, $U \setminus (A \sqcup B) = U \setminus (B \sqcup A)$. □

Ce lemme est bien plus fort qu'un simple résultat de confluence. En premier lieu, la confluence est une confluence forte dans le sens que le diagramme de confluence en "diamant" se referme en une seule étape. Par ailleurs, cette fermeture est le reflet de la non-ambiguïté des calculs dans les système orthogonaux. C'est à dire que la confluence provient ici de la contraction des même radicaux à des étapes différentes. Par contraste, la confluence des système à paires critiques [Hue 80] apparaît un peu comme un "miracle syntaxique".

La démonstration du lemme des déplacements parallèles utilise largement le deux premières conditions d'orthogonalité : absence de paires critique et linéarité des membres gauches. En revanche, ll faut remarquer que, si on a bien utilisé la condition d'orthogonalité $\mathcal{V}_i(\beta) \subseteq \mathcal{V}_i(\alpha)$ (1), on a pu se passer de la condition $\mathcal{V}_a(\beta) \subseteq \mathcal{V}_a(\alpha)$ (2). Et de fait, seuls importent ici les radicaux résidus, dont nous avons déjà vu que l'on prouve qu'ils sont des radicaux sans le secours de la condition (2) (cf. le lemme 2.3.7).

La technique du réétiquetage permet une preuve facile des résultats d'invariance des résidus, descendants et héritiers des familles de radicaux par les dérivations permutés.

Lemme 3.2.4 *Soient deux réductions A: $M \to N$ et B: $M \to P$ dans un système de réécriture complètement étiqueté Σ^l. Soit $\mathcal{F}$ une famille de radicaux de M. On a les égalités suivantes :*

$$\mathcal{F} \setminus (A \sqcup B) = \mathcal{F} \setminus (A \sqcup B) \tag{3.1}$$
$$\mathcal{F} \setminus\!\setminus (A \sqcup B) = \mathcal{F} \setminus\!\setminus (A \sqcup B) \tag{3.2}$$
$$\mathcal{F} \setminus\!\setminus\!\setminus (A \sqcup B) = \mathcal{F} \setminus\!\setminus\!\setminus (A \sqcup B) \tag{3.3}$$

Démonstration : La démonstration de ces propositions se fait à partir de la proposition *1* du lemme des déplacements parallèles, en utilisant des réétiquetages. Pour montrer 3.1, on réétiquette M en

M', tel que M', tel que M' est cohérent et $\mathcal{F}$ est une famille complète dans M' puis on applique la caractérisation des résidus de radicaux par les étiquettes complètes (lemme 2.3.8). Pour montrer 3.3, on réétiquette toutes les règles de Σ^l et le terme M en $\Sigma^{l'}$ et M', de sorte que les dérivations $A' \sqcup B'$ et $B' \sqcup A'$ soient fortement cohérentes (définition 2.5.5). Il reste alors, pour chaque occurrence de radical u de la famille $\mathcal{F}$, à employer la caractérisation des héritiers par les étiquettes complètes issue des lemmes 2.5.4 et 2.5.7 pour prouver l'égalité $u \backslash\!\backslash\!\backslash (A \sqcup B) = u \backslash\!\backslash\!\backslash (B \sqcup A)$ et donc, a fortiori, le résultat 3.3. Enfin, l'égalité 3.2 est une conséquence immédiate de 3.1 et 3.3, puisque l'on a, pour toute dérivation C, $\mathcal{F} \backslash\!\backslash C = (\mathcal{F} \setminus C) \cup (\mathcal{F} \backslash\!\backslash\!\backslash C)$. □

3.3 Développements finis

Le théorème des développements finis affirme que les dérivations qui satisfont certaines conditions sont toutes finies. Pour bien comprendre la nature de ce théorème, il n'est pas inutile de préciser ce que l'on entend par dérivation infinie. Soit un terme M. Une dérivation finie A: $M \rightarrow^* N$ de longueur n est la concaténation de n de réductions. Comme on l'a déjà vu, une attitude pragmatique est de caractériser la dérivation A par son terme initial M, ainsi que par une suite finie $(U_i)_{1 \leq i \leq n}$ d'ensembles d'occurrences de radicaux à contracter successivement. Une dérivation infinie est donc tout simplement définie par un terme initial M et par une suite infinie d'ensembles d'occurrences de radicaux à contracter successivement, que l'on peut noter $(U_i)_{1 \leq i}$. Ainsi, si on se donne deux règles de réécriture $K \rightarrow L$ et $L \rightarrow K$, alors la dérivation infinie issue du terme $M = F(L, K)$ qui contracte alternativement les deux radicaux possibles est exactement représentée par la suite $(U_i)_{1 \leq i}$ avec, pour tout entier $p > 1$, $U_{2p-1} = \{1\}$ et $U_{2p} = \{2\}$.

Une première remarque sur les dérivations infinies est qu'elles n'aboutissent jamais, c'est particulièrement net dans le cas de l'exemple déjà abordé :

$$A\colon F(K, L) \xrightarrow{\{1\}} F(L, L) \xrightarrow{\{2\}} F(L, K) \xrightarrow{\{1\}} F(K, L) \xrightarrow{\{2\}} \ldots$$

L'écriture A: $M \rightarrow^* N$ doit donc être réservée aux dérivations finies. Soit donc une dérivation (potentiellement) infinie que l'on note A: $M \rightarrow^* \ldots$ Une opération sensée sur la dérivation A est la décomposition $A = BC$, où B: $M \rightarrow^* N$ est une dérivation finie et C: $N \rightarrow^* \ldots$ est une dérivation (potentiellement) infinie.

3.3.1 Préordres, théorème de Kruskal

Dans un souci d'exhaustivité, je donne une présentation rapide du théorème de Kruskal (ou plutôt d'un de ses corollaires). Pour une présentation plus complète et pédagogique, on pourra consulter [Gal 89].

Définition 3.3.1 (Préordre) *Étant donné un ensemble $\mathcal{A}$, un préordre $\preceq$ est une relation binaire sur $\mathcal{A}$ qui est réflexive et transitive.*

À partir d'un préordre $\preceq$, on définit le préordre strict $\prec$ en posant :

$$M \prec N \Leftrightarrow M \preceq N \text{ and } N \not\preceq M$$

Un préordre sur un ensemble $\mathcal{A}$ est *bien fondé*, quand il n'existe pas de suite infinie d'éléments de $\mathcal{A}$ qui soit strictement décroissante. C'est à dire qu'il n'existe pas de suite $(a_i)_{i>1}$ d'éléments de $\mathcal{A}$, telle que $\forall i > 1, x_{i+1} \prec x_i$. On peut maintenant comprendre une version restreinte du théorème de Kruskal [Lév 80, remark 1] ou [Gal 89, proposition 4.9].

Theorème 3.3.2 (Kruskal) *Soit un préordre $\preceq$ sur les termes (étiquetés) qui vérifie les deux conditions de plongement :*

1. $\forall i \in [1 \dots a],\ M_i \prec f^l(M_1, M_2, \dots, M_a)$
2. $\forall i \in [1 \dots a],\ M_i \prec N_i \Rightarrow f^l(M_1, \dots, M_i, \dots, M_a) \prec f^l(M_1, \dots, N_i, \dots, M_a)$

Alors, le préordre $\preceq$ est bien fondé.

Les théorèmes des références citées portent sur les termes non-étiquetés construits à partir d'un ensemble fini de symboles fonctionnels. En fait, leurs preuves s'adaptent aisément au cas des termes étiquetés construits à partir d'un ensemble fini de symboles fonctionnels.

3.3.2 Dérivations bornées

L'idée est de limiter le niveau de création de radicaux dans les dérivations. Il s'agit là d'une notion qui est exprimable de façon très simple à l'aide des étiquettes complètes.

Définition 3.3.3 (Niveau de création) *Soit un radical de nom α, on définit le niveau de création du radical R comme l'entier $\mathcal{C}(\alpha)$, avec :*

$$\left\{ \begin{array}{rcl} \mathcal{C}(x) & = & 0 \\ \mathcal{C}(f^l(\alpha_1, \dots, \alpha_a)) & = & max(\mathcal{C}'(l), \mathcal{C}(\alpha_1), \dots, \mathcal{C}(\alpha_a)) \end{array} \right. \quad \textit{and} \quad \left\{ \begin{array}{rcl} \mathcal{C}'(a) & = & 0 \\ \mathcal{C}'(<\alpha,a>) & = & 1 + \mathcal{C}(\alpha) \\ \mathcal{C}'(l.e) & = & max(\mathcal{C}'(l), \mathcal{C}'(e)) \end{array} \right.$$

C'est à dire que le niveau de création d'un radical R est le degré d'imbrication maximal des noms de radicaux dans le nom de R.

On se donne une borne Max, qui est un entier strictement positif. Étant donné un radical R de nom α, on appelle poids relativement à la borne Max l'entier $\mathcal{P}_{\text{Max}}(\alpha)$ défini comme suit :

$$\left\{ \begin{array}{ll} \mathcal{P}_{\text{Max}}(\alpha) = \text{Max} - \mathcal{C}(\alpha) & \text{si } \mathcal{C}(\alpha) \leq \text{Max} \\ \mathcal{P}_{\text{Max}}(\alpha) = 0 & \text{si } \mathcal{C}(\alpha) > \text{Max} \end{array} \right.$$

Par définition, le radical R est *borné* par Max, si et seulement si son poids relativement à Max est strictement positif ($\mathcal{C}(\alpha) < \text{Max}$).

Définition 3.3.4 (Réduction bornée) *Soit une réduction A: $M \xrightarrow{U} N$. Par définition, la réduction A est bornée par Max, si et seulement si l'ensemble U n'est pas vide et ne contient que des occurrences de radicaux bornés.*

Une dérivation bornée est une dérivation dont toutes les étapes sont des réductions bornées. La notion classique de développement donne lieu à des dérivations bornées particulières :

Définition 3.3.5 (Développements) *Soit un terme M, et soit U un ensemble de radicaux du terme M. On appelle développement de U, toute dérivation issue de M et construite de la façon suivante :*

- *La dérivation vide O est un développement de U.*
- *Soit A: $M \to^* N$ un développement de U. On construit un nouveau développement en concaténant la dérivation A avec la réduction B: $N \xrightarrow{V} P$, où V est un sous-ensemble non-vide de $U \setminus B$.*

Autrement dit, un développement ne contracte que les résidus d'un ensemble de radicaux donnés initialement. Il est immédiat qu'un développement est une dérivation bornée puisque les noms de radicaux se transmettent des radicaux initiaux aux radicaux résidus (lemme 2.3.8). Nous verrons dans le chapitre suivant un exemple moins trivial de dérivation bornée, les développements d'une dérivation.

Les étiquettes complètes renferment beaucoup d'information. Ici, on recherche plutôt une technique simple pour limiter le degré de création des radicaux au cours des dérivations, qu'une description exhaustive des relation de création entre radicaux. Pour cet objectif limité, les étiquettes simples sont suffisantes.

Définition 3.3.6 (Poids dans les étiquettes simples) *Soit un radical R* simplement étiqueté *de nom α et soit une borne Max. On note $|\tau(\alpha)|$ la longueur de l'étiquette de tête du nom α. Le poids relativement à Max du radical R est l'entier $\mathcal{P}_{\mathrm{Max}}(\alpha)$, défini comme suit :*

$$\begin{cases} \mathcal{P}_{\mathrm{Max}}(\alpha) = Max - |\tau(\alpha)| & si\ |\tau(\alpha)| \leq Max \\ \mathcal{P}_{\mathrm{Max}}(\alpha) = 0 & si\ |\tau(\alpha)| > Max \end{cases}$$

On définit ensuite les radicaux, les réductions et les dérivations bornées comme dans le cas des étiquettes complètes. Pour tout nom de radical α complètement étiqueté, on considère le nom de radical simplement étiqueté $S(\alpha)$ qui est la simplification de α (section 3.1.2). On montre très facilement l'inégalité : $|\tau(\mathcal{S}(\alpha))| \leq \mathcal{C}(\alpha)$. Par conséquent, la simplification par le morphisme S d'une dérivation complètement étiquetée bornée est une dérivation simplement étiquetée bornée. Il suffit donc de montrer l'inexistence des dérivations bornées infinies dans le cas de la réécriture simplement étiquetée, pour que cette propriété soit également vraie dans le cas de la réécriture complètement étiquetée.

Soit une règle simplement étiquetée $\alpha \to \beta$. De par le procédé d'étiquetage des règles, toutes les étiquettes qui décorent les sous-termes durs du membre droit β sont strictement plus longues que l'étiquette de tête du membre gauche α. C'est pourquoi la taille des étiquettes simples est une mesure grossière du "degré de création" du sous-terme qu'elle décore. Cette mesure de l'avancement des calculs est moins stricte que la mesure des degrés d'imbrications des noms de radicaux dans le cas de des étiquettes complètes. Par exemple, elle ne rend pas compte directement des règles de projection. Cependant, contraindre cette mesure simplifiée suffira pour exclure les dérivations infinies.

3.3.3 Finitude des dérivations bornées

Pour montrer que les dérivations bornées sont de longueur finie, on va construire un nouveau préordre sur les termes (simplement étiquetés), à partir de celui qui ordonne les étiquettes simples selon l'ordre inverse de leur longueur. On applique le schéma de construction des "recursive path orderings" (r.p.o.) donné dans [Lév 80]. Comme le nouveau préordre n'est pas un r.p.o. très compliqué, je donne toutes les preuves.

Définition 3.3.7 *Soient deux termes simplement étiquetés M et N. Par définition, on a $M \prec N$ si l'une au moins des trois conditions suivantes est vérifiée :*

(1) $M = f^l(M_1, \ldots, M_a)$, $N = g^m(N_1, \ldots, N_b)$, $\mathcal{P}_{\mathrm{Max}}(l) < \mathcal{P}_{\mathrm{Max}}(m)$ *et* $\forall i \in [1 \ldots a],\ M_i \prec N$

(2) $N = g^m(N_1, \ldots, N_b)$ *et* $\exists i \in [1 \ldots b],\ M \prec N_i$ *ou* $M = N_i$

(3) $M = f^l(M_1, \ldots, M_a)$, $N = f^l(N_1, \ldots, N_a)$ *et* $(M_1, M_2, \ldots, M_a) \prec_a (N_1, N_2, \ldots, N_a)$

On note $\preceq$ la fermeture réflexive de $\prec$. Dans cette définition $\prec_a$ est l'extension naturelle de $\prec$ aux n-uplets de taille a :

$$(M_1, M_2, \ldots, M_a) \prec_a (N_1, N_2, \ldots, N_a) \Leftrightarrow \forall i \in [1 \ldots a],\ M_i \preceq N_i \text{ et } \exists i \in [1 \ldots a],\ M_i \prec N_i$$

Lemme 3.3.8 *La relation $\prec$ satisfait les deux conditions de plongement.*

1. $\forall i \in [1 \ldots a],\ M_i \prec f^l(M_1, M_2, \ldots, M_a)$
2. $\forall i \in [1 \ldots a],\ M_i \prec N_i \Rightarrow f^l(M_1, \ldots, M_i, \ldots, M_a) \prec f^l(M_1, \ldots, N_i, \ldots, M_a)$

Démonstration : Pour montrer *1*, il suffit d'appliquer le cas (2) de la définition de $\prec$. Pour montrer *2*, on applique le cas (3) de la définition de $\prec$. □

Lemme 3.3.9 *La relation $\preceq$ est un préordre et la relation $\prec$ est le préordre strict associé.*

Démonstration : Il suffit de montrer que $\prec$ est transitive et n'est pas réflexive. Soit un terme M, on montre facilement par induction sur M que l'on a pas $M \prec M$. On montre ensuite la transitivité : soient trois termes M, N et P, tels que $M \prec N \prec P$. La preuve est par induction sur la somme de la taille de ces trois termes $|M| + |N| + |P|$. Chaque fois que nécessaire dans l'étape inductive de la preuve, on pose $M = f^l(M_1, M_2, \ldots, M_a)$, $N = g^m(N_1, N_2, \ldots, N_b)$ et $P = h^n(P_1, P_2, \ldots, P_c)$. On distingue les cas :

1-1 On a $\mathcal{P}_{\text{Max}}(l) < \mathcal{P}_{\text{Max}}(m) < \mathcal{P}_{\text{Max}}(n)$ et donc $\mathcal{P}_{\text{Max}}(l) < \mathcal{P}_{\text{Max}}(n)$, par transitivité de $<$. Pour tout i de l'intervalle $[1 \ldots a]$, on a $M_i \prec N \prec P$. Par induction, on a $M_i \prec P$ et donc $M \prec P$ par (1).

2-1 Il existe un entier i dans $[1 \ldots b]$, avec $M \preceq N_i$. Or, par (1), on a $N_i \prec P$. Si $M = N_i$ on a donc directement $M \prec P$. Sinon, on a $M \prec N_i \prec P$ et on termine en appliquant l'hypothèse d'induction.

3-1 Ici on a $l = m$ et $\mathcal{P}_{\text{Max}}(m) < \mathcal{P}_{\text{Max}}(n)$, soit $\mathcal{P}_{\text{Max}}(l) < \mathcal{P}_{\text{Max}}(n)$. En outre, pour tout entier i de l'intervalle $[1 \ldots a]$, on a $N_i \prec P$. Par ailleurs, par définition de $\prec_a$, on a $M_i \preceq N_i$. On poursuit la preuve en appliquant l'hypothèse d'induction dans le cas $M_i \prec N_i \prec P$, et le bon sens dans le cas $M_i = N_i \prec P$; ce qui permet de déduire $M \prec P$ par (1).

*-2 Il existe i dans l'intervalle [1... c], tel que $N \preceq P_i$. Si $N \prec P_i$, alors, on applique l'hypothèse d'induction à $M \prec N \prec P_i$ et on a $M \prec P$ par (2). Sinon, on a $N = P_i$, donc $M \prec P_i$ et on conclut directement $M \prec P$ par (2).

1-3 Ici on a $\mathcal{P}_{\text{Max}}(l) < \mathcal{P}_{\text{Max}}(m)$ et $m = n$, on en déduit immédiatement $\mathcal{P}_{\text{Max}}(l) < \mathcal{P}_{\text{Max}}(n)$. On achève ensuite la preuve comme dans le cas 1-1.

2-3 Il existe i dans l'intervalle entier $[1 \ldots b]$, tel que $M \preceq N_i$. en outre par (3) on a $N_i \preceq P_i$. L'implication $M \preceq N_i \preceq P_i \Rightarrow M \preceq P_i$ est une conséquence facile de l'hypothèse d'induction. On termine la preuve en utilisant (2).

3-3 Ici $f = g = h$ et $a = b = c$. Pour tout entier i dans $[1 \ldots a]$ on montre facilement $M_i \preceq P_i$, à partir de $M_i \preceq N_i \preceq P_i$ et de l'hypothèse d'induction. De même pour l'entier j dans $[1 \ldots a]$, tel que $M_j \prec N_j$, on a $M_j \prec P_j$, à partir de $M_j \prec N_j \preceq P_j$. On prouve donc $M \prec P$ par (3).

Les cas de base sont en fait examinés dans l'étape inductive ci-dessus, au cours des cas 2-* ($M = x$ ou $M = f^l()$ et $\exists i \in [1 \ldots b],\ N_i = M$), des cas 1-* et des cas *-1 ($a = 0$, $b = 0$ ou $c = 0$). □

Ces deux derniers lemmes montrent que la relation $\preceq$ est un préordre bien fondé. Il reste à faire le lien entre $\preceq$ et les dérivations bornés.

Lemme 3.3.10 *Soit une dérivation bornée A: $M \to^* N$. Alors, on a $N \prec M$.*

Démonstration: Soit une réduction A: $M \xrightarrow{U} N$ qui est telle que tous les radicaux de U sont bornés. On montre $N \preceq M$ avec $U \neq \emptyset \Rightarrow N \prec M$.

1. Si A est du type (id), alors $U = \emptyset$, $x^l = N \preceq M = x^l$.
2. Si A est du type (rec), alors on pose :

$$\frac{A_i\colon M_i \xrightarrow{U_i} N_i \text{ si } i \in \mathrm{dom}(f) \qquad A_i\colon M_i = N_i \text{ si } i \notin \mathrm{dom}(f)}{A\colon f^l(M_1, M_2, \ldots, M_a) \xrightarrow{U} f^l(N_1, N_2, \ldots, N_a)}\,(\mathrm{rec})$$

   On prouve facilement $N \preceq M$, par le cas (3) de la définition de $\prec$ et à partir de l'hypothèse d'induction $\forall i \in [1 \ldots a],\ N_i \preceq M_i$. En outre, si U n'est pas vide, alors l'un au moins des U_i n'est pas vide et donc, par induction, l'une au moins des inégalités $N_i \preceq M_i$ est stricte. Par (3) de la définition de $\prec$, on a finalement $M \prec N$.
3. Si A est du type (red), alors U contient au moins l'occurrence à la racine ϵ et est donc non-vide. On pose :

$$\frac{A_x\colon \sigma(x) \to \rho(x) \text{ si } x \in \mathcal{V}_a(\alpha) \qquad A_x\colon \sigma(x) = \rho(x) \text{ si } x \in \mathcal{V}_i(\alpha)}{A\colon \sigma(\alpha) \to \rho(\beta)}\,(\mathrm{red})$$

   Pour tout sous-terme P de β, on prouve $\rho(P) \prec \sigma(\alpha)$ par induction sur la structure de P. Rappelons que les variables de β ne portent pas d'étiquette et que les autres sous-termes $P = g^l(P_1, P_2, \ldots, P_b)$ portent une étiquette de la forme $l = \tau(\alpha).a$, où a est une étiquette alphabétique.

   - Si P est une variable x, alors on a $\rho(x) \preceq \sigma(x)$, à partir de A_x: $\sigma(x) \to \rho(x)$, et $\sigma(x) \prec \sigma(\alpha)$ par application répétée du cas (2) de la définition de $\prec$ —on a bien l'inégalité stricte, car α ne peut pas se réduire à une variable—. On termine la preuve en utilisant la transitivité de $\prec$.
   - Si P est de la forme $g^l(P_1, P_2, \ldots, P_b)$, alors on a $\mathcal{P}_{\mathrm{Max}}(P) < \mathcal{P}_{\mathrm{Max}}(\alpha)$. En effet la réduction A est bornée par Max et on a donc $|\tau(\alpha)| < \mathrm{Max}$, soit $|l| = |\tau(\alpha)| + 1 \leq \mathrm{Max}$ et $\mathcal{P}_{\mathrm{Max}}(P) = \mathrm{Max} - |l| < \mathrm{Max} - |\tau(\alpha)| = \mathcal{P}_{\mathrm{Max}}(\alpha)$. Il reste à faire appel, si nécessaire ($b > 0$), à l'hypothèse d'induction $\forall i \in [1 \ldots b],\ \rho(P_i) \prec M$, pour prouver $\rho(P) \prec M$ par le cas (1) de la définition de $\prec$. □

On peut enfin conclure :

Proposition 3.3.11 *Toutes les dérivations bornées sont de longueur finie.*

Démonstration : Toute dérivation infinie $A{:}\, M_0 \xrightarrow{U_1} M_1 \xrightarrow{U_2} M_2 \xrightarrow{U_3} \ldots$ définit une suite infinie de termes $(M_i)_{0 \leq i}$. Si la dérivation A est bornée, alors cette suite est strictement décroissante pour le préordre $\prec$ associé à cette borne, ce qui est exclu, le préordre $\prec$ étant bien fondé. □

Le théorème des développements finis est un corollaire de ce résultat, car les développements sont des dérivations bornées particulières.

Un aspect intéressant de cette présentation technique des développements finis est qu'elle n'utilise aucune des hypothèses d'orthogonalité. A priori, il n'y a pas de difficulté particulière pour étiqueter des systèmes non-orthogonaux, mais il est clair que la confluence du système non-étiqueté n'entraînera généralement pas la confluence du système étiqueté. Par ailleurs, il n'est pas immédiat de définir les résidus dans les systèmes de réécriture qui ne sont pas orthogonaux. De sorte que le résultat sur les dérivations bornées, qui reste valable dans le cadre dans le cadre des systèmes non-orthogonaux, devient difficile à interpréter en termes de développements et qu'il perd de sa signification intuitive.

Chapitre 4

Étude des dérivations

4.1 Résidus des dérivations

Tout le but de cette section est d'étendre l'opération résidu aux dérivations. En fait, étant données une dérivation A: $M \rightarrow^* N$ et une réduction B: $M \xrightarrow{U} Q$, on sait déjà calculer l'ensemble $U \setminus A$ des résidus par la dérivation A des radicaux contractés par la réduction B (définition 1.2.18 et lemme 2.5.3). On pose donc la réduction résidu $B \setminus A$ égale à la réduction issue du terme N qui contracte les radicaux de $U \setminus A$. Soit ensuite la suite des étapes élémentaires de la dérivation A :

$$A\colon M = M_0 \xrightarrow{U_1} M_1 \xrightarrow{U_2} \ldots \xrightarrow{U_n} M_n = N$$

Le lemme des déplacements parallèles va permettre la définition de d'une dérivation A': $N \rightarrow^* Q$ qui est la dérivation résidu de la dérivation A par la réduction B. On la note bien évidemment $A' = A \setminus B$. La dérivation A' possède n étapes élémentaires :

$$A'\colon P = M'_0 \xrightarrow{U'_1} M'_1 \xrightarrow{U'_2} \ldots \xrightarrow{U'_n} M'_n = Q$$

Les étapes élémentaires A'_i de la dérivation A'_1 se calculent itérativement. À chaque étape élémentaire A_i de la dérivation A correspondent une étape résidu A'_i et une réduction intermédiaire B_i utilisée pour le calcul de la réduction $B \setminus A$. Initialement, on pose $B_0 = B$, puis on calcule les étapes A'_i dans l'ordre des i croissants :

$$\forall i \in\ [1 \ldots n], \quad A'_i\colon M'_{i-1} \xrightarrow{U_i \setminus B_{i-1}} M'_i \quad \text{et} \quad B_i = B_{i-1} \setminus A'_i$$

La figure 4.1 illustre ce processus itératif. Le point important est que, pour tout entier i de l'in-

$$\begin{tikzcd}
M_0 \arrow[r, "A_1"'] \arrow[d, "B_0"'] & M_1 \arrow[r, "A_2"'] \arrow[d, "B_1"'] & M_2 \cdots\cdots & M_{i-1} \arrow[r, "A_i"'] \arrow[d, "B_{i-1}"'] & M_i \cdots\cdots & M_n \arrow[d, "B_n"'] \\
M'_0 \arrow[r, "A'_1"'] & M'_1 \arrow[r, "A'_2"'] & M'_2 \cdots\cdots & M'_{i-1} \arrow[r, "A'_i"'] & M'_i \cdots\cdots & M'_n
\end{tikzcd}$$

Figure 4.1 : Calcul des étapes élémentaires de la dérivation $A' = A \setminus B$.

tervalle $[1 \ldots n[$, les réductions A'_i et A'_{i+1} sont concaténables en raison de la convergence en M'_i

des dérivations $A_i \sqcup B_{i-1}$ et $B_{i-1} \sqcup A_i$ (cf. le lemme des déplacements parallèles). Il est clair que la réduction B_n est la réduction qui contracte les résidus des radicaux de U par la dérivation A, on a donc $B_n = B \setminus A$. On peut donc, en généralisant la notation $\sqcup$, affirmer que, pour toute dérivation A: $M \to^* N$ et toute réduction B: $M \to P$, les deux dérivations $A \sqcup B$ et $B \sqcup A$ sont convergentes, puisqu'elles aboutissent toutes deux en $Q = M'_n$.

Il reste maintenant à itérer la construction de la dérivation résidu par une réduction pour définir les résidus les plus généraux. On considère deux dérivations A: $M \to^* N$ et B: $M \to^* P$, mais c'est cette fois la dérivation B que l'on présente sous forme d'une décomposition en étapes élémentaires :

$$B\colon M = M_0 \xrightarrow{U_1} M_1 \xrightarrow{U_2} \dots \xrightarrow{U_n} M_n = P$$

On considère ensuite une suite de dérivations $A_0, A_1, \dots, A_n$ définie par

$$A_0 = B \qquad \forall i \in [1 \dots n], \quad A_i = A_{i-1} \setminus B_i$$

Et on pose finalement $A \setminus B = A_n$. Cette construction est illustrée par la figure 4.2. Il faut noter que la construction générale de la dérivation $A \setminus B$ englobe la définition du résidu d'une réduction A par une dérivation B que l'on a vue au tout début de ce chapitre.

$$\begin{tikzcd}
M_0 \arrow[r, "B_1"'] \arrow[d, "A_0"', "*"] & M_1 \arrow[r, "B_2"'] \arrow[d, "A_1"', "*"] & M_2 \cdots\cdots\cdots & M_{i-1} \arrow[r, "B_i"'] \arrow[d, "A_{i-1}"', "*"] & M_i \cdots\cdots\cdots \arrow[d, "A_i"', "*"] & M_n \arrow[d, "A_n"', "*"] \\
M'_0 \arrow[r, "B'_1"'] & M'_1 \arrow[r, "B'_2"'] & M'_2 \cdots\cdots\cdots & M'_{i-1} \arrow[r, "B'_i"'] & M'_i \cdots\cdots\cdots & M'_n
\end{tikzcd}$$

Figure 4.2 : Calcul de la dérivation $A_n = A \setminus B$.

Un autre aspect de la construction de la figure 4.2 est l'existence d'une dérivation B': $N \to^* Q$ dont les étapes élémentaires se calculent en même temps que les dérivations A_i, à raison d'une étape B'_i par étape B_i :

$$\forall i \in [1 \dots n], \qquad B'_i = B_i \setminus A_{i-1}$$

La construction de la dérivation B' est correcte, car, à chaque étape, la convergence des dérivations $A_{i-1} \sqcup B_i$ et $B_i \sqcup A_{i-1}$ assure que la réduction B'_i aboutit au terme M'_i, dont est issue l'étape élémentaire suivante $B'_{i+1} = B_{i+1} \setminus A_i$. Enfin, la convergence des dérivations $A_{n-1} \sqcup B_n$ et $B_n \sqcup A_{n-1}$ assure que les dérivations B' et $A_n = A_{n-1} \setminus B_n$ aboutissent au même terme $Q = M'_n$.

La propriété de Church-Rosser est donc prouvée. Quelles que soient les dérivations A: $M \to^* N$ et B: $M \to^* P$, on sait construire deux dérivations $A \setminus B$ et B', avec $A \setminus B$: $N \to^* Q$ et B': $P \to^* Q$. Il reste bien entendu à montrer que l'on a $B' = B \setminus A$. Pour le moment, nous n'en savons rien et on pose $B \rhd A = B'$.

Les deux constructions des figures 4.1 et 4.2 se traduisent par quatre formules.

Définition 4.1.1 *Soit* $A\colon M \to^* N$ *et* $B\colon M \to^* P$ *deux dérivations issues d'un même terme. On définit la dérivation résidu* $A \setminus B$ *à partir des égalités suivantes :*

A	B	$A \setminus B$
O	$\lvert B \rvert = 1$	O
$A_1 A_2$	$\lvert B \rvert = 1$	$(A_1 \setminus B)\,(A_2 \setminus (B \setminus A_1))$
quelconque	O	A
quelconque	$B_1 B_2$	$(A \setminus B_1) \setminus B_2$

Les constructions préalablement présentées sous forme de figures illustrent bien que cette définition ne dépend pas des décompositions $A = A_1 A_2$ et $B = B_1 B_2$ choisies dans les cas (2) et (4). Il est également clair que, pour toutes dérivations A et B, on a $|A \setminus B| = |A|$. On outre, on définit temporairement l'opérateur $\triangleright$ sur les dérivations issues d'un même terme par :

A	B	$A \triangleright B$
$\lvert A \rvert \leq 1$	quelconque	$A \setminus B$
quelconque	$\lvert B \rvert \leq 1$	$A \setminus B$
O	quelconque	O
$A_1 A_2$	B	$(A_1 \triangleright B)\,(A_2 \triangleright (B \setminus A_1))$

Le quatrième cas de cette définition suppose la convergence des dérivations $A_1(B \setminus A_1)$ et $B(A_1 \triangleright B)$, qui est vraie par construction.

Lemme 4.1.2 *Pour toutes dérivations* $A\colon M \to^* N$ *et* $B\colon M \to^* P$*, on a l'égalité* $A \setminus B = A \triangleright B$*.*

Démonstration : On procède par induction sur la somme $|A| + |B|$. Les cas de base, $|A| \leq 1$ ou $|B| \leq 1$, sont résolus directement par la définition de l'opérateur $\triangleright$. Soient ensuite deux dérivations $A = A_1 A_2$ et $B = B_1 B_2$, où A_1 et B_1 sont des réductions ($|A_1| = |B_1| = 1$). Par le quatrième et le deuxième cas de la définition des résidus, on a :

$$A \setminus B = ((A_1 \setminus B_1)\,(A_2 \setminus (B_1 \setminus A_1))) \setminus B_2$$

On peut ensuite appliquer l'hypothèse d'induction ($|A_1 \setminus B_1| + |A_2 \setminus (B_1 \setminus A_1)| + |B_2| = |A_1| + |A_2| + |B_2| < |A| + |B|$) :

$$A \setminus B = ((A_1 \setminus B_1)\,(A_2 \setminus (B_1 \setminus A_1))) \triangleright B_2$$

On poursuit par le quatrième cas de la définition de l'opérateur $\triangleright$:

$$A \setminus B = ((A_1 \setminus B_1) \triangleright B_2)\,((A_2 \setminus (B_1 \setminus A_1)) \triangleright (B_2 \setminus (A_1 \setminus B_1)))$$

Par un aller retour entre les opérateurs $\setminus$ et $\triangleright$ et par le quatrième cas de la définition de $\setminus$, on prouve facilement l'égalité (1) ci-après :

$$(A_1 \setminus B_1) \triangleright B_2 = (A_1 \setminus B_1) \setminus B_2 = A_1 \setminus B = A_1 \triangleright B$$

Pour le second terme on utilise encore une fois l'hypothèse d'induction, puis le quatrième cas de la définition des résidus :

$$(A_2 \setminus (B_1 \setminus A_1)) \triangleright (B_2 \setminus (A_1 \setminus B_1)) = A_2 \setminus ((B_1 \setminus A_1)\,(B_2 \setminus (A_1 \setminus B_1)))$$

Par un aller retour entre $\backslash$ et $\triangleright$ et par le quatrième cas de la définition de l'opérateur $\triangleright$, il vient :

$$(B_1 \setminus A_1)\,(B_2 \setminus (A_1 \setminus B_1)) = (B_1 \triangleright A_1)\,(B_2 \triangleright (A_1 \setminus B_1)) = B \triangleright A_1 = B \setminus A_1$$

Soit finalement, en réutilisant l'égalité (1) ci-dessus :

$$A \setminus B = (A_1 \triangleright B)\,(A_2 \setminus (B \setminus A_1))$$

Il reste à utiliser une dernière fois l'hypothèse d'induction et le quatrième cas de la définition de $\triangleright$:

$$A \setminus B = (A_1 \triangleright B)\,(A_2 \triangleright (B \setminus A_1)) = A \triangleright B$$

□

Une première conséquence un peu technique de ce lemme est que l'on peut s'affranchir des contraintes de la définition des résidus.

Lemme 4.1.3 *Soient deux dérivations A: $M \to^* N$ et B: $M \to^* P$. Quelles que soient les décompositions $A = A_1 A_2$ et $B = B_1 B_2$, on a :*

$$\begin{aligned} O \setminus B &= O && (4.1)\\ (A_1 A_2) \setminus B &= (A_1 \setminus B)\,(A_2 \setminus (B \setminus A_1)) && (4.2)\\ A \setminus O &= A && (4.3)\\ A \setminus (B_1 B_2) &= (A \setminus B_1) \setminus B_2 && (4.4) \end{aligned}$$

Une autre conséquence très importante du lemme 4.1.2 est le lemme des déplacement parallèles généralisés.

Lemme 4.1.4 (Déplacements parallèles généralisés) *Soient deux dérivations A et B, issues d'un même terme M. On a les propositions suivantes :*

1. *Les dérivations $A \sqcup B$ et $B \sqcup A$ aboutissent au même terme.*
2. *Pour tout ensemble d'occurrences U de M, on a l'égalité $U \setminus (A \sqcup B) = U \setminus (B \sqcup A)$.*
3. *Pour toute famille de radicaux $\mathcal{F}$ dans M, on a les égalités :*

$$\begin{aligned} \mathcal{F} \setminus (A \sqcup B) &= \mathcal{F} \setminus (A \sqcup B)\\ \mathcal{F} \backslash\!\backslash (A \sqcup B) &= \mathcal{F} \backslash\!\backslash (A \sqcup B)\\ \mathcal{F} \backslash\!\backslash\!\backslash (A \sqcup B) &= \mathcal{F} \backslash\!\backslash\!\backslash (A \sqcup B) \end{aligned}$$

Démonstration : On sait déjà que les dérivations $A(B \triangleright A)$ et $B(A \setminus B)$ sont convergentes. De $B \triangleright A = B \setminus A$, il vient immédiatement que les dérivations $A \sqcup B$ et $B \sqcup A$ convergent. Ce résultat est valable quel que soit le terme initial M. On prouve donc facilement les propriétés d'invariance *2* et *3* en appliquant la propriété de convergence *1* à des réétiquetages adéquat des dérivations A et B (cf. les preuve des lemmes 3.2.3 et 3.2.4). □

Il est souvent commode de donner une présentation figurative du lemme des déplacements parallèles généralisés et de sa "preuve" par induction. Soient donc deux dérivation A et B, issues d'un même terme et décomposables sous la forme $A = A_1 A_2$ et $B = B_1 B_2$. On pose les définitions suivantes :

$$A_1' = A_1 \setminus B_1, \quad B_1' = B_1 \setminus A_1, \quad A_2' = A_2 \setminus B_1', \quad B_2' = B_2 \setminus A_1'$$

$$A_1'' = A_1' \setminus B_2, \quad A_2'' = A_2' \setminus B_1', \quad B_1'' = B_1' \setminus A_1, \quad B_2'' = B_2' \setminus A_1'$$

Alors, la figure 4.3 exprime la convergence des dérivations $A \sqcup B$ et $B \sqcup A$ comme la fermeture du plus grand carré, qui est elle-même une conséquence de la fermeture des quatre carrés qui le décomposent.

$$
\begin{tikzcd}
M \arrow[r, "A_1", "*"{pos=0.95}] \arrow[d, "B_1"', "*"{pos=0.95}] & {} \arrow[r, "A_2", "*"{pos=0.95}] \arrow[d, "B_1'"', "*"{pos=0.95}] & {} \arrow[d, "B_1''"', "*"{pos=0.95}] \\
{} \arrow[r, "A_1'", "*"{pos=0.95}] \arrow[d, "B_2"', "*"{pos=0.95}] & M_1 \arrow[r, "A_2'", "*"{pos=0.95}] \arrow[d, "B_2'"', "*"{pos=0.95}] & {} \arrow[d, "B_2''"', "*"{pos=0.95}] \\
{} \arrow[r, "A_1''", "*"{pos=0.95}] & {} \arrow[r, "A_2''", "*"{pos=0.95}] & M_2
\end{tikzcd}
$$

Figure 4.3 : Décomposition de $(A_1A_2) \sqcup (B_1B_2)$ et $(B_1B_2) \sqcup (A_1A_2)$

Lemme 4.1.5 *Pour toute dérivation A: $M \to^* N$ et tout entier n, on a les propriétés :*

$$\emptyset^n \setminus A = \emptyset^n \qquad A \setminus \emptyset^n = A \qquad A \setminus A = \emptyset^{|A|}$$

Démonstration:

1. On montre cette égalité par induction sur n. Le cas de base $\emptyset \setminus A = \emptyset$ est immédiat, car par la définition 1.2.18 des résidus d'ensembles d'occurrences, l'ensemble vide n'a pas de résidus. On a ensuite :
$$\begin{array}{rcll} \emptyset^{n+1} \setminus A & = & (\emptyset \setminus A)\,(\emptyset^n \setminus (A \setminus \emptyset)) & \text{décomposition 4.2} \\ & = & \emptyset(\emptyset^n \setminus (A \setminus \emptyset)) & \text{cas de base} \\ & = & \emptyset\,\emptyset^n & \text{induction} \end{array}$$

2. On procède aussi par induction sur n. Mais il faut d'abord montrer le cas de base $A \setminus \emptyset = A$ par induction sur $|A|$. Si la dérivation A est en fait une réduction, alors, par le lemme 1.2.10, les radicaux contractés par la réduction A sont leurs propres résidus dans M par la réduction $\emptyset$. Sinon, la dérivation A se décompose en $A = BC$, et on a :
$$\begin{array}{rcll} A \setminus \emptyset & = & (B \setminus \emptyset)(C \setminus (\emptyset \setminus B)) & \text{décomposition 4.4} \\ & = & B(C \setminus (\emptyset \setminus B)) & \text{induction, } |B| < |A| \\ & = & B(C \setminus \emptyset) & \text{1 ci-dessus} \\ & = & BC & \text{induction, } |C| < |A| \end{array}$$
On termine la preuve par l'étape inductive sur n :
$$\begin{array}{rcll} A \setminus \emptyset^{n+1} & = & (A \setminus \emptyset) \setminus \emptyset^n & \text{décomposition 4.4} \\ & = & A \setminus \emptyset^n & \text{cas de base} \\ & = & A & \text{induction} \end{array}$$

3. Cette dernière égalité se montre par induction sur $|A|$. Si la dérivation A est en fait une réduction, alors on a $\mathcal{R}(A) \setminus A = \emptyset$ (lemme 1.2.11). Soit, du point de vue des réductions,

$A \setminus A = \emptyset$. Sinon, on décompose la dérivation A en deux dérivations non vides $A = BC$:

$$\begin{array}{rcll} A \setminus A & = & (A \setminus B) \setminus C & \text{décomposition 4.4} \\ & = & ((B \setminus B)\,(C \setminus (B \setminus B))) \setminus C & \text{décomposition 4.2} \\ & = & (\emptyset^{|B|}(C \setminus \emptyset^{|B|})) \setminus C & \text{induction, } |B| < |A| \\ & = & (\emptyset^{|B|}C) \setminus C & \text{2 ci-dessus} \\ & = & (\emptyset^{|B|} \setminus C)\,(C \setminus (C \setminus \emptyset^{|B|})) & \text{décomposition 4.2} \\ & = & \emptyset^{|B|}\,(C \setminus C) & \text{1 et 2 ci-dessus} \\ & = & \emptyset^{|B|}\emptyset^{|C|} & \text{induction, } |C| < |B| \end{array}$$

□

Enfin, il reste à présenter un résultat particulièrement important qui traduit les très fortes similitudes qui existent entre les deux dérivations “permutées” $A \sqcup B$ et $B \sqcup A$.

Lemme 4.1.6 (Lemme du cube) *Soient deux dérivations A: $M \to^* N$ et B: $M \to^* P$. Alors, pour toute dérivation C: $M \to^* Q$. On a l'égalité :*

$$C \setminus (A \sqcup B) = C \setminus (B \sqcup A)$$

Démonstration : La preuve [Hue 80] est par induction sur $|A|+|B|+|C|$. Le cas où la dérivation C est la dérivation vide O se résoud directement par l'équation 4.1. Sinon, on décompose la dérivation C sous la forme $C = C_1C_2$, où C_2 est une réduction ($|C_2| = 1$). Soit, par la décomposition 4.2, appliquée deux fois :

$$\begin{array}{rcl} C \setminus (A \sqcup B) & = & (C_1 \setminus (A \sqcup B))\,(C_2 \setminus ((A \sqcup B) \setminus C_1)) \\ C \setminus (B \sqcup A) & = & (C_1 \setminus (B \sqcup A))\,(C_2 \setminus ((B \sqcup A) \setminus C_1)) \end{array}$$

L'égalité recherchée se ramène donc à deux égalités. La première $C_1 \setminus (A \sqcup B) = C_1 \setminus (B \sqcup A)$ vient par hypothèse d'induction ($|A| + |B| + |C_1| < |A| + |B| + |C|$). Pour la seconde, on montre d'abord un résultat intermédiaire :

$$\begin{array}{rcll} (A \sqcup B) \setminus C_1 & = & (A \setminus C_1)\,((B \setminus A) \setminus (C_1 \setminus A)) & \text{décomposition 4.2} \\ & = & (A \setminus C_1)\,(B \setminus (A\,(C_1 \setminus A))) & \text{recomposition 4.4} \\ & = & (A \setminus C_1)\,(B \setminus (A \sqcup C_1)) & \text{définition de } \sqcup \\ & = & (A \setminus C_1)\,(B \setminus (C_1 \sqcup A)) & \text{induction, } |A| + |C_1| + |B| < |A| + |B| + |C| \\ & = & (A \setminus C_1)\,(B \setminus (C_1\,(A \setminus C_1))) & \text{définition de } \sqcup \\ & = & (A \setminus C_1)\,((B \setminus C_1) \setminus (A \setminus C_1)) & \text{décomposition 4.4} \\ & = & (A \setminus C_1) \sqcup (B \setminus C_1) & \text{définition de } \sqcup \end{array}$$

Par symétrie, on a aussi $(B \sqcup A) \setminus C_1 = (B \setminus C_1) \sqcup (A \setminus C_1)$. On peut maintenant appliquer le lemme des déplacements parallèles généralisés (lemme 4.1.4-2) à l'ensemble des radicaux contractés par la réduction C_2 pour obtenir l'égalité, $C_2 \setminus ((A \setminus C_1) \sqcup (B \setminus C_1)) = C_2 \setminus ((B \setminus C_1) \sqcup (A \setminus C_1))$. Soit finalement : $C_2 \setminus ((A \sqcup B) \setminus C_1) = C_2 \setminus ((B \sqcup A) \setminus C_1)$ □

La figure 4.4 illustre le lemme du cube d'une façon que l'on va décrire. La fermeture des trois carrés “supérieurs” entièrement tracés en gras, résulte du lemme des déplacements parallèles généralisés, appliqué aux couples de dérivations (A, B), (B, C) et (A, C). La convergence des trois flèches tracées au trait fin A', B' et C' traduit le lemme du cube, cette convergence entraîne la fermeture du cube complet. Il faut bien comprendre que c'est la fermeture complète du cube qui traduit le lemme du cube et non pas celle d'une de ses faces “inférieures”. Considérons par exemple les flèches

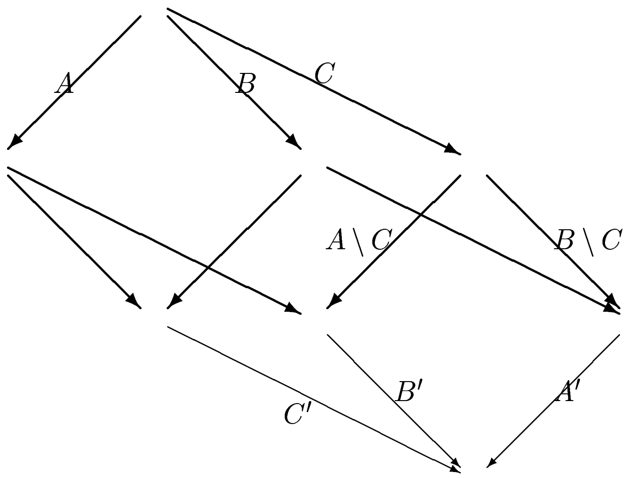


Figure 4.4 : Le lemme du cube

marquées A' et B'. Ces deux flèches s'obtiennent par un transport le long du cube qui commence par suivre l'arête marquée C. La flèche B est tout d'abord portée sur la flèche $B \setminus C$. De même, la flèche A est portée sur la flèche $A \setminus C$. La fermeture de la face "postérieure" du cube (celle qui délimitée par les arêtes $A \setminus C$, $B \setminus C$, A' et B') résulte ensuite du transport mutuel des flèches $B \setminus C$ et $A \setminus C$. Autrement dit, c'est le lemme des déplacement parallèles appliqué aux dérivations $A \setminus C$ et' $B \setminus C$ qui permet de fermer la face "postérieure" du cube. En symboles, on a :

$$A' = (A \setminus C) \setminus (B \setminus C) \quad \text{et} \quad B' = (B \setminus C) \setminus (A \setminus C)$$

De même, en transportant la face (C, B) le long de l'arête A du cube, on prouve la fermeture d'une autre face inférieure du cube. Le lemme du cube intervient alors pour fermer une partie du cube lui même, en prouvant que les deux faces inférieures précédemment fermées ont une arête commune, qui correspond à la dérivation $B' = B \setminus (C \sqcup A) = B \setminus (A \sqcup C)$.

4.2 Treillis des dérivations

4.2.1 Dérivations équivalentes

Soit un terme M et deux dérivations A et B issues de M. Les dérivations $A \sqcup B$ et $B \sqcup A$ sont très semblables. En effet, elles ne se contentent pas d'aboutir au même terme N, mais en plus, résidus, descendants et héritiers dans N sont les mêmes, que l'on choisisse d'aller de M en N par $A \sqcup B$ ou $B \sqcup A$. Toutes ces propriétés de conservation permettent de définir des relations d'équivalence entre les dérivations. Toutes ces équivalences identifient les dérivations permutées $A \sqcup B$ et $B \sqcup A$.

Notation : Soit un terme complètement étiqueté M. On note $\mathcal{D}(M)$ l'ensemble des dérivations finies qui sont issues du terme M.

Les relations d'équivalence qui identifient les dérivations permutées sont à définir sur l'ensemble de dérivations $\mathcal{D}(M)$, le terme M étant fixé. Certaines de ces relations sont étudiées et hiérarchisées dans [Klo 80]. Nous n'examinerons que la plus précise de ces équivalences, celle qui est donnée par le lemme du cube [Hue 79, Lév 78].

Définition 4.2.1 *Étant donné un terme M, on appelle équivalence des dérivations la relation binaire "$\equiv$" définie sur l'ensemble des dérivations issues de M par :*

$$A \equiv B \Leftrightarrow \forall C \in \mathcal{D}(M),\ C \setminus A = C \setminus A$$

Deux dérivations équivalentes ont le même terme final. En effet, soient A: $M \rightarrow^* N$ et B: $M \rightarrow^* P$ deux dérivations équivalentes. En posant $C = O$ dans la définition de $A \equiv B$, on obtient $O \setminus A = O \setminus B$. Or (définition 4.1.1), la dérivation $O \setminus A$ est la dérivation vide issue de N et $O \setminus A$ est la dérivation issue de P. On a donc $N = P$.

Corollaire 4.2.2 *Soient deux dérivations A: $M \rightarrow^* N$ et B: $M \rightarrow^* P$. Alors, on peut reformuler le lemme du cube.*

$$A \sqcup B \equiv B \sqcup A$$

Proposition 4.2.3 *L'équivalence $\equiv$ est une congruence. C'est à dire que, pour toutes les dérivations équivalentes A: $M \rightarrow^* N$ et B: $M \rightarrow^* N$, on a les résultats suivants :*

1. *Pour toute dérivation C: $N \rightarrow^* P$, on a $AC \equiv BC$.*
2. *Pour toute dérivation C: $P \rightarrow^* M$, on a $CA \equiv CB$.*
3. *Pour toute dérivation C: $M \rightarrow^* P$, on a $A \sqcup C \equiv B \sqcup C$.*

Démonstration:

1. Pour toute dérivation D, issue du terme M, on a :
$$D \setminus (AC) = (D \setminus A) \setminus C = (D \setminus B) \setminus C = D \setminus (BC)$$
2. Pour toute dérivation D, issue du terme P, on a :
$$D \setminus (CA) = (D \setminus C) \setminus A = (D \setminus C) \setminus B = D \setminus (CB)$$
3. Enfin, on peut déduire la dernière équivalence, par définition de $A \equiv B$, puis par 1 ci-dessus :
$$A \sqcup C = A\,(C \setminus A) = A\,(C \setminus B) \equiv B\,(C \setminus B) = B \sqcup C$$ □

Corollaire 4.2.4 *L'opérateur $\sqcup$ est compatible avec l'équivalence $\equiv$.*

Démonstration : Soient quatre dérivations A, A', B et B', avec $A \equiv A'$ et $B \equiv B'$. Par la propriété *3* ci-dessus et le lemme du cube, on a les équivalences successives :

$$A \sqcup B \equiv A \sqcup B' \equiv B' \sqcup A \equiv B' \sqcup A' \equiv A' \sqcup B'$$

□

On peut maintenant préciser les rapports entre la dérivation vide O: $M \rightarrow^* M$ et la réduction vide $\emptyset$: $M \rightarrow M$.

Lemme 4.2.5 *Pour toute dérivation A issue d'un terme M, on a l'équivalence :*

$$A \equiv O \Leftrightarrow A = \emptyset^n,\ \textit{pour } n \geq 0\ .$$

Démonstration : Supposons $A \equiv O$, alors, par définition de l'équivalence $\equiv$, on a $A \setminus A = A \setminus O$. Soit, par le lemme 4.1.5 et la définition des résidus de dérivation (équation 4.3), on a l'égalité $\emptyset^{|A|} = A$. Réciproquement, si la dérivation A est de la forme $A = \emptyset^n$, alors, pour toute dérivation C issue de M, on a $C \setminus \emptyset^n = C$ par le lemme 4.1.5. Par ailleurs, on a $C \setminus O = C$. Soit finalement, $\emptyset^n \equiv O$. □

Autrement dit, une dérivation est équivalente à la dérivation vide, si et seulement si toutes ses étapes élémentaires sont égales à la réduction vide. Une telle dérivation est une *dérivation nulle*.

4.2.2 Autres formulations de l'équivalence des dérivations

Définition 4.2.6 (Inclusion des dérivations) *Soit un terme M. On note $A \sqsubseteq B$ la relation définie sur les dérivations issues du terme M par :*

$$A \sqsubseteq B \Leftrightarrow A \setminus B = \emptyset^{|A|}$$

Les relations $\sqsubseteq$ et $\equiv$ sont intimement liées.

Lemme 4.2.7 *Soient deux dérivations A: $M \to^* N$ et B: $M \to^* P$. On a les équivalences :*

$$A \sqsubseteq B \Leftrightarrow B \equiv B \sqcup A \Leftrightarrow \exists C\colon N \to^* P,\ AC \equiv B$$

Démonstration : On note (1), (2) et (3) les trois propositions ci-dessus. Montrons tout d'abord l'implication (1) $\Rightarrow$ (2). Si on suppose $A \sqsubseteq B$, alors on a $B \sqcup A = B\,(A \setminus B) = B\emptyset^{|A|}$. Or, par le lemme 4.2.5, on a $\emptyset^{|A|} \equiv O$. Soit encore $B\emptyset^{|A|} \equiv B$ par le lemme 4.2.3. D'où il vient finalement $B \sqcup A \equiv B$ par transitivité. Pour montrer (2) $\Rightarrow$ (3), on emploie le lemme des déplacements parallèles. Si on suppose $B \equiv B \sqcup A$, alors on a $B \equiv A \sqcup B = A\,(B \setminus A)$. Enfin, on montre (3) $\Rightarrow$ (1). Si on suppose qu'il existe une dérivation C, telle que l'on a $AC \equiv B$, alors, par définition de l'équivalence $\equiv$, on a $(AC) \setminus B = (AC) \setminus (AC)$. Soit, par le lemme 4.1.5, $(AC) \setminus B = \emptyset^{|AC|}$. Et donc, en utilisant la décomposition 4.2, $A \setminus B = \emptyset^{|A|}$. Soit finalement, $A \sqsubseteq B$ □

Corollaire 4.2.8 *Étant donné un terme M, l'inclusion $\sqsubseteq$ est un préordre sur l'ensemble $\mathcal{D}(M)$ et l'équivalence $\equiv$ est la relation d'équivalence associée. C'est à dire que, quelles que soient les dérivations A et B de $\mathcal{D}(M)$, on a l'équivalence logique :*

$$A \equiv B \Leftrightarrow A \sqsubseteq B \text{ et } B \sqsubseteq A$$

Démonstration : La réflexivité de $\sqsubseteq$ est immédiate, par l'égalité $A \setminus A = \emptyset^{|A|}$. Soient ensuite A, B et C, trois dérivations de $\mathcal{D}(M)$. Des propriété de congruence de l'équivalence $\equiv$, il vient :

$$(A \sqsubseteq B \text{ et } B \sqsubseteq C) \Rightarrow \left\{ \begin{array}{l} \exists A',\ AA' \equiv B \\ \quad \text{et} \\ \exists B',\ BB' \equiv C \end{array} \right. \Rightarrow A(A'B') \equiv C \Rightarrow A \sqsubseteq C$$

Soient enfin A et B, deux dérivations issues du terme M. On les équivalences successives :

$$A \sqsubseteq B \text{ et } B \sqsubseteq A \Leftrightarrow B \equiv B \sqcup A \text{ et } A \equiv A \sqcup B \Leftrightarrow A \equiv B$$

□

La relation d'inclusion est très facilement décidable (il suffit de calculer une dérivation résidu). Ce résultat donne donc une procédure de décision pour la relation d'équivalence entre les dérivations.

Définition 4.2.9 (Équivalence de permutations (Berry)) *L'équivalence "$\sim$" ou équivalence de permutations est définie sur l'ensemble des dérivations issues d'un même terme de la manière suivante :*

- *Si A: $M \xrightarrow{U} N$ et B: $M \xrightarrow{V} P$ sont deux réductions, alors on pose $A \sqcup B \sim B \sqcup A$.*
- *Si A: $M \to^* N$ et B: $M \to^* N$ sont deux dérivations équivalentes ($A \sim B$), alors, pour toute dérivation C: $N \to^* P$, on pose $AC \sim BC$.*

- *Si A: $M \to^* N$ et B: $M \to^* P$ sont deux dérivations équivalentes ($A \sim B$), alors, pour toute dérivation P: $Q \to^* M$, on pose $CA \sim CB$.*
- *Si A: $M \to^* N$, B: $M \to^* P$ et C: $M \to^* Q$ sont trois dérivations telles que l'on a $A \sim B$ et $B \sim C$, alors on pose $A \sim C$.*
- *On pose $\emptyset \sim O$.*

Autrement dit, la relation $\sim$ est la plus petite congruence qui fait commuter le diagramme en diamant des déplacements parallèles. Il est facile de vérifier que la relation $\sim$ est bien une relation d'équivalence. On montre aussi très facilement que, pour tout entier n positif ou nul, on a $\emptyset^n \sim O$.

Lemme 4.2.10 *Les équivalences "$\equiv$" et "$\sim$" sont égales.*

Démonstration: On commence par prouver que, pour toutes dérivations A et B, on a $A \sqcup B \sim B \sqcup A$. On procède par induction sur $|A| + |B|$. Si $A = O$ ou $B = O$, alors, en posant $A = O$ par exemple, on obtient $A \sqcup B = B \sqcup A = B$. Soient maintenant deux dérivations $A = A_1 A_2$ et $B = B_1 B_2$, où A_1 et B_1 sont des réductions. On décompose les dérivations $A \sqcup B$ et $B \sqcup A$ comme indiqué par la figure 4.3. L'équivalence $A_1 B'_1 \sim B_1 A'_1$ (1) est assurée par le premier cas de la définition de l'équivalence de permutation. L'hypothèse d'induction appliquée aux trois autres carrés élémentaires de la figure 4.3 donne les trois équations :

$$B'_1 A'_2 \sim A_2 B''_1 \text{ (2)}, \quad B_2 A''_1 \sim A'_1 B'_2 \text{ (3)}, \quad B'_2 A''_2 \sim A'_2 B''_2 \text{ (4)}$$

Compte tenu des propriétés de congruence de l'équivalence $\sim$, on a les relations suivantes (on emploie successivement les équations (3), (1) et (4) et (2)) :

$$B \sqcup A = B_1(B_2 A''_1)A''_2 \sim (B_1 A'_1)(B'_2 A''_2) \sim A_1(B'_1 A'_2)B''_2 \sim A_1 A_2 B''_1 B''_2 = A \sqcup B$$

On suppose ensuite que les dérivations A et B sont telles que l'on a $A \equiv B$. De $A \sqsubseteq B$ et $B \sqsubseteq A$, on tire $B \sqcup A = B\emptyset^{|A|}$ et $A \sqcup B = A\emptyset^{|B|}$. Comme on vient de le montrer, on a donc $B\emptyset^{|A|} \sim A\emptyset^{|B|}$. Par ailleurs, puisque que l'on a $\emptyset^{|A|} \sim O$ et que $\sim$ est une congruence, on a $B = BO \sim B\emptyset^{|A|}$. De même, on prouve $A \sim A\emptyset^{|B|}$. On a donc finalement $B \sim A$ par transitivité de $\sim$. Réciproquement, on montre par induction sur la définition de $\sim$ que, pour toutes dérivations A et B, on a $A \sim B \Rightarrow A \equiv B$. Tout a déjà été montré dans la section précédente :

- Ce cas est une version restreinte aux réductions du lemme du cube.
- Voir la proposition 4.2.3-1.
- Voir la proposition 4.2.3-2.
- La relation $\equiv$ est transitive.
- Il s'agit du lemme 4.2.5 □

Nous disposons maintenant en tout de trois formulations de l'équivalence des dérivations. La première de ces formulations (définition 4.2.1) insiste sur un aspect "contextuel" de l'équivalence des dérivations : deux dérivations sont équivalentes quand elles se comportent exactement de la même façon vis à vis des résidus. La seconde formulation ($A \equiv B \Leftrightarrow A \sqsubseteq B \sqsubseteq A$) permet de répondre rapidement à la question "les dérivations A et B sont elles équivalentes ?". Enfin, la formulation par l'équivalence de permutations nous révèle la face intuitive de l'équivalence des dérivations : deux dérivations sont équivalentes quand elles contractent les même radicaux dans un ordre différent. En outre, l'emploi de l'équivalence de permutations est pratique lorsque l'on veut énumérer toutes les dérivations équivalentes à une dérivation donnée.

Dans une situation donnée, une des formulations de l'équivalence est souvent plus naturelle que les autres, comme le montre la preuve de ce résultat récapitulatif :

Proposition 4.2.11 *Soit deux dérivations équivalentes A et B issues d'un même terme M. Alors, on a les propriétés de conservation suivantes.*

1. *Il existe un terme N, avec $A\colon M{\to}^*N$ et $B\colon M{\to}^*N$.*
2. *Pour tout ensemble d'occurrences U de M, on a l'égalité : $U \setminus A = U \setminus B$.*
3. *Pour toute famille $\mathcal{F}$ de radicaux de M, on a les égalités :*

$$\mathcal{F} \setminus A = \mathcal{F} \setminus B \qquad \mathcal{F} \setminus\!\!\setminus A = \mathcal{F} \setminus\!\!\setminus B \qquad \mathcal{F} \setminus\!\!\setminus\!\!\setminus A = \mathcal{F} \setminus\!\!\setminus\!\!\setminus B$$

Démonstration : On peut facilement (re)démonter ces résultats à partir du lemme des déplacements parallèles (lemmes 3.2.3 et 3.2.4) et par induction sur la définition de l'équivalence de permutations. □

Proposition 4.2.12 *Étant donné un terme M, l'ensemble des classes d'équivalence des dérivations finies issues du terme M muni de l'ordre $\sqsubseteq$ forme un sup-demi-treillis dont $\sqcup$ est l'opération de borne supérieure. On note $<\mathcal{D}(M)/{\equiv}, \sqsubseteq, \sqcup>$ cette structure.*

Démonstration : On sait déjà que les opérations $\sqsubseteq$ et $\sqcup$ sont compatibles avec l'équivalence $\equiv$. Soient deux dérivations A et B dans $\mathcal{D}(M)$. On a $A \setminus (A \sqcup B) = (A \setminus A) \setminus (B \setminus A) = \emptyset^{|A|}$ (décomposition 4.4 et lemme 4.1.5), soit l'inclusion $A \sqsubseteq A \sqcup B$. Symétriquement, on a l'inclusion $B \sqsubseteq B \sqcup A$. Soit maintenant une dérivation C qui majore à la fois A et B. Par la décomposition 4.2, on a, $(A \sqcup B) \setminus C = (A \setminus C)\,((B \setminus A) \setminus (C \setminus A))$. Par l'hypothèse $A \sqsubseteq C$, la dérivation $A \setminus C$ est nulle. Par ailleurs, en suivant la décomposition 4.4, on peut réécrire la dérivation $(B \setminus A) \setminus (C \setminus A)$ comme $B \setminus (A \sqcup C)$, puis, par le lemme du cube, comme $B \setminus (C \sqcup A)$. Soit finalement :

$$(B \setminus A) \setminus (C \setminus A) = (B \setminus C) \setminus (A \setminus C) = \emptyset^{|B|} \setminus \emptyset^{|A|} = \emptyset^{|B|}$$

□

Une classe d'équivalence de dérivations digne d'intérêt est celle des dérivations normalisantes, c'est le plus grand élément du demi treillis $<\mathcal{D}(M)/\equiv, \sqsubseteq, \sqcup>$ quand le terme M est normalisable.

Lemme 4.2.13 *Soit un terme M qui admet une forme normale N et soit une dérivation normalisante $A\colon M{\to}^*N$, alors on a les deux résultats suivants :*

$$\forall B \in \mathcal{D}(M),\ B \sqsubseteq A \qquad \forall B\colon M{\to}^*N, A \equiv B$$

Démonstration : Comme il n'y a pas de radicaux dans la forme normale N, la dérivation $B \setminus A$ qui est issue de N ne peut pas ne pas être nulle. On a donc $B \sqsubseteq A$ dans le cas général. Si en outre la dérivation B est normalisante, alors on a $A \sqsubseteq B$ par symétrie. □

4.3 Standardisation

L'idée de la standardisation des dérivations est de donner une méthode d'organisation des contractions de radicaux qui puisse s'appliquer systématiquement pour joindre deux termes M et N par une dérivation $A\colon M{\to}^*N$, chaque fois que possible. En particulier, si le terme N est la forme normale du terme M, alors on recherche une méthode pour normaliser le terme M.

L'esprit général de cette méthode est de contracter les radicaux les plus extérieurs en premier, c'est à dire de travailler de la racine d'un terme vers ses feuilles. Si l'idée de base de cette technique est simple, sa description complète est moins immédiate, car il faut tenir compte du fait que l'ordre préfixe sur les occurrences n'est pas total et considérer les classes d'équivalence de dérivations.

$$\begin{tikzcd}
M \arrow[r, "C"'] \arrow[d, "B"', "*"] & Q \arrow[r, "D"', "*"] \arrow[d, "B'"', "*"] & N \\
P \arrow[r, "C'"'] & Q' \arrow[r, "D'"', "*"] & N'
\end{tikzcd}$$

Figure 4.5 : Preuve du lemme 4.3.2

4.3.1 Radicaux contractés

Certains des radicaux contractés à une étape élémentaire donnée d'une dérivation A: $M \rightarrow^* N$ sont déjà présents dans le terme initial M.

Définition 4.3.1 (Radicaux contractés) *Soit une dérivation* A: $M_0 \xrightarrow{U_1} M_1 \xrightarrow{U_2} \dots \xrightarrow{U_n} M_n$. *Soit* u *l'occurrence dans le terme* M_0 *d'un radical* R. *Par définition,* R *est un radical contracté par* A *et on note* $u \in \mathcal{R}(A)$, *si et seulement si il existe un entier* i *dans l'intervalle* $[1 \dots n]$, *avec :*

$$u \setminus A[1, i-1] \cap U_i \neq \emptyset$$

C'est à dire qu'un radical R est contracté par la dérivation A, si et seulement si lui même ou un de ses résidus est contracté par une étape élémentaire de A. De toute évidence, on a l'équivalence suivante :

$$A \equiv O \Leftrightarrow \mathcal{R}(A) = \emptyset$$

En outre, pour toute dérivation présentée comme une concaténation AB, on a l'équivalence :

$$u \in \mathcal{R}(AB) \Leftrightarrow u \in \mathcal{R}(A) \text{ ou } \exists v \in u \setminus A,\ v \in \mathcal{R}(B)$$

Étant donnée une dérivation A, on a pas en général $\mathcal{R}(A) \setminus A = \emptyset$, car rien ne dit que tous les résidus des radicaux de $\mathcal{R}(A)$ sont contractés par les étapes élémentaires de la dérivation A. Soit maintenant une dérivation B issue du même terme que la dérivation A, toujours à cause des résidus dupliqués, l'inclusion $\mathcal{R}(A) \setminus B \subseteq \mathcal{R}(A \setminus B)$ est fausse dans le cas général. Pour s'en convaincre il suffit de considérer les règles de réécriture$\{K \rightarrow L, F(x) \rightarrow G(x,x)\}$ et les dérivations :

$$A\colon F(K) \xrightarrow{\{\epsilon\}} G(K,K) \xrightarrow{\{1\}} G(L,K) \qquad B\colon F(K) \xrightarrow{\{\epsilon\}} G(K,K)$$

On a ici $\mathcal{R}(A) = \{\epsilon, 1\}$ et $\mathcal{R}(A) \setminus A = \{2\}$. En outre, on a la dérivation $A \setminus B$:

$$A \setminus B\colon G(K,K) \xrightarrow{\emptyset} G(K,K) \xrightarrow{\{1\}} G(L,K)$$

Soit $\mathcal{R}(A \setminus B) = \{1\}$, tandis que l'on a $\mathcal{R}(A) \setminus B = \{1, 2\}$. Toutefois, les radicaux contractés par la dérivation $A \setminus B$ qui sont des résidus par la dérivation B de radicaux du terme M sont bien les résidus par B de radicaux contractés par la dérivation A.

Lemme 4.3.2 *Soient deux dérivations* A: $M \rightarrow^* N$ *et* B: $M \rightarrow^* P$. *Alors pour toute occurrence* u *d'un radical dans le terme* P, *on a l'implication :*

$$u \in \mathcal{R}(A \setminus B) \text{ et } \exists v \in \mathcal{R}(M),\ u \in v \setminus B \Rightarrow v \in \mathcal{R}(A)$$

Démonstration : La preuve est par induction sur la longueur de la dérivation A. Si la dérivation A est la dérivation vide O, alors $A \setminus B = O \setminus B = O$ (équation 4.1), et donc $\mathcal{R}(A \setminus B) = \emptyset$. Soit maintenant une dérivation A, décomposée en une réduction C: $M \xrightarrow{U} Q$ et en une dérivation D: $Q \to^* N$. Par la décomposition 4.2, on a l'égalité $A \setminus B = C'D'$ (voir la figure 4.5). On suppose tout d'abord que le radical d'occurrence u dans P est contracté par la première étape C' de la dérivation $A \setminus B$. Alors, par définition des résidus de réductions, on a $u \in U \setminus B$, soit $v \in U$ et $v \in \mathcal{R}(A)$. Si on a pas $u \in \mathcal{R}(C')$, alors il existe dans le terme Q', une occurrence de radical w, avec $w \in u \setminus C'$ et $w \in \mathcal{R}(D')$. On a donc $w \in v \setminus (B \sqcup C)$. Or, par le lemme des déplacements parallèles généralisées, on a l'invariance $v \setminus (B \sqcup C) = v \setminus (C \sqcup B)$. Par conséquent, il existe un radical d'occurrence w' dans le terme Q avec $w' \in v \setminus C$ et $(w' \setminus B') \cap \mathcal{R}(D') \neq \emptyset$. Par hypothèse d'induction, on a $w' \in \mathcal{R}(D)$. Soit finalement, $v \in \mathcal{R}(A)$. □

D'autres radicaux initiaux restent totalement ignorés par les dérivations.

Définition 4.3.3 (Occurrences préservées) *Soit une dérivation* A: $M_0 \xrightarrow{U_1} M_1 \xrightarrow{U_2} \ldots \xrightarrow{U_n} M_n$. *Soit une occurrence* u *dans* M_0. *Par définition, l'occurrence* u *est préservée par la dérivation* A *et on note* $u \in id(A)$, *si et seulement si on a :*

$$\forall i \in [1 \ldots n]\ \forall v \in U_i,\ \ u \prec v \ \ ou\ u \mid v$$

Autrement dit une occurrence u préservée par une dérivation A est une occurrence préservée par toutes les étapes élémentaires de A. Par conséquent, on a l'égalité $u \setminus A = \{u\}$ et tous les sous-termes M_i/u ont la même tête (lemme 1.2.10). Si l'occurrence u est l'occurrence dans le terme initial M_0 d'un radical R de nom α, alors on parle de radical préservé et on note $u \in \mathcal{I}d(A)$. Les radicaux extérieurs sont des radicaux préservés puis contractés.

Définition 4.3.4 (Radicaux extérieurs) *Soit une dérivation* A: $M_0 \xrightarrow{U_1} M_1 \xrightarrow{U_2} \ldots \xrightarrow{U_n} M_n$. *Soit* u *une occurrence d'un radical* R *dans* M_0. *Par définition,* R *est un radical extérieur pour la dérivation* A *et on note* $u \in \mathcal{E}xt(A)$, *si et seulement si il existe un entier* i *dans l'intervalle* $[1 \ldots n]$, *tel que les deux propositions suivantes sont satisfaites :*

1. *Le radical* R *est préservé par la dérivation* $A[1, i-1]$
2. *Le radical résidu de* R *par la dérivation* $A[1, i-1]$, *est parmi les plus extérieurs de ceux qui sont contractés par la réduction* A_i ($u \in inf_{\preceq}(U_i) = \mathcal{E}xt(A_i)$).

Cette définition est tout simplement l'extension aux dérivations de la définition des radicaux les plus extérieurs contractés par une réduction. Dans le cas où A est une réduction, on a l'égalité $\mathcal{E}xt(A) = inf_{\preceq}(\mathcal{R}(A))$ (définition 1.2.14). Dans le cas où A est une dérivation, seule l'inclusion $\mathcal{E}xt(A) \subseteq \mathcal{R}(A)$ est toujours vérifiée.

Trois lemmes faciles donnent les premières propriétés des radicaux extérieurs. Tout d'abord, les radicaux extérieurs pour une dérivation n'ont pas de résidu par cette dérivation.

Lemme 4.3.5 *Pour toute dérivation* A, *on a l'égalité* $\mathcal{E}xt(A) \setminus A = \emptyset$.

Démonstration : Soit u une occurrence de radical dans $\mathcal{E}xt(A)$. En reprenant les notations de la définition de $\mathcal{E}xt(A)$, on a :

$$u \setminus A = ((u \setminus (A[1, i-1])) \setminus A_i) \setminus A[i+1, n] = (\{u\} \setminus A_i) \setminus A[i+1, n] = \emptyset \setminus A[i+1, n] = \emptyset$$

□

Lemme 4.3.6 *Soit une dérivation AB. Alors, on a l'inclusion $\mathcal{E}xt(A) \subseteq \mathcal{E}xt(AB)$. Si en outre, la dérivation A a au moins une étape élémentaire de type (red), alors on a l'égalité $\mathcal{E}xt(A) = \mathcal{E}xt(AB)$.*

Démonstration : Il s'agit de conséquences immédiates de la définition 4.3.4. □

Lemme 4.3.7 *On a l'équivalence suivante, valable pour toute dérivation A :*

$$A \equiv O \Leftrightarrow \mathcal{E}xt(A) = \emptyset$$

Démonstration: Seul le sens réciproque $\Leftarrow$ n'est pas évident. Soit une dérivation A, telle que l'on a $\mathcal{E}xt(A) = \emptyset$.

1. Si la dérivation A est une réduction A: $M \xrightarrow{U} N$, alors on a $inf_{\preceq}(U) = \emptyset$, et donc, nécessairement, $U = \emptyset$.
2. Sinon, on décompose la dérivation A en deux étapes $A = BC$. Par le lemme 4.3.6, on a $\mathcal{E}xt(B) = \emptyset$, et donc, par induction, $B = \emptyset^{|B|}$. On remplace alors la dérivation B par sa valeur dans l'égalité $\mathcal{E}xt(BC) = \emptyset$, d'où l'on tire facilement $\mathcal{E}xt(C) = \emptyset$, puis $C = \emptyset^{|C|}$ par induction. □

4.3.2 Décomposition des dérivations

Soit une réduction A: $M \to N$. On dit que A est une *réduction interne*, quand A ne contracte pas de radical à la racine de M. C'est à dire que A est une réduction interne, si et seulement si A est du type (rec) ou du type (id). Une *dérivation interne* est une dérivation dont toutes les étapes élémentaires sont des réductions internes. C'est à dire, qu'étant donnée une décomposition en étapes élémentaires de la forme :

$$A\colon M = M_0 \xrightarrow{U_1} M_1 \xrightarrow{U_2} \dots \xrightarrow{U_n} M_n = N$$

La dérivation A est interne, si et seulement si l'occurrence ϵ n'est pas élément des ensembles de radicaux contractés U_i, pour i variant de 1 à n. On note parfois A: $M {\to_{\mathrm{int}}}^* N$ une dérivation interne. Dans le cas où le terme M est de la forme $f^l(P_1, \dots, P_a)$, alors on dit que la dérivation A est *interne et décomposable*. Dans ce cas, toutes les étapes élémentaires de la dérivation A sont des réductions de type (rec) et on prouve facilement par induction sur la longueur de A que le terme N est de la forme $f^l(Q_1, \dots, Q_a)$. En outre, on peut également décomposer les termes intermédiaires M_i et les ensembles U_i de la façon suivante :

$$\forall i \in [0 \dots n],\ M_i = f^l(M_{(i,1)}, M_{(i,2)}, \dots, M_{(i,a)}) \qquad \forall i \in [1 \dots n],\ U_i = (\bigcup_{j \in \mathrm{dom}(f)} j.U_{(i,j)})$$

Et donc, pour tout indice j de $\mathrm{dom}(f)$, il existe une dérivation A/j définie par :

$$A/j\colon M/j = M_{(0,j)} \xrightarrow{U_{(1,j)}} M_{(1,j)} \xrightarrow{U_{(2,j)}} \dots \xrightarrow{U_{(n,j)}} M_{(n,j)} = N/j$$

Tandis que, pour tout indice j interdit à partir du symbole f, on a l'identité A/j: $M/j = N/j$. Les dérivations (et les identités) A/j forment les *dérivations arguments* de la dérivation A.

L'idée de la *décomposition canonique* est d'itérer la décomposition en arguments.

Définition 4.3.8 (Décomposition canonique) *Soit une dérivation A: $M \to^* N$. La décomposition canonique de la dérivation A, notée $\mathcal{C}(A)$, est un ensemble de couples formés chacun d'une occurrence et d'une dérivation. L'ensemble $\mathcal{C}(A)$ est défini de la façon suivante :*

1. *Si le terme M est une variable, alors on pose $\mathcal{C}(A) = \emptyset$.*
2. *Si la dérivation A possède au moins une étape élémentaire de type (red), alors on pose :*

$$\mathcal{C}(A) = \{(\epsilon, A\colon M \to^* N)\}$$

3. *Sinon, la dérivation A est interne et décomposable, avec $M = f^l(M_1, \ldots, M_a)$. On peut donc procéder à la décomposition en arguments de la dérivation A. Ensuite, pour chaque indice i autorisé à partir du symbole f, on procède à la décomposition canonique de la dérivation argument A/i et on pose :*

$$\mathcal{C}(A) = \bigcup_{i \in \mathrm{dom}(f)} \{\, (i.u, B\colon P \to^* Q) \mid (u, B\colon P \to^* Q) \in \mathcal{C}(A/i) \,\}$$

Les premières composantes des couples de la décomposition canonique de la dérivation A forment l'ensemble des *occurrences canoniques* pour la dérivation A, qui est noté $c(A)$.

Lemme 4.3.9 *On montre l'équivalence suivante par induction sur le terme M :*

$$u \in c(A) \quad \Leftrightarrow \quad \left\{ \begin{array}{l} \forall i \in [1 \ldots n] \ \forall v \in U_i,\ u \preceq v \ \textit{ou} \ u \mid v \\ \quad \textit{et} \\ \exists i \in [1 \ldots n],\ u \in U_i \end{array} \right.$$

Par conséquent, quelles que soient les dérivations A et B concaténables, on a la caractérisation inductive suivante des occurrences canoniques :

$$u \in c(AB) \Leftrightarrow \left\{ \begin{array}{l} u \in c(A) \text{ et } u \in c(B) \\ \quad \text{ou} \\ u \in c(A) \text{ et } u \in id(B) \\ \quad \text{ou} \\ u \in id(A) \text{ et } u \in c(B) \end{array} \right.$$

Exemple : Soient les règles $I(x) \to x$ et $A \to B$. Et soient les dérivations A_1 et A_2 (on oublie les étiquettes par souci de simplicité).

$$A_1\colon F(C, I(A)) \overset{\{2.1\}}{\to} F(C, I(B)) \overset{\{2\}}{\to} F(C, B) \qquad A_2\colon F(C, I(A)) \overset{\{2, 2.1\}}{\to} F(C, B)$$

Les décompositions canoniques de A_1 et de A_2 sont respectivement :

$$\{(2, I(A) \overset{\{1\}}{\to} I(B) \overset{\{\epsilon\}}{\to} B)\} \qquad\qquad \{(2, I(A) \overset{\{\epsilon, 1\}}{\to} B)\}$$

Soit une dérivation $A\colon M_0 \overset{U_1}{\to} M_1 \overset{U_2}{\to} \ldots \overset{U_n}{\to} M_n$, et soit u une occurrence canonique pour la dérivation A. On note A/u la dérivation qui est associée à l'occurrence u dans la décomposition canonique $\mathcal{C}(A)$, ou dérivation argument. Pour tout ensemble d'occurrences U, on note U/u l'ensemble défini par :

$$U/u = \{\, v \mid u.v \in U \,\}$$

On montre facilement par induction sur M, que la dérivation A/u est la dérivation décrite par la décomposition en étapes élémentaires :

$$A/u\colon M_0/u \overset{U_1/u}{\to} M_1/u \overset{U_2/u}{\to} \ldots \overset{U_n/u}{\to} M_n/u$$

Ce type de dérivation argument A/u comprend toujours une étape élémentaire de type (red) (cf. le cas de base *2* de la définition constructive de la décomposition canonique). On montre aussi, toujours par la même méthode, l'égalité suivante :

$$\forall i \in \ [1 \dots n],\ U_i = \bigcup_{u \in C(A)} u.(U_i/u)$$

Cette égalité exprime la complétude de la décomposition canonique. Elle entraîne :

Corollaire 4.3.10 *Soit une dérivation A: $M \to^* N$. On a les équivalences :*

1. *La dérivation A est nulle, si et seulement si l'ensemble $c(A)$ est vide.*
2. *La dérivation A possède au moins une étape de type (red), si et seulement si l'ensemble $c(A)$ est égal à $\{\epsilon\}$.*
3. *La dérivation A est interne et non-nulle dans tous les autres cas.*

Toujours par induction sur M et à partir des propriétés correspondantes des dérivations internes décomposables (c'est à dire, à partir des définitions correspondantes sur les réductions de type (rec)), on a les résultats récapitulatifs suivants :

Proposition 4.3.11 *Pour toute dérivation A: $M \to^* N$ et toute occurrence u canonique pour A, on a les égalités :*

$$(u.v) \setminus A = u.(v \setminus A/u) \qquad (u.v) \setminus\!\!\setminus A = u.(v \setminus\!\!\setminus A/u) \qquad (u.v) \setminus\!\!\setminus\!\!\setminus A = u.(v \setminus\!\!\setminus\!\!\setminus A/u)$$

En outre, toute occurrence v qui est un préfixe strict de l'occurrence u ($v \prec u$) est préservée par la la dérivation A ($v \in id(A)$). Enfin, on les égalités :

$$\mathcal{R}(A) = \bigcup_{u \in C(A)} u.\mathcal{R}(A/u) \qquad\qquad \mathcal{E}xt(A) = \bigcup_{u \in C(A)} u.\mathcal{E}xt(A/u)$$

Enfin, il existe au sujet des occurrences canonique, une propriété très semblable au lemme 2.4.3 d'existence du radical créateur.

Lemme 4.3.12 *Soit une dérivation interne A: $M_0 \xrightarrow{U_1} M_1 \xrightarrow{U_2} \dots \xrightarrow{U_n} M_n$, telle qu'il existe un radical de nom α à la racine du terme N. Alors, il y a deux possibilités mutuellement exclusives :*

1. *Il existe un radical à la racine dans le terme M et ce radical est nécessairement un radical de nom α préservé par la dérivation A.*
2. *Il n'existe pas de radical à la racine dans le terme M, alors il existe au moins une occurrence canonique pour la dérivation A qui est une occurrence dure du nom α.*

Démonstration : La proposition *1* est une conséquence directe du lemme 2.3.8 et de l'égalité $\epsilon \setminus A = \{\epsilon\}$, valide pour toute dérivation interne A. Si il n'existe pas de radical à la racine dans le terme M_0, alors α n'est pas un préfixe du terme M. Il existe donc au moins une occurrence u non-vide, qui est à la fois une occurrence de M_0 et une occurrence dure de α ($\alpha/u = f^l(\alpha_1, \dots, \alpha_a)$), avec :

$$M_0/u = x^m \quad \text{ou} \quad M_0/u = g^m(P_1, \dots, P_b),\ \text{avec} f^l \neq g^m$$

C'est à dire que les sous-termes M_0/u et M_n/u n'ont pas la même tête. Par le lemme 1.2.10, l'occurrence u ne peut pas être préservée par toutes les étapes élémentaires de la dérivation A. Il existe donc un entier i compris entre 1 et n, ainsi qu'une occurrence v de l'ensemble U_i qui est un préfixe de l'occurrence u. Soit, par la décomposition $U_i = \bigcup_{w \in \mathcal{C}(A)} w.(U_i/w)$, il existe une occurrence canonique w, avec $w \preceq v \preceq u$. □

Soit maintenant une dérivation de la forme AB, où A est une dérivation interne et B est une dérivation qui commence par une étape de type (red). On cherche à standardiser la dérivation AB, c'est à dire à permuter ses étapes de façon à contracter d'abord les radicaux les plus extérieurs. Si on se trouve dans le premier cas du lemme 4.3.12, on va donc contracter en premier le radical à la racine présent dans le terme M, qui n'est préservé par la dérivation A que pour être contracté plus tard. Sinon, le deuxième cas du lemme 4.3.12 donne l'amorce d'une procédure de recherche d'un radical à contracter en premier. En effet, il existe alors au moins une dérivation argument A/u dont les étapes de type (red) "participent" à la création du radical à la racine contracté par la première étape élémentaire de B. Il reste ensuite à décomposer la dérivation A/u sous forme préfixe-suffixe, afin d'isoler sa première étape de type (red), puis à réappliquer l'un ou l'autre cas du lemme 4.3.12 au préfixe interne de A/u ainsi dégagé.

4.3.3 Radicaux standards

La définition des radicaux standards suit l'intuition développée à la fin de la section précédente.

Définition 4.3.13 *Soit une dérivation A: $M \to^* N$. On définit l'ensemble $\mathcal{E}(A)$ des radicaux standards pour la dérivation A de la façon suivante :*

1. *Si le terme M est une variable, alors on pose $\mathcal{E}(A) = \emptyset$.*
2. *Si la dérivation A possède au moins une étape élémentaire de type (red) alors on décompose A en la concaténation BCD, avec :*

$$B\colon M \to_{\text{int}}{}^* P, \quad C\colon P = l.\sigma(\alpha) \overset{\{\epsilon\}\cup\dots}{\longrightarrow} l.\rho(\beta) = Q \quad et \quad D\colon Q \to^* N$$

   (a) *Si il existe un radical à la racine dans le terme M, alors on pose : $\mathcal{E}(A) = \{\epsilon\}$.*

   (b) *Sinon, il existe des occurrences canoniques pour la dérivation B qui sont aussi des occurrences dures du nom α et on pose :*

$$\mathcal{E}(A) = \bigcup_{u \in \mathcal{C}(B) \cap \overline{\mathcal{O}}(\alpha)} u.\mathcal{E}(B/u)$$

3. *Enfin, si la dérivation A est interne et décomposable, alors on pose :*

$$\mathcal{E}(A) = \bigcup_{u \in \mathcal{C}(A)} u.\mathcal{E}(A/u)$$

Par construction, il est immédiat que toutes les occurrences de $\mathcal{E}(A)$ sont des occurrences de radicaux du le terme M ($\mathcal{E}(A) \subseteq \mathcal{R}(M)$).

Exemple : On reprend les règles $\{F(x, B) \to G(x, x), I(x) \to x, A \to B, C \to D\}$ et les dérivations A_1 et A_2 :

$$A_1\colon F(A, I(A)) \overset{\{2.1\}}{\to} F(A, I(B)) \overset{\{2\}}{\to} F(A, B) \qquad A_2\colon F(A, I(A)) \overset{\{2,2.1\}}{\to} F(A, B)$$

Par les étapes 3 et 2-(a) de la définition des radicaux standards, on montre facilement $\mathcal{E}(A_1) = \mathcal{E}(A_2) = \{2\}$. Considérons maintenant la dérivation A_3.

$$A_3\colon F(C, I(A)) \overset{\{1,2,2.1\}}{\to} F(D, B) \overset{\{\epsilon\}}{\to} G(D, D).$$

On se trouve dans le cas 2-(b) de la définition des radicaux standards. C'est à dire que l'on décompose A_3 en $B_3C_3D_3$, avec :

$$B_3\colon F(C, I(A)) \xrightarrow{\{1,2,2,1\}} F(D, B) \qquad C_3\colon F(D, B) \xrightarrow{\{\epsilon\}} G(D, D) \qquad D_3 = O$$

La décomposition canonique de B_3 est :

$$\{\,(1, B_3/1\colon C \xrightarrow{\{\epsilon\}} D),\ (2, B_3/2\colon I(A) \xrightarrow{\{\epsilon,1\}} B)\,\}$$

L'occurrence 2 est la seule des occurrences canoniques qui est dure dans le nom $F(x, B)$ du premier radical à la racine contracté par A_3. On a donc $\mathcal{E}(A_3) = 2.\mathcal{E}(B_3/2) = 2.\{\epsilon\} = 2$.

La décomposition selon les occurrences canoniques —c'est à dire le procédé de décomposition des dérivations de la racine (de l'extérieur) des termes vers leurs feuilles (leur intérieur)— fournit bien des radicaux extérieurs.

Lemme 4.3.14 *Pour toute dérivation $A\colon M \to^* N$, on a l'inclusion :*

$$\mathcal{E}(A) \subseteq \mathcal{E}xt(A)$$

Démonstration: On reprend les cas de la définition inductive $\mathcal{E}(A)$.

1. Si le terme M est une variable, alors on a $\mathcal{E}(A) = \mathcal{E}xt(A) = \emptyset$.
2. Si la dérivation A se décompose en $A = BCD$, où B est une dérivation interne, C une réduction de type (red) et D une dérivation quelconque.
   (a) Si il existe un radical à la racine dans le terme M, alors on a $\mathcal{E}(A) = \{\epsilon\} \subseteq \mathcal{E}xt(A)$. (le radical à la racine, quand il est contracté, est toujours extérieur)
   (b) Sinon, par les propriétés 4.3.11 et la définition de $\mathcal{E}(A)$, on a les deux égalités :
   $$\mathcal{E}xt(B) = \bigcup_{u \in \mathcal{C}(B)} u.\mathcal{E}xt(B/u) \qquad \mathcal{E}(A) = \bigcup_{u \in \mathcal{C}(B) \cap \overline{\mathcal{O}}(\alpha)} u.\mathcal{E}(B/u)$$
   D'où l'on tire directement $\mathcal{E}(A) \subseteq \mathcal{E}xt(B)$ par induction appliquée aux dérivations B/u. Soit finalement, $\mathcal{E}(A) \subseteq \mathcal{E}xt(A)$ par le lemme 4.3.6.
3. Si la dérivation A est interne et décomposable, alors on procède comme dans le cas 2.-(b). □

Les dérivations non-nulles admettent au moins un radical standard.

Lemme 4.3.15 *Pour toute dérivation $A\colon M_0 \xrightarrow{U_1} M_1 \xrightarrow{U_2} \dots \xrightarrow{U_n} M_n$, on a l'équivalence :*

$$A \equiv O \Leftrightarrow \mathcal{E}(A) = \emptyset$$

Démonstration: Seul le sens réciproque $\Leftarrow$ n'est pas une conséquence directe de la définition. Soit donc une dérivation A, dont une étape au moins n'est pas vide.

2. Si une des étapes élémentaires de la dérivation A est du type (red), alors on procède à la décomposition $A = BCD$, où la réduction $C\colon P \to Q$ est la première étape (red) de A. On retrouve ensuite les deux cas de la définition de radicaux standards.
   (a) Si le radical contracté par C est le résidu par B du radical à la racine présent dans le terme initial M, alors $\mathcal{E}(A) = \{\epsilon\}$ n'est pas vide.

(b) Sinon, il existe au moins une occurrence u qui est canonique pour B et dure dans le nom α du radical à la racine contracté par la réduction C (lemme 4.3.12). La dérivation B/u, qui est une dérivation de la décomposition canonique de B, possède au moins une étape élémentaire du type (red). A fortiori, la dérivation B/u n'est pas nulle. Par induction, l'ensemble $\mathcal{E}(B/u)$ n'est pas vide et donc, l'ensemble $\mathcal{E}(A)$ qui contient $u.\mathcal{E}(B/u)$ ne l'est pas non plus.

3. Si la dérivation A est interne et non-nulle, alors (corollaire 4.3.10) il existe au moins une occurrence canonique u pour la dérivation A et on peut conclure comme dans le cas 2.-(b). □

Cette preuve met en lumière le fait que l'étape *3* de la définition des radicaux standards n'est pas utilisée inductivement. En effet, il n'y a induction que sur des dérivations qui font partie d'une décomposition canonique et on sait que de telles dérivations ne sont jamais des dérivations internes.

On retrouve ensuite une difficulté liée à l'absence d'une définition des radicaux créateurs par delà les dérivation qui soit compatible avec l'équivalence des dérivations. En effet, il est tentant de chercher à caractériser les radicaux standards pour une dérivation A, comme les plus extérieurs parmi les créateurs des radicaux à la racine contractés par la première étape (red) de certaines des dérivations A/u de la décomposition canonique de A. Faute de pouvoir traduire cette intuition en termes de radicaux uniquement, on peut tout même énoncer un lemme qui en reprend l'esprit, mais utilise les occurrences de sous-termes quelconques.

Lemme 4.3.16 *Soit une dérivation* A: $M{\to}^*N$. *Et soit* u *une occurrence de radical de* $\mathcal{E}(A)$. *Alors, on a* $u\backslash\!\backslash A = u\backslash\!\backslash\!\backslash A = \{v\}$, *où* v *est une occurrence canonique pour la dérivation* A.

Démonstration: Par le lemme 4.3.14, on a immédiatement $u \setminus A = \emptyset$, soit $u\backslash\!\backslash A = u\backslash\!\backslash\!\backslash A$. Il suffit donc de montrer l'égalité $u\backslash\!\backslash A = \{v\}$, où v est une occurrence canonique de A. On reprend encore une fois la définition de $\mathcal{E}(A)$ pour A non-nulle.

2. Si la dérivation A n'est pas interne, alors on procède à la décomposition habituelle $A = BCD$.

$$B\colon M{\to_{\text{int}}}^*P, \quad C\colon P = l.\sigma(\alpha) \stackrel{\{\epsilon\}\cup\cdots}{\longrightarrow} l.\rho(\beta) = Q \quad \text{et} \quad D\colon Q{\to}^*N$$

(a) Si on a $\mathcal{E}(A) = \{\epsilon\}$, parce que le radical à la racine contracté à l'étape C est déjà présent dans le terme M, alors on a $\epsilon\backslash\!\backslash B = \epsilon \setminus B = \{\epsilon\}$ —par hypothèse—, $\epsilon\backslash\!\backslash C = \epsilon\backslash\!\backslash\!\backslash C = \{\epsilon\}$ —par définition des résidus et des héritiers— et $\epsilon\backslash\!\backslash C = \{\epsilon\}$ —car on ne peut pas effacer l'occurrence à la racine—. Soit finalement, $\epsilon\backslash\!\backslash A = \{\epsilon\}$.

(b) Sinon, toute occurrence u de $\mathcal{E}(A)$ est de la forme $v.w$, où v est une occurrence canonique pour la dérivation B, tandis que w est une occurrence de $\mathcal{E}(B/v)$. Puisque la dérivation B/v comporte au moins une étape de type (red), elle admet la seule occurrence vide ϵ comme occurrence canonique. Par induction, on a donc $w\backslash\!\backslash(B/v) = \{\epsilon\}$. Soit, par la proposition 4.3.11, $u\backslash\!\backslash B = v$. Or, l'occurrence canonique v est aussi une occurrence dure du nom α du radical à la racine contracté par la réduction C, soit, par définition des descendants, $v\backslash\!\backslash C = \{\epsilon\}$. En outre, on a $\epsilon\backslash\!\backslash D = \{\epsilon\}$. Soit finalement, $u\backslash\!\backslash A = ((u\backslash\!\backslash B)\backslash\!\backslash C)\backslash\!\backslash D = \{\epsilon\}$, l'occurrence ϵ étant la seule occurrence canonique de la dérivation A.

3. Si A est interne, alors toute occurrence u de $\mathcal{E}(A)$ est de la forme $u = v.w$, où v est une occurrence canonique pour la dérivation A et w est une occurrence de $\mathcal{E}(A/v)$. Ici comme dans le cas 2.-(b) ci-dessus, on a $w\backslash\!\backslash(A/v) = \{\epsilon\}$ par hypothèse d'induction, puis $u\backslash\!\backslash A = \{v\}$. □

```
let droite = function
   <α,a>     →  <α,a>
 | l.<α,a>  →  <α,a>
```

Figure 4.6 : Fonction pour accéder à l'étiquette élémentaire la plus à droite.

Nous disposons d'une double approche des "traces" que laissent les radicaux contractés dans les termes. Le lemme que nous venons de voir règle le problème du point de vue des héritiers, il nous reste à examiner le point de vue des étiquettes complètes. Nous allons voir que la trace laissée par la contraction des radicaux standards est si précise, qu'elle permet de retrouver ces radicaux.

4.3.4 Caractérisation des radicaux standards

Le but de cette section est de montrer que la connaissance du terme de départ et du terme d'arrivée d'une dérivation A complètement étiquetée suffit pour calculer l'ensemble des radicaux standards pour la dérivation A.

Par souci de précision, ce calcul est décrit comme un programme CAML. On suppose connus le type `atomique` des étiquettes alphabétiques (ensemble $\mathcal{A}$), le type `variable` des noms de variables (ensemble $\mathcal{X}$), ainsi que le type `symbole` des symboles fonctionnels (ensemble $\mathcal{F}$). On peut ensuite construire les types `elementaire` des étiquettes élémentaires, `complet` des étiquettes complètes et `terme` des termes complètement étiquetés :

```
type elementaire = Alpha of atomique | Catch of terme & atomique
and  complet = Vide | Elem of elementaire | Concat of complet & elementaire
and  terme = Var of complet & variable | App of symbole & complet & terme list
```

Afin de ne pas trop surcharger les notations, les syntaxes déjà définies sont conservées. Par exemple, on continue à écrire $<\alpha_1,a_1>.\mathtt{a_2}$ au lieu de

$$\mathtt{Concat(Elem(Catch(}\alpha_1,\mathtt{a_1})),\mathtt{Alpha(a_2))}$$

De même, on continue à écrire $\mathtt{f^1(M_1,M_2,\ldots,M_a)}$, pour le terme qui s'écrit $\mathtt{App(f,1,L_M)}$ en CAML traditionnel. La liste $\mathtt{L_M}$ est la liste des a arguments du terme précédent que l'on note $[\mathtt{M_1};\mathtt{M_2};\ldots;\mathtt{M_a}]$ ou $\mathtt{M_1}::\mathtt{M_2}::\ldots::\mathtt{M_a}::[]$ en CAML traditionnel et $(M_1,M_2,\ldots,M_a)$ dans la syntaxe étendue. Notons également que la syntaxe CAML des étiquettes complètes permet d'accéder rapidement à l'étiquette élémentaire la plus à droite dans une étiquette non-vide (cf. la figure 4.6). Enfin, les occurrences sont des listes d'entiers, l'occurrence vide ϵ étant la liste vide [].

Lemme 4.3.17 *Soit une dérivation A: $M \rightarrow^* N$. Alors on a l'égalité $\tau(M)=\tau(N)$, si et seulement si la dérivation A est interne.*

Démonstration : Il suffit de considérer la décomposition A: $M_0 \stackrel{U_1}{\rightarrow} M_1 \stackrel{U_2}{\rightarrow} \ldots \stackrel{U_n}{\rightarrow} M_n$ de A. Pour tout entier i compris entre 1 et n, si l'étape élémentaire A_i est de type (red) on a $\tau(M_{i-1}) = l.\tau(\alpha_i)$ et $\tau(M_i) = l.<\alpha_i,a_i>$, où α_i est le nom du radical à la racine contracté par A_i. Sinon, l'étape élémentaire A_i est interne et on a $\tau(M_{i-1}) = \tau(M_i)$. Il est clair, qu'à chaque étape (red), la taille de l'étiquette élémentaire la plus à droite de $\tau(M_i)$ ne fait que croître. Par conséquent, si A n'est pas interne, les étiquettes $\tau(M_0)$ et $\tau(M_n)$ diffèrent par leur étiquette élémentaire la plus à droite. □

```
let rec noms l (e0,e) =
  if e0 = e then l
  else
    match e with
      <α,_> → noms [] (α::l) (e0,τ(α))
;;

let premier (eM,eN) = match noms (eM,eN) with α::_ → α
;;
```

Figure 4.7 : Noms des radicaux à la racine contractés par les étapes de A: $M\to^* N$.

En fait, on peut faire mieux. Étant donnée une dérivation A: $M\to^* N$, la connaissance des étiquettes élémentaires les plus à droite dans $\tau(M)$ et $\tau(N)$ suffit pour donner les noms des radicaux à la racine contractés par les étapes élémentaires de A. Les étiquettes $\tau(M)$ et $\tau(N)$ se décomposent comme $\tau(M) = l.e_M$ et $\tau(N) = l.e_N$, où l est une étiquette éventuellement vide. L'appel de fonction `noms` [] ($\mathtt{e_M}, \mathtt{e_N}$) (figure 4.7) donne la liste des noms des radicaux à la racine contractés par les étapes élémentaire de la dérivation A. Le premier argument `l` de la fonction `noms` a un rôle technique, il sert à accumuler les noms de radicaux pour les restituer dans l'ordre dans lequel ils sont contractés. Notons que `noms` [] ($\mathtt{e_M}, \mathtt{e_N}$) renvoie la liste vide [] en résultat, si et seulement si la dérivation A est interne. Si la dérivation A n'est pas interne, alors on peut utiliser l'appel de fonction fonction `premier` ($\mathtt{e_M}, \mathtt{e_N}$) pour obtenir le nom du radical à la racine contracté par la première étape (red) de la dérivation A.

```
let rec canonique u (M,N) = match (M,N) with
  x^l,y^m                              → []
| _,y^m                                → [(u,(M,N))]
| f^l(M1,M2,...,Ma),g^m(N1,N2,...,Na) →
  if l=m then
    canonique_args u 1 ((M1,M2,...,Ma),(N1,N2,...,Na))
  else
    [(u,M,N)]

and canonique_args u i = function
  ([],[])        → []
| (M::lM,N::lN)  → canonique (u.i) (M,N)@canonique_args u (i+1) (lM,lN)
```

Figure 4.8 : La fonction `canonique`

Le test du lemme 4.3.17 est également utile pour calculer la décomposition canonique d'une dérivation A: $M\to^* N$, donnée sous la forme de la paire (`M,N`) de son terme de départ et de son terme d'arrivée. La figure 4.8 donne la fonction CAML `canonique` qui réalise ce calcul. L'appel de fonction `canonique` ϵ (`M,N`) renvoie la décomposition canonique de A, sous forme d'une liste de triplets (`u,M/u,N/u`), où u est une occurrence canonique pour la dérivation A, tandis que les sous-termes M/u et N/u sont les termes de départ et d'arrivée de la dérivation argument A/u associée à l'occurrence canonique u. Le premier argument de la fonction `canonique` a le rôle technique de mémoriser l'occurrence courante candidate pour être une occurrence canonique. Enfin, en CAML, la notation $\mathtt{l_1@l_2}$ désigne la concaténation des listes $\mathtt{l_1}$ et $\mathtt{l_2}$.

La structure la fonction `canonique` est exactement celle de la définition 4.8 de la décomposition canonique d'une dérivation $A\colon M{\to}^*N$. Sa correction repose donc entièrement sur le lemme 4.3.17. À savoir, que le test $\mathtt{l} = \mathtt{m}$, où $\mathtt{l} = \tau(\mathtt{M})$ et $\mathtt{m} = \tau(\mathtt{M})$ est vrai, si et seulement si la dérivation $(\mathtt{M},\mathtt{N})$ est interne.

Proposition 4.3.18 *Soient une dérivation $A\colon M{\to}^*N$. Par la correction de la fonction* `canonique` *et par le corollaire 4.3.10 on a les équivalences suivantes :*

1. *L'appel* `canonique u` $(\mathtt{M},\mathtt{N})$ *renvoie la liste vide, si et seulement si la dérivation A est nulle. On vérifie facilement que cette condition est équivalente à l'égalité des termes M et N.*
2. *L'appel* `canonique u` $(\mathtt{M},\mathtt{N})$ *renvoie une liste de la forme* $[(\mathtt{u},\_)]$*, si et seulement si la dérivation A a au moins une étape élémentaire de type (red).*
3. *L'appel* `canonique` ϵ $(\mathtt{M},\mathtt{N})$ *n'est ni de la forme 1 ni de la forme 2, si et seulement si la dérivation A est interne et non-nulle.*

```
let rec contribue_rec u (M,α) = match (M,α) with
  _,y                              →  []
| f^l(M_1,M_2,...,M_a),g^m(α_1,α_2,...,α_a) →
  if l=m then
    contribue_args u 1 ((M_1,M_2,...,M_a),(α_1,α_2,...,α_a)))
  else
    [(u,M,τ(α))]

and contribue_args u i = function
  ([],[])          →  []
| (M::l_M,α::l_α)  →
    contribue (u.i) (M,α)@contribue_args u (i+1) (l_M,l_α)
;;

let contribue u (f^{l.e}(M_1,M_2,...,M_a),α) = contribue_rec u (f^e(M_1,M_2,...,M_a),α)
;;
```

Figure 4.9 : La fonction `contribue`

On souhaite ensuite écrire une fonction CAML qui réalise la preuve constructive du lemme 4.3.12. Il s'agit de la fonction `contribue` de la figure 4.9. Cette fonction a exactement la structure de la preuve inductive la plus naturelle de la proposition suivante : "étant donnée une dérivation $A\colon M{\to}^*N$ et un préfixe linéaire α du terme N, alors il existe des occurrences canoniques pour la dérivation A qui sont aussi des occurrences dures du préfixe α, si et seulement si le terme linéaire α n'est pas un préfixe du terme M". La correction de la fonction `contribue_rec` par rapport à cette preuve repose uniquement sur le fait que le test $\mathtt{l} = \mathtt{m}$, où $\mathtt{l} = \tau(\mathtt{M})$ et $\mathtt{m} = \tau(\alpha) = \tau(\mathtt{N})$ est vrai, si et seulement si l'occurrence ϵ est canonique pour la dérivation A. Le premier argument `u` de la fonction `contribue_rec` est une occurrence qui a le même rôle technique que le premier argument de la fonction `canonique`. Enfin, la fonction `contribue` prépare le couple d'arguments $(\mathtt{M},\alpha)$ dans le cas où c'est un terme de la forme $\mathtt{l}.\alpha$ qui est candidat pour être un préfixe commun des termes `M` et `N`.

```
let rec standard_rec u (M,l) =
  let α = premier (droite (τ(M)),droite l)) in
  match contribue u (M,α) with
    []     →  [u]
  | args  →  standards args

and standards = function
  []          →  []
| (u,M,l)::l_d →  standard_rec u (M,l)@standards l_d
;;

let standard (M,N) = match canonique ϵ (M,N) with
  []      →  []
| [(ϵ,_)] →  standard_rec ϵ (M,τ(N))
| args    →  standards args
;;
```

Figure 4.10 : La fonction `standard`

Proposition 4.3.19 *Soient une dérivation interne A: $M{\to}^*N$ et un nom de radical α, tel que α est le nom du radical à la racine du terme N. Soit une occurrence u quelconque. Alors, on les deux équivalences suivantes :*

1. *L'appel* `contribue u (M,`α`)` *renvoie la liste vide [], si et seulement si il y a un radical à la racine dans le terme M.*
2. *Sinon, l'appel* `contribue u (M,`α`)` *renvoie une liste de triplets qui représente l'ensemble :*

$$\{\,(u.v, M/v, \tau(\alpha/v)) \mid v \in c(A) \cap \overline{\mathcal{O}}(\alpha)\,\}$$

En outre, puisque la dérivation A est interne, les occurrences v de $c(A)$ ne peuvent pas être l'occurrence vide ϵ.

Lemme 4.3.20 *Soit une dérivation A: $M{\to}^*N$. Alors on peut calculer l'ensemble $\mathcal{E}(A)$ des radicaux standards pour la dérivation A, à partir des termes M et N uniquement.*

Démonstration : Une preuve effective de ce lemme est donnée par la procédure décrite dans la figure 4.10. La fonction `standard` elle même ressemble suffisamment à la définition 4.3.13 des radicaux standard, pour que sa correction découle directement de celle des fonctions `canonique` et `standard_rec`. La correction de la fonction `canonique` a déjà été prouvée (cf. la proposition 4.3.18). La correction de la fonction `standard_rec` est la proposition suivante : pour toute dérivation A: $M{\to}^*N$ qui comprend au moins une étape de type (red) et toute occurrence u, l'appel `standard_rec u (M,`τ`(N))` renvoie l'ensemble $u.\mathcal{E}(A)$ sous la forme d'une liste d'occurrences. On procède par induction sur le terme M. Dans tous les cas, et puisque la dérivation A n'est pas interne, il existe une dérivation interne B: $M{\to}^*P$, une réduction de type (red) C: $P \to Q$ et une dérivation quelconque D: $Q{\to}^*N$ qui réalisent la décomposition $A = BCD$. De fait, l'appel de fonction `premier (droite (`τ`(M)),droite (`τ`(N)))` renvoie le nom α du radical à la racine contracté par la réduction C, qui est bien la première étape (red) de la dérivation A. La suite de la preuve s'appuie sur la correction de la fonction `contribue` (proposition 4.3.19).

- La cas de base correspond à la détection par la fonction `contribue` d'un radical à la racine du terme M. On a bien alors $u.\mathcal{E}(A) = u.\{\epsilon\} = u$.
- Sinon, il n'y a pas de radical à la racine dans le terme M et l'appel de fonction `contribue u` α, renvoie une liste de triplets qui est l'ensemble :

$$E = \{\, (u.v, M/v, \tau(\alpha/v)) \mid v \in c(B) \cap \overline{\mathcal{O}}(\alpha) \,\}$$

Pour toute occurrence v qui est à la fois canonique pour la dérivation B et dure dans le nom de radical α, on considère la dérivation B/v: $M/v \to^* P/v$. De par la correction de la fonction `contribue`, la dérivation B/v possède au moins une étape (red). La fonction `contribue` renvoie, en plus de l'occurrence v, le terme M/v ainsi que l'étiquette $\tau(\alpha/v) = \tau(P/v)$. On peut donc appliquer l'hypothèse d'induction et calculer l'ensemble $u.(v.\mathcal{E}(B/v))$ par l'appel de fonction `standard_rec (u.v) (M/v,`τ`(`α`/v))`. La fonction `standard_args` réalise ces appels récursifs pour tous les triplets de l'ensemble E et accumule les résultats, pour finalement calculer la liste d'occurrences qui représente l'ensemble :

$$\bigcup_{v \in CB \cap \overline{\mathcal{O}}(\alpha)} (u.v).\mathcal{E}(B/v) = u.\mathcal{E}(A)$$

La terminaison de la fonction `standard_rec` est certaine, car dans les appel récursifs qui sont de la forme générale `standard_rec (u.v) (M/v,`τ`(`α`/v))`, on a $v \succ \epsilon$, et donc, le sous-terme M/v est un sous-terme strict du terme M. Enfin, les appel initiaux de la fonction `standard_rec` sont bien formés, car il ont lieu, soit sur la dérivation A elle même quand elle possède au moins une étape élémentaire de type (red), soit sur les dérivations de la décomposition canonique de A qui sont toujours des dérivations non-internes. □

Corollaire 4.3.21 *Pour toutes dérivations complètement étiquetées A: $M \to^* N$ et B: $M \to^* N$, on a l'égalité $\mathcal{E}(A) = \mathcal{E}(B)$.*

Ce corollaire s'applique en particulier au cas où les dérivations A et B sont équivalentes. Dans la section suivante, nous allons voir comment le lemme 4.3.20 peut être utilisé pour montrer que les dérivations équivalentes sont caractérisées par leur termes de départ et d'arrivée uniquement.

4.3.5 Caractérisation des dérivations équivalentes

Soit une dérivation A: $M \to^* N$, pour montrer le résultat annoncé, on va construire une dérivation équivalente à A particulière, à partir des termes M et N uniquement. Cette dérivation particulière ne contracte que des radicaux standards pour elle-même.

Définition 4.3.22 *Une dérivation A: $M \to^* N$ est standard, si et seulement si on se trouve dans un des deux cas suivants :*

1. *La dérivation A est la dérivation vide O.*
2. *La dérivation A est de la forme $A = BC$, où B: $M \xrightarrow{U} P$ est une réduction telle que l'on a $U \subseteq \mathcal{E}(A)$, et C: $P \to^* N$ est une dérivation standard.*

À toute dérivation A: $M \to^* N$, on peut associer un *équivalent standard* qui est une dérivation B construite comme suit :

1. Si $\mathcal{E}(A) = \emptyset$ (et donc, $A \equiv O$), alors on pose $B = O$.

$$
B\left\{
\begin{array}{rcl}
M_0 & \xrightarrow{A_0 = A \quad *} & N \\
\mathcal{R}(B_1) \subseteq \mathcal{E}(A_0)\Big\downarrow & & \emptyset\Big\downarrow \\
M_1 & \xrightarrow{A_1 = A_0 \setminus B_1 \quad *} & N \\
\mathcal{R}(B_2) \subseteq \mathcal{E}(A_1)\Big\downarrow & & \emptyset\Big\downarrow \\
M_2 & \xrightarrow{A_2 = A_1 \setminus B_2 \quad *} & N \\
\Big| & & \Big| \\
\vdots & & \vdots \\
\Big\downarrow & & \Big\downarrow \\
M_{n-1} & \xrightarrow{A_{n-1} = A_{n-2} \setminus B_{n-1} *} & N \\
\mathcal{R}(B_n) \subseteq \mathcal{E}(A_{n-1})\Big\downarrow & & \emptyset\Big\downarrow \\
M_n & \xrightarrow[\mathcal{E}(A_n) = \emptyset \Rightarrow A_n = \emptyset^{|A|}]{A_n = A_{n-1} \setminus B_n *} & N
\end{array}
\right.
$$

Figure 4.11 : Construction de l'équivalent standard de la dérivation A

2. Sinon, on choisit un sous-ensemble non-vide U de radicaux pris dans $\mathcal{E}(A)$ et on définit ensuite B comme la dérivation composée de la première étape élémentaire $B_1 \colon M \overset{U}{\to} N$, suivie d'un équivalent standard de la dérivation $A \setminus B_1$.

La figure 4.11 illustre la construction de la n-ième et dernière étape élémentaire d'un équivalent standard de la dérivation A.

Tout d'abord, la construction d'un équivalent standard B termine toujours, car la dérivation B est bornée au sens de la définition 3.3.4. En effet, si on se place à l'étape i et que l'on considère l'occurrence u d'un radical contracté par la réduction B_i, alors il vient $u \in \mathcal{E}(A_i)$ et donc $u \in \mathcal{R}(A_i)$, par construction même de l'étape B_i. C'est à dire que le radical d'occurrence u dans M_{i-1} a le même nom qu'un radical R contracté par une des étapes élémentaires de la dérivation A_i. Or (cf. la figure 4.2), le radical R est résidu d'un radical contracté par l'étape élémentaire correspondante de la dérivation A, radical dont il conserve bien entendu le nom. Les noms des radicaux susceptibles d'être contractés par la dérivation B sont donc à choisir parmi les noms des radicaux contractés par les étapes élémentaires de la dérivation A. On appelle une dérivation qui obéit à cette contrainte, un *développement de la dérivation* A. Puisque les noms de radicaux contractés par les étapes élémentaires de la dérivation A finie forment un ensemble fini, il est clair que les développements de A sont des dérivations bornées. Elles sont donc nécessairement finies par la proposition 3.3.11.

Ensuite, à toute étape i de la construction de B, on a $\mathcal{R}(B_i) \subseteq \mathcal{E}(A_i)$ et donc (lemme 4.3.14), $\mathcal{R}(B_i) \subseteq \mathcal{E}xt(A_i)$. D'où il vient, par le lemme 4.3.5, l'égalité $B_i \setminus A_i = \emptyset$. Soit finalement, $B = B[1,n] \sqsubseteq A$. Si on se place maintenant à la dernière étape n de la construction de B, on a l'égalité $\mathcal{E}(A_n) = \emptyset$, d'où l'on tire $A_n = \emptyset^{|A|}$ par le lemme 4.3.15. Soit encore, $A \sqsubseteq B$. Par conséquent, les dérivations A et B sont équivalentes.

Pour justifier la terminologie "équivalent standard" il reste à vérifier que B est bien une dérivation standard au sens de la définition 4.3.13. Il suffit de remarquer que, pour tout entier i compris entre 1 et n, les dérivations $B[i,n]$ et A_i sont issues du même terme (M_{i-1}) et aboutissent au même terme (N). Soit, par le corollaire 4.3.21, les radicaux contractés par la réduction B_i sont pris dans $\mathcal{E}(B[i,n]) = \mathcal{E}(A_i)$.

Le résultat de la construction d'un équivalent standard dépend de la méthode de choix des radicaux à prendre dans $\mathcal{E}(A_i)$ à chaque étape. Je propose, parmi d'autres, deux méthodes systématiques de choix de ces radicaux :

1. Choisir le radical le plus à gauche dans l'ensemble $\mathcal{E}(A_i)$.
2. Contracter tous les radicaux de $\mathcal{E}(A_i)$ à la fois.

Il est clair que ces choix sont arbitraires. Par le premier de ces choix on obtient la dérivation *standard gauche* équivalente à la dérivation A, notée $\mathcal{G}(A)$; tandis que le second produit la *standard maximale* équivalente à la dérivation A, notée $\mathcal{M}(A)$. Imposer une de ces méthodes revient à rendre unique la dérivation standard dans une classe d'équivalence donnée. En effet, soient A et B deux dérivations équivalentes, alors on montre facilement, par induction sur la construction de l'équivalent standard, et à l'aide de l'implication $A \equiv B \Rightarrow \mathcal{E}(A) = \mathcal{E}(B)$ (corollaire 4.3.21), que l'on a $\mathcal{G}(A) = \mathcal{G}(B)$ par exemple.

Theorème 4.3.23 (Caractérisation des dérivations) *Quelles que soient les dérivations complètement étiquetées A: $M \to^* N$ et A: $M \to^* P$, issues d'un même terme M, on a l'équivalence logique :*

$$A \equiv B \Leftrightarrow N = P$$

Démonstration : Le sens direct a déjà été démontré. Pour montrer le sens réciproque, il suffit de considérer les deux dérivations gauches $\mathcal{G}(A)$ et $\mathcal{G}(B)$. Par construction d'un équivalent standard, on a $A \equiv \mathcal{G}(A)$ et $B \equiv \mathcal{G}(B)$. Or, par le lemme 4.3.20, appliqué à chaque étape de la construction des dérivations gauches, les dérivations $\mathcal{G}(A)$ et $\mathcal{G}(B)$ sont exactement identiques. Il reste à conclure par transitivité de la relation $\equiv$. $\square$

4.3.6 Définition alternative des radicaux standard

Dans cette section je montre que les radicaux standards sont exactement les radicaux "*outside-in*" de [Hue 79, définition 3.8]. On commence par deux lemmes techniques.

Lemme 4.3.24 *Soit une dérivation A: $M \to^* N$ interne et décomposable ($M = f^l(M_1, \ldots, M_a)$). Alors, pour tout indice i autorisé à partir du symbole f, on a les deux égalités suivantes :*

$$\mathcal{E}xt(A) = \bigcup_{i \in \mathrm{dom}(f)} i.\mathcal{E}xt(A/i) \qquad \mathcal{E}(A) = \bigcup_{i \in \mathrm{dom}(f)} i.\mathcal{E}(A/i)$$

Démonstration : La première égalité est une conséquence immédiate des définitions (cf. le début de la section 4.3.2). Pour prouver la seconde, on remarque tout d'abord que, par définition de la décomposition canonique (cas *3* de la définition 4.8), on a les égalités :

$$c(A) = \bigcup_{i \in \mathrm{dom}(f)} i.c(A/i) \qquad \forall i \in \mathrm{dom}(f)\ \forall u \in c(A/i),\ (A/i)/u = A/(i.u)$$

D'où, par le cas *3* de la définition des radicaux standards, il vient :

$$\mathcal{E}(A) = \bigcup_{u \in \mathcal{C}(A)} u.\mathcal{E}(A/u) = \bigcup_{i \in \mathrm{dom}(f)} i.(\bigcup_{u \in \mathcal{C}(A/i)} u.\mathcal{E}((A/i)/u)) = \bigcup_{i \in \mathrm{dom}(f)} i.\mathcal{E}(A/i)$$

$\square$

Lemme 4.3.25 *Soient deux dérivations A: $M \to^* N$ et B: $N \to^* P$. Soit un radical R d'occurrence u dans le terme M, tel que R est préservé par A et que son unique résidu par la dérivation A est standard pour la dérivation B. Alors, le radical R est standard pour la dérivation AB.*

Démonstration: On procède par induction selon l'ordre lexicographique sur la paire $<M, n_\epsilon(B)>$, où $n_\epsilon(B)$ est le nombre d'étapes élémentaires de type (red) de la dérivation B.

1. Si le terme M est une variable, alors il n'y pas de radicaux dans M.
2. Si la dérivation B n'est pas interne ($n_\epsilon(B) > 0$), alors on procède à la décomposition habituelle en posant $B = CDE$, où D: $Q \to R$ est la première étape élémentaire de type (red) de la dérivation B. Soit α le nom du radical à la racine contracté par la réduction D. Il y a ensuite deux sous-cas :

   (a) Si l'occurrence u est l'occurrence vide ϵ. Alors, on remarque que l'occurrence ϵ ne peut être introduite dans $\mathcal{E}(B)$ que par le cas *2.-(a)* de la définition des radicaux standards. C'est à dire qu'il existe un radical à la racine R' dans le terme N et que la dérivation interne C le préserve. Par hypothèse le radical R' du terme N est l'unique résidu du radical à la racine R présent dans le terme M et préservé par la dérivation A. Par conséquent, la dérivation AC préserve le radical à la racine R et la décomposition $AB = (AC)DE$ prouve que l'on a $\mathcal{E}(AB) = \{\epsilon\}$ par le cas *2.-(a)* de la définition des radicaux standards.

   (b) Sinon, on se trouve dans le cas *2.-(b)* de la définition de $\mathcal{E}(B)$. C'est à dire que l'on a $u = v.w$, où v est une occurrence qui est à la fois dure dans le nom α et canonique pour la dérivation interne C, tandis que w est une occurrence standard pour la dérivation C/w. Par conséquent (définition 4.3.13-3), on a $u \in \mathcal{E}(C)$. Par application de l'hypothèse d'induction aux dérivations A et C ($n_\epsilon(C) = 0 < n_\epsilon(B)$), on a donc $u \in \mathcal{E}(AC)$. Or, l'occurrence v est canonique pour la dérivation AC (lemme 4.3.9). Par ailleurs, la dérivation AC étant interne, on a $v \in \mathcal{E}((AC)/w)$, par définition 4.3.13 des radicaux standards. D'où, par le cas *2.-(b)* de la même définition appliqué à la décomposition $AB = (AC)DE$, il vient finalement $u \in \mathcal{E}(AB)$.

3. Si B est interne et décomposable, alors (définition 4.3.13-3) l'occurrence u est de la forme $v.w$, avec $v \in c(B)$ et $w \in \mathcal{E}(B/v)$. Par le lemme 4.3.24, il existe un indice i autorisé à partir du symbole f, tel que l'on a $v = i.v_i$. En outre, on a les appartenances $v_i.w \in \mathcal{E}xt(A/i)$, $v_i \in c(B/i)$ et $w \in \mathcal{E}((B/i)/v_i)$. Si l'occurrence canonique v_i n'est pas vide, alors la dérivation B/i est interne et on a $v_i.w \in \mathcal{E}(B/i)$ par définition des radicaux standards. Sinon, on a $v_i = \epsilon$ et directement $v_i.w = w \in \mathcal{E}((B/i)/\epsilon) = \mathcal{E}(B/i)$. Par induction, appliquée aux dérivations A/i et B/i, il vient $v_i.w \in \mathcal{E}(A/i)$. Soit finalement (lemme 4.3.24), $u = i.(v_i.w) \in \mathcal{E}(AB)$. □

Lemme 4.3.26 *Soit A: $M \to^* N$, une dérivation standard. Alors, on a l'égalité $\mathcal{E}xt(A) = \mathcal{E}(A)$.*

Démonstration : L'inclusion $\mathcal{E}(A) \subseteq \mathcal{E}xt(A)$ est vraie quelle que soit la dérivation A (lemme 4.3.14). Réciproquement, soit u une occurrence $\mathcal{E}xt(A)$, alors il existe une décomposition de la dérivation A de la forme $A = BCD$, où B est une dérivation qui préserve le radical d'occurrence u, tandis que C est une réduction qui le contracte. Puisque la dérivation A est standard, on a $u \in \mathcal{E}(CD)$. Par conséquent (lemme 4.3.25 ci-dessus), on a $u \in \mathcal{E}(BCD)$. □

Le lemme 3.22 de [Hue 79] est une propriété caractéristique des radicaux "*outside-in*". En montrant que les radicaux standards la satisfont, on montre donc l'équivalence des deux notions "*outside-in*" et standard.

Lemme 4.3.27 *Pour toute dérivation A, on a l'égalité :*

$$\mathcal{E}(A) = \bigcap_{B \equiv A} \mathcal{E}xt(B)$$

Démonstration : Pour toute dérivation B équivalente à la dérivation A, on a $\mathcal{E}(B) = \mathcal{E}(A)$ (corollaire 4.3.21). Par le lemme 4.3.14, on a aussi $\mathcal{E}(B) \subseteq \mathcal{E}xt(B)$. Par conséquent, on a l'inclusion $\mathcal{E}(A) \subseteq \bigcap_{B \equiv A} \mathcal{E}xt(B)$. Pour prouver que cette inclusion n'est pas stricte, il suffit de montrer qu'un au moins des ensembles $\mathcal{E}xt(B)$ est égal à $\mathcal{E}(B) = \mathcal{E}(A)$. Il suffit donc, par le lemme 4.3.26, de choisir pour B un équivalent standard de A, tel que construit à la section précédente. □

Cette propriété donne une vision très intuitive mais non-constructive de ce qu'est un radical standard. Un radical standard pour une dérivation A est un radical extérieur pour la dérivation A qui reste extérieur si on remplace A par un de ses équivalents.

4.4 Stratégies

La standardisation est un résultat extrêmement intéressant pour l'étude des classes de dérivations. En bref, elle permet de bien décrire l'ensemble des dérivations qui vont d'un terme M vers un autre terme N connu à l'avance. Dans la pratique, on se pose un autre type de problème. Étant donné un terme M, que peut-on dire des dérivations standards issues de ce terme ? Existe-t-il une une *stratégie*, c'est à dire une méthode pour choisir les radicaux à contracter, qui permet d'atteindre la forme normale de M si elle existe ? Plus généralement, comment calculer à partir du terme M, tout en étant certain d'atteindre un résultat ?

4.4.1 Radicaux standards à partir d'un terme

On veut tout d'abord définir une stratégie qui ne produira que des dérivations standards. Pour ce faire, on cherche des radicaux qui soient standards pour toutes les dérivations issues d'un terme et non plus seulement pour une classe d'équivalence de dérivations. Et, de fait, seule la première étape (red) d'une dérivations détermine l'ensemble de ses radicaux standards.

Lemme 4.4.1 *Soit une dérivation A: $M \rightarrow^* N$, qui n'est pas une dérivation interne. Alors, pour toute dérivation B: $N \rightarrow^* P$, on a l'égalité $\mathcal{E}(AB) = \mathcal{E}(A)$.*

Démonstration : Il s'agit d'une conséquence immédiate de la définition 4.3.13. Pour calculer $\mathcal{E}(A)$, on doit utiliser le deuxième cas de cette définition et décomposer la dérivation A en $A_1A_2A_3$, où la réduction A_2 est la première étape (red) de la dérivation A. Pour calculer $\mathcal{E}(AB)$, on a une décomposition très semblable, de la forme $AB = A_1A_2(A_3B)$. Dans les deux cas, les ensembles $\mathcal{E}(A)$ et $\mathcal{E}(AB)$ se calculent à l'aide de la dérivation interne A_1 uniquement. □

Dès lors, les radicaux standards sont indépendants du choix d'une dérivation non-interne particulière.

Lemme 4.4.2 *Soient un terme M et deux dérivations A: $M \rightarrow^* N$ et B: $M \rightarrow^* P$ qui ont toutes deux une étape élémentaire de type (red). Alors, on a l'égalité $\mathcal{E}(A) = \mathcal{E}(B)$.*

Démonstration : On considère les deux dérivations $A \sqcup B$ et $B \sqcup A$. Par le lemme 4.4.1, on a $\mathcal{E}(A) = \mathcal{E}(A \sqcup B)$ et $\mathcal{E}(B) = \mathcal{E}(B \sqcup A)$. Or, les dérivations $A \sqcup B$ et $B \sqcup A$ étant équivalentes, on a l'égalité $\mathcal{E}(A \sqcup B) = \mathcal{E}(B \sqcup A)$. □

Ce résultat motive une définition des radicaux standards à partir d'un terme donné.

Définition 4.4.3 *Soit un terme M. On définit l'ensemble $\mathcal{E}(M)$ des radicaux standards à partir du terme M de la manière suivante :*

1. *Si le terme M se réduit à une variable x^l, alors on pose $\mathcal{E}(M) = \emptyset$.*
2. *Si le terme M est de la forme $M = f^l(M_1, M_2, \ldots, M_a)$, alors on distingue deux cas :*
   (a) *Si toutes les dérivations issues de M sont internes, alors on pose :*
   $$\mathcal{E}(M) = \bigcup_{i \in \mathrm{dom}(f)} i.\mathcal{E}(M_i)$$
   (b) *Sinon, il existe une dérivation A: $M \to^* N$ qui n'est pas interne, et on pose :*
   $$\mathcal{E}(M) = \mathcal{E}(A)$$

Par définition, les radicaux standards à partir du terme M sont des radicaux pris parmi les radicaux initiaux du terme M. Soit l'inclusion $\mathcal{E}(M) \subseteq \mathcal{R}(M)$. On peut aller plus loin en prouvant qu'il existe au moins un radical standard à partir de tout terme qui n'est pas en forme normale.

Proposition 4.4.4 *Soit un terme M. Alors le terme M est en forme normale, si et seulement si l'ensemble $\mathcal{E}(M)$ est vide.*

Démonstration: Si le terme M est en forme normale, alors on a $\mathcal{R}(M) = \emptyset$ et donc $\mathcal{E}(M) = \emptyset$. Réciproquement, soit un terme M qui n'est pas en forme normale.

1. Si toutes les dérivations non-nulles issues de M sont internes, alors on a nécessairement $\epsilon \notin \mathcal{R}(M)$, $M = f^l(M_1, \ldots, M_a)$, et il existe un indice i autorisé à partir du symbole f, tel que l'on a $\mathcal{R}(M_i) \neq \emptyset$. Par induction il existe une occurrence u dans $\mathcal{E}(M_i)$. Par conséquent, l'ensemble $\mathcal{E}(M)$ qui contient au moins l'occurrence $i.u$ n'est pas vide.
2. Sinon, il existe une dérivation A: $M \to^* N$ qui n'est pas interne et on pose $\mathcal{E}(M) = \mathcal{E}(A)$. La dérivation A, dont une étape au moins contracte au moins le radical à la racine, est non-nulle et on a donc $\mathcal{E}(A) \neq \emptyset$ (lemme 4.3.15). □

Il existe une caractérisation alternative de l'ensemble $\mathcal{E}(M)$ qui est très semblable à celle de $\mathcal{E}(A)$.

Définition 4.4.5 *Soit une dérivation A: $M \to^* N$. Par définition, le radical initial R d'occurrence u dans le terme M est simplement extérieur pour la dérivation A, et on note $u \in \mathcal{E}xt'(A)$, si et seulement si le radical R est préservé par la dérivation A ou extérieur pour la dérivation A.*

$$\mathcal{E}xt'(A) = \mathcal{I}d(A) \cup \mathcal{E}xt(A)$$

En particulier, si A n'est pas une dérivation interne, alors il n'y a pas de radical initial qui est préservé par A, et on a l'égalité $\mathcal{E}xt'(A) = \mathcal{E}xt(A)$.

Lemme 4.4.6 *Soient deux dérivations équivalentes A: $M \to^* N$ et B: $M \to^* N$. Alors la dérivation A est interne, si et seulement si la dérivation B est interne.*

Démonstration : On prouve facilement ce résultat par induction sur la définition de l'équivalence de permutations (définition 4.2.9). Une autre technique est d'utiliser le lemme 4.3.17, les dérivations A et B étant internes, si et seulement si on a l'égalité $\tau(M) = \tau(N)$. □

Lemme 4.4.7 *Soit A: $M \to^* N$ une dérivation qui comprend au moins une étape élémentaire de type (red). Alors, on a l'égalité :*

$$\mathcal{E}(A) = \bigcap_{B \equiv A} \mathcal{E}xt\text{'}(B)$$

Démonstration : Par la caractérisation alternative de $\mathcal{E}(A)$, on a :

$$\mathcal{E}(A) = \bigcap_{B \equiv A} \mathcal{E}xt(B)$$

En outre par le lemme 4.4.6, toute dérivation B équivalente à la dérivation A n'est pas interne. Par conséquent, d'après la remarque qui suit la définition 4.4.5, on a ici $\mathcal{E}xt\text{'}(B) = \mathcal{E}xt(B)$ quelles que soient les dérivations B équivalentes à A. □

Lemme 4.4.8 *Soit un terme M. On a la caractérisation alternative de l'ensemble $\mathcal{E}(M)$.*

$$\mathcal{E}(M) = \bigcap_{A \in \mathcal{D}(M)} \mathcal{E}xt\text{'}(A)$$

Démonstration: On prouve cette égalité par induction sur la définition 4.4.3.

1. Si $M = x^l$, alors tous les ensembles en jeu sont vides.
2. Sinon, on a $M = f^l(M_1, M_2, \ldots, M_a)$.

   (a) Si les les dérivations de $\mathcal{D}(M)$ sont toutes internes, alors elles sont toutes de la forme A: $M \to^* N$, avec $N = f^l(N_1, \ldots, N_a)$. En outre, pour tout indice i du domaine de f, on peut considérer la dérivation argument A_i: $M_i \to^* N_i$. Par construction de la décomposition en arguments, il vient facilement les égalités $\mathcal{I}d(A) = \bigcup_{i \in \mathrm{dom}(f)} i.\mathcal{I}d(A_i)$ et $\mathcal{E}xt(A) = \bigcup_{i \in \mathrm{dom}(f)} i.\mathcal{E}xt(A_i)$. Soit finalement, par définition de $\mathcal{E}xt\text{'}(A)$, on a l'égalité $\mathcal{E}xt\text{'}(A) = \bigcup_{i \in \mathrm{dom}(f)} i.\mathcal{E}xt\text{'}(A_i)$ (1). Mieux, toute dérivation A: $M \to^* N$ étant interne, il y a une correspondance bijective entre l'ensemble $\mathcal{D}(M)$ et le produit cartésien $\mathcal{D}(M_{i_1}) \times \mathcal{D}(M_{i_2}) \times \ldots \times \mathcal{D}(M_{i_n})$ des ensembles des dérivations issues des fils autorisés de M (l'ensemble ordonné $\{i_1, i_2, \ldots\ i_n\}$ est le domaine de f). Soit finalement :

   $$\begin{array}{rcll} \mathcal{E}(M) & = & \bigcup_{i \in \mathrm{dom}(f)} i.\mathcal{E}(M_i) & \text{définition de } \mathcal{E}(M) \\ & = & \bigcup_{i \in \mathrm{dom}(f)} i.(\bigcap_{A_i \in \mathcal{D}(M_i)} \mathcal{E}xt\text{'}(A_i)) & \text{induction} \\ & = & \bigcap_{A \in \mathcal{D}(M)} (\bigcup_{i \in \mathrm{dom}(f)} i.\mathcal{E}xt\text{'}(A_i)) & \text{correspondance bijective ci-dessus} \\ & = & \bigcap_{A \in \mathcal{D}(M)} \mathcal{E}xt\text{'}(A) & \text{égalité (1)} \end{array}$$

   (b) Sinon, le résultat vient directement par le lemme 4.4.7. □

Un corollaire immédiat de ce lemme est l'inclusion $\mathcal{E}(M) \subseteq \mathcal{E}xt(M)$ qui vient en considérant la réduction A: $M \overset{\mathcal{E}xt(M)}{\longrightarrow} N$ pour laquelle on a l'égalité $\mathcal{E}xt\text{'}(A) = \mathcal{E}xt(M)$. Une autre conséquence de la caractérisation alternative des radicaux standards à partir du terme M est l'identification de l'ensemble $\mathcal{E}(M)$ aux "*external redexes in term M*" de [Hue 79, définition 3.23].

4.4.2 Radicaux nécessaires

La définition des radicaux standards à partir d'un terme a quelque chose de frustrant dans le sens qu'elle peut paraître exagérément contraignante. En effet, comme le montre le lemme 4.4.8 les radicaux standards à partir d'un terme M sont contractés par toutes les dérivations issues du terme M qui vont "suffisamment loin". Mais il est clair qu'il peut exister de tels radicaux qui ne sont pas parmi les radicaux initiaux les plus extérieurs.

Définition 4.4.9 *Soient un terme M et un radical R d'occurrence u dans M. Par définition, le radical R est nécessaire à partir du terme M et on note $u \in \mathcal{N}(M)$, si et seulement si pour toute dérivation A: $M \to^* N$, on a $u \in \mathcal{R}(A)$ ou $u \setminus A \neq \emptyset$.*

Dans cette définition, le cas $u \setminus A \neq \emptyset$ prend en compte les dérivations qui n'ont pas encore contracté un résidu du radical R. Le cas où le terme M admet une forme normale permet d'éclaircir ce point.

Lemme 4.4.10 *Soit un terme M normalisable et soit N la forme normale de M. Alors, les radicaux nécessaires sont exactement les radicaux contractés par toutes les dérivation normalisantes :*

$$\mathcal{N}(M) = \bigcap_{A:\, M \to^* N} \mathcal{R}(A)$$

Démonstration : Soient R un radical d'occurrence $u \in \mathcal{N}(M)$ et A: $M \to^* N$ une dérivation normalisante. Par définition de $\mathcal{N}(M)$, on a $u \in \mathcal{R}(A)$ ou bien le radical R a au moins un résidu par A dans le terme N. Or, il n'y a pas de radicaux dans la forme normale N, on a donc $u \in \mathcal{R}(A)$. Réciproquement, soit un radical R d'occurrence u dans M qui est contracté par toutes les dérivations normalisantes. Soit A: $M \to^* P$ une dérivation quelconque issue de M. Puisque le terme M est normalisable, il existe au moins une dérivation normalisante B: $M \to^* N$, avec $B \equiv A \sqcup B$ (lemme 4.2.13). La dérivation $A \sqcup B$ est normalisante, on a donc $u \in \mathcal{R}(A \sqcup B)$ par hypothèse. On conclut en disant que si le radical R n'est pas contracté par la dérivation A, alors un de ses résidus par A doit exister pour être contracté par la dérivation $B \setminus A$. $\square$

Comme on peut s'y attendre, les radicaux standards à partir d'un terme M sont un cas particulier de radicaux nécessaires.

Proposition 4.4.11 *Pour tout terme M, on a l'inclusion $\mathcal{E}(M) \subseteq \mathcal{N}(M)$.*

Démonstration : Par la caractérisation alternative de $\mathcal{E}(M)$ (lemme 4.4.8), on a :

$$\mathcal{E}(M) = \bigcap_{A \in \mathcal{D}(M)} \mathcal{E}xt'(A)$$

Soit une occurrence u dans $\mathcal{E}(M)$. Pour toute dérivation A: $M \to^* N$, on a donc $u \in \mathcal{E}xt'(A)$. Par définition de $\mathcal{E}xt'(A)$, l'occurrence u est préservée par la dérivation A (auquel cas on a $u \setminus A = \{u\}$), ou élément de $\mathcal{E}xt(A)$ (auquel cas le radical M/u est contracté par A). $\square$

Comme on a l'inclusion $\mathcal{E}(M) \subseteq \mathcal{E}xt(M)$, on a aussi l'inclusion $\mathcal{E}(M) \subseteq \mathcal{E}xt(M) \cap \mathcal{N}(M)$.

Corollaire 4.4.12 *Le terme M est en forme normale, si et seulement si $\mathcal{N}(M)$ est vide.*

Démonstration : Si M est en forme normale alors $\mathcal{N}(M) \subseteq \mathcal{R}(M) = \emptyset$. Si M n'est pas en forme normale, alors (lemme 4.4.4) on a $\emptyset \neq \mathcal{E}(M) \subseteq \mathcal{N}(M) \cap \mathcal{E}xt(M)$. $\square$

Autrement dit, dans tout terme qui n'est pas en forme normale, il existe des radicaux nécessaires, dont au moins un qui est un des radicaux les plus extérieurs du terme M.

L'emploi des descendants est la technique naturelle pour suivre les radicaux par delà les dérivations sans s'aider des étiquettes.

Lemme 4.4.13 *Soient un terme M et un radical R d'occurrence u dans M. Alors on a l'équivalence :*

$$u \in \mathcal{N}(M) \Leftrightarrow \forall A \in \mathcal{D}(M),\ u\backslash\!\!\backslash A \neq \emptyset$$

Démonstration : Soit u tel que l'on a $u\backslash\!\!\backslash A \neq \emptyset$. Alors, par définition on a $u \setminus A \neq \emptyset$ ou $u\backslash\!\!\backslash\!\!\backslash A \neq \emptyset$. En revenant aux définitions, on voit très facilement que $u\backslash\!\!\backslash\!\!\backslash A \neq \emptyset$ entraîne $u \in \mathcal{R}(A)$. Réciproquement, soit une occurrence de $\mathcal{R}(M)$ telle qu'il existe une dérivation A, avec $u\backslash\!\!\backslash A = \emptyset$. Soit B: $M \to^* N$ un équivalent standard de la dérivation A. Par le lemme 4.2.11-3, on a $u\backslash\!\!\backslash B = \emptyset$. D'où l'on tire immédiatement $u \setminus B = \emptyset$. Soit maintenant une occurrence v de $\mathcal{R}(B)$. Par définition, il existe une décomposition $B = B_1B_2B_3$, avec B_1: $M \to^* P$, B_2: $P \to Q$ et B_3: $Q \to^* N$, ainsi qu'une occurrence w du terme P, avec $w \in v \setminus B_1$ et $w \in \mathcal{R}(B_2)$. Or, la dérivation B est standard, on a donc $w \in \mathcal{E}(B_2B_3)$, soit, par le lemme 4.3.16, $w\backslash\!\!\backslash\!\!\backslash B_2B_3 \neq \emptyset$, d'où l'on tire finalement $w\backslash\!\!\backslash B \neq \emptyset$. Il est donc impossible que le radical initial $R = M/u$ soit contractée par la dérivation B. En d'autres termes, on a trouvé une dérivation B telle que $u \setminus B = \emptyset$ et $u \notin \mathcal{R}(B)$. □

Cette propriété caractéristique des radicaux nécessaires est la définition des radicaux nécessaire donnée par Z. Kasidashvili [Kha 90] (dans une version ultérieure de ce travail, la définition 4.4.9 est également donnée).

4.4.3 Appel par nécessité, appel par nom

L'appel par nécessité est une stratégie de réduction qui consiste à ne contracter que des radicaux nécessaires à chaque étape élémentaire, et ceci tant que possible.

Définition 4.4.14 *Soit un terme M. Une dérivation A: $M \to^* N$ suit la stratégie de l'appel par nécessité, si et seulement si on se trouve dans un des deux cas suivants :*

1. *Le terme M est en forme normale et la dérivation A est vide.*
2. *La dérivation A est de la forme $A = BC$, où B: $M \xrightarrow{U} P$ est une réduction non-vide telle que l'on a $U \subseteq \mathcal{N}(M)$, et la dérivation C suit la stratégie de l'appel par nécessité.*

Pour alléger la terminologie, on nomme appel par nécessité toute dérivation qui suit la stratégie de l'appel par nécessité. La définition de l'appel par nécessité est correcte du fait de l'équivalence "$\mathcal{N}(M)$ est vide, si et seulement si le terme M est en forme normale".

La stratégie d'appel par nécessité est une stratégie normalisante.

Proposition 4.4.15 *Soit un terme M normalisable et soit N la forme normale de M. Alors, tous les appels par nécessité issus de M sont de longueur finie et aboutissent en N.*

Démonstration : Soit un appel par nécessité A. Par unicité de la forme normale, si la dérivation A est finie, alors elle est de la forme A: $M \to^* N$. Par hypothèse, il existe une dérivation normalisante B: $M \to^* N$. Pour toute dérivation finie C: $M \to^* P$, il existe une dérivation $D = B \setminus C$, qui joint le terme P à la forme normale N. Soit ensuite un radical R d'occurrence $v \in \mathcal{N}(P)$. Par le lemme 4.4.13, il vient $u\backslash\!\!\backslash D \neq \emptyset$. Soit ici, une forme normale ne contenant aucun radical, on a $u \setminus D = \emptyset$, et donc, $u\backslash\!\!\backslash\!\!\backslash D \neq \emptyset$. Autrement dit (lemme 2.5.4), le nom α de R apparaît dans une étiquette de la forme normale N. On a donc montré que l'ensemble des noms des radicaux susceptibles d'être contractés par la dérivation A est fini. La dérivation A est donc bornée au sens de la section 3.3.2 et nécessairement finie. □

Ce lemme montre la correction de la stratégie d'appel par nécessité pour le calcul de la forme normale. On peut le généraliser aux stratégies qui sont presque des stratégies d'appel par nécessité, c'est à dire qui ne repoussent pas indéfiniement la contraction d'un radical nécessaire. C'est par exemple le cas de la stratégie "parallel-outermost", qui, à chaque étape, contracte tous les radicaux de $\mathcal{E}xt(M)$, dont on sait qu'un au moins est nécessaire.

Proposition 4.4.16 *Soit un terme M normalisable et soit N la forme normale de M. Alors, tous les appels par nécessité issus de M sont de longueur finie et aboutissent en N.*

Démonstration : Soit A une dérivation qui est presque un appel par nécessité. Comme dans la proposition précédente et par unicité de la forme normale, il suffit de montrer la finiture de la dérivation A. □

On examine ensuite une stratégie plus précise.

Définition 4.4.17 (Forme normale de tête) *Un terme M est en forme normale de tête, si toutes les dérivations issues de M sont des dérivations internes.*

Par définition, une dérivation A: $M \to^* N$ est une dérivation normalisante de tête, quand son terme final N est en forme normale de tête. Le terme N est alors une forme normale de tête du terme M. Il est immédiat qu'il n'y pas unicité de la forme normale de tête dans le cas général. Cependant, toutes les formes normales de tête d'un terme donné ont la même tête. En effet, soient deux dérivations normalisantes de tête A: $M \to^* N$ et B: $M \to^* P$. Alors, par le lemme des déplacements parallèles il existe deux dérivations de la forme C: $N \to^* Q$ et D: $P \to^* Q$. Par définition des formes normales de tête, les dérivations C et D sont internes. Par conséquent les trois termes N, Q et P ont la même tête.

Définition 4.4.18 *Soit un terme M. Une dérivation A: $M \to^* N$ suit la stratégie de l'appel par nom, si et seulement si on se trouve dans un des deux cas suivants :*

1. *Le terme M est en forme normale de tête et la dérivation A est vide.*
2. *Le terme M n'est pas en forme normale de tête et la dérivation A est de la forme $A = BC$, où B: $M \xrightarrow{U} P$ est une réduction non-vide telle que l'on a $U \subseteq \mathcal{E}(M)$ et la dérivation C suit la stratégie de l'appel par nom.*

Il faut bien remarquer que la condition d'arrêt de l'appel par nom "le terme M est en forme normale de tête" est plus fine que celle de l'appel par nécessité "le terme M est en forme normale". En reprenant cette dernière condition dans la définition de l'appel par nom, on n'aurait rien défini de plus qu'une certaine stratégie d'appel par nécessité. Or, comme on va le voir, la forme normale de tête atteinte en premier par un appel par nom est particulière, il serait dommage de l'ignorer.

Lemme 4.4.19 *Soient deux dérivations A et B issues d'un même terme M, telles que la dérivation A est interne. Alors la dérivation $B \setminus A$ est interne, si et seulement si la dérivation B l'est.*

Démonstration : On procède par induction sur la somme $|A| + |B|$. Il y a trois cas de base. Si la dérivation A est vide, alors (équation 4.3) on a $B \setminus A = B \setminus O = B$. Si la dérivation B est vide, alors (4.1.3-4.1) on a $B \setminus A = O \setminus A = O$. Enfin, si A et B sont des réductions, alors la réduction A n'est pas de type (red). On voit facilement, en revenant à la définition des résidus que, pour toute occurrence u dans M on a l'équivalence $u = \epsilon \Leftrightarrow u \setminus A = \{\epsilon\}$. Par conséquent l'occurrence à la racine appartient à $\mathcal{R}(B)$, si et seulement si elle appartient à $\mathcal{R}(B \setminus A) = \mathcal{R}(A) \setminus B$. Pour l'étape d'induction, on procède comme indiqué par les diagrammes des figures 4.12. Dans ces diagrammes, la lettre i désigne une dérivation interne et la lettre e une dérivation qui n'est pas interne. Le diagramme du haut correspond au cas où la dérivation B est interne, les deux diagrammes du bas à celui où la dérivation B n'est pas interne. □

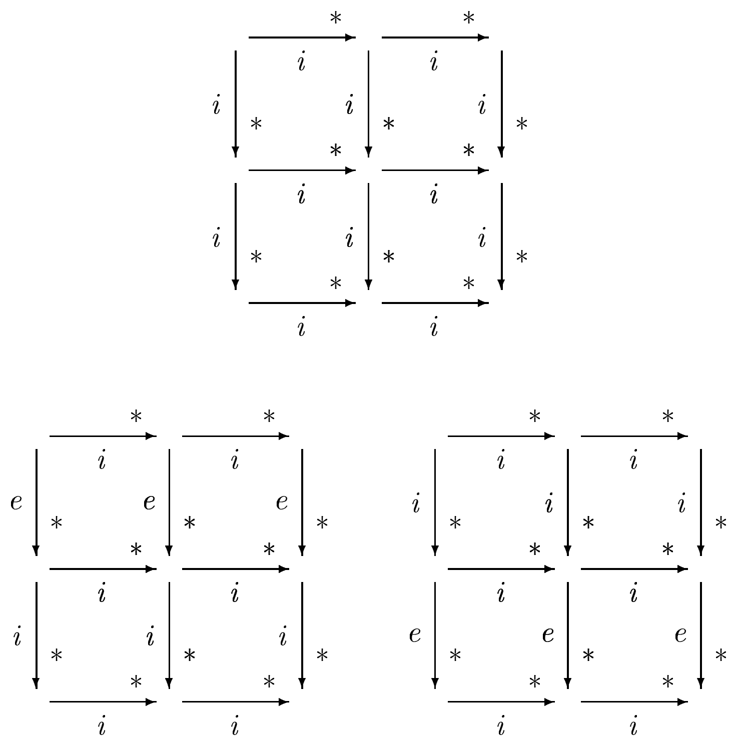


Figure 4.12 : Étape d'induction de la preuve du lemme 4.4.19

Lemme 4.4.20 *Soit une dérivation A: $M \rightarrow^* N$ dont le terme initial M n'est pas en forme normale de tête et le terme final N est en forme de tête. Alors, on a les deux résultats suivants :*

1. *La dérivation A admet au moins une étape élémentaire de type (red).*
2. *Pour toute occurrence u dans $\mathcal{E}(M)$, on a $u \in \mathcal{E}xt(A)$ et $u \backslash\!\backslash A = u \backslash\!\backslash\!\backslash A = \{\epsilon\}$*

Démonstration : Puisque M n'est pas en forme normale de tête, il existe une dérivation B: $M \rightarrow^* P$ qui n'est pas interne. Si la dérivation A était interne, alors, par le lemme 4.4.19, la dérivation $B \setminus A$ serait une dérivation non-interne et le terme N ne serait pas en forme normale de tête. Par conséquent la dérivation A admet au moins une étape de type (red). Dès lors, par définition, on a $\mathcal{E}(M) = \mathcal{E}(A)$. Donc, par le lemme 4.3.14, on a l'inclusion $\mathcal{E}(M) \subseteq \mathcal{E}xt(A)$. En outre, par le lemme 4.3.16, on sait que si u est une occurrence dans $\mathcal{E}(A)$, alors elle admet un unique descendant-héritier qui est une occurrence canonique de la dérivation A. Une dérivation non-interne n'admet que l'occurrence vide ϵ comme occurrence canonique. On a donc ici $u \in \mathcal{E}(M)$ entraîne $u \backslash\!\backslash A = u \backslash\!\backslash\!\backslash A = \{\epsilon\}$. □

On peut enfin affirmer la correction de l'appel par nom.

Proposition 4.4.21 *Soit un terme M. Si il existe une dérivation A: $M \rightarrow^* N$ dont le terme final N est en forme normale de tête, alors il existe un terme P unique tel que tous les appels par nom sont des dérivations finies équivalentes de la forme B: $M \rightarrow^* P$, avec $B \sqsubseteq A$. Ce terme P est la forme normale de tête du terme M.*

Démonstration : Pour toute dérivation C: $M \rightarrow^* Q$ telle que le terme Q n'est pas en forme normale de tête, on considère la dérivation $A' = A \setminus C$. La dérivation A' est de la forme A': $Q \rightarrow^* N'$, où le terme N' est en forme normale de tête avec $\tau(N') = \tau(N)$ (considérer la dérivation interne $C \setminus A\colon N \to^* N'$). Par conséquent (lemme 4.4.20), pour tout radical d'occurrence u dans $\mathcal{E}(Q)$, on a $u \backslash\!\backslash\!\backslash A' = \{\epsilon\}$. C'est à dire que, par le lemme 2.5.4, le nom du radical d'occurrence u dans le terme Q apparaît dans l'étiquette de tête $\tau(N') = \tau(N)$. Par conséquent, tous les radicaux susceptibles d'être contractés par une dérivation qui suit la stratégie de l'appel par nom sont bornés. Soit finalement, tous les appels par nom sont des dérivations bornées qui sont nécessairement de longueur finie (proposition 3.3.11).

On montre ensuite que tous les appels par nom sont équivalents. Soit $B\colon M \to^* P$ un appel par nom issu de M. On commence par montrer l'inclusion $B \sqsubseteq A$ par induction sur la longueur de la dérivation B. Dans le cas $B = O$, on a $O \sqsubseteq A$ quelle que soit la dérivation A. Sinon, l'appel par nom B se décompose en $B = B_1 B_2$, où B_1 est une réduction $B_1\colon M \to Q$ qui contracte des radicaux standards à partir de M et $B_2\colon Q \to^* P$ est un appel par nom. Puisque la dérivation B n'est pas vide, le terme M n'est pas en forme normale de tête. Dès lors (lemme 4.4.20) et puisque les radicaux extérieurs sont contractés sans laisser de résidu, on a $\mathcal{E}(M) \setminus A = \emptyset$. Soit encore, $B_1 \setminus A = \emptyset$. Par hypothèse d'induction, appliquée à la dérivation $A \setminus B_1\colon Q \to^* P$ et à l'appel par nom B_2, on a $B_2 \setminus (A \setminus B_1) \equiv O$. Soit finalement, $B \setminus A = (B_1 \setminus A)\,(B_2 \setminus (A \setminus B_1)) \equiv O$ et $B \sqsubseteq A$.

Enfin, soit maintenant deux appels par nom $B\colon M \to^* P$ et $B'\colon M \to^* P'$. Par le résultat du paragraphe précédent, on a $B' \sqsubseteq B$ et $B \sqsubseteq B'$. Soit finalement, $B \equiv B'$ et $P = P'$. □

La stratégie de l'appel par nom est donc correcte pour le calcul de la forme normale de tête, la forme normale de tête P du terme M étant un terme défini dans le lemme ci-dessus et qui possède les deux propriétés suivantes :

1. Le terme P est une forme normale de tête du terme M.
2. Pour toute forme normale de tête N du terme M, il existe une dérivation $A\colon P \to^* N$.

4.4.4 Calcul des préfixes stables

Cette section généralise l'idée de la forme normale de tête à tout un préfixe.

Définition 4.4.22 (Préfixes stables) *Soit un terme M. Le terme linéaire γ est un préfixe stable dans le terme M, si et seulement si il est préservé par toutes les dérivations issues de M.*

Remarquons qu'un terme admet le préfixe γ comme préfixe stable, si et seulement si il existe une substitution σ avec $\sigma(\gamma)$ et que, pour toute occurrence u dure dans γ, le sous-terme M/u est en forme normale de tête. Ainsi, le préfixe $\gamma = x$ est toujours stable.

Un terme M admet *une forme normale relativement au préfixe* γ, quand il existe une dérivation $A\colon M \to^* N$ telle que le préfixe γ est un préfixe stable de N.

Étant donnés un terme M et un préfixe γ qui n'est pas une variable, une dérivation A issue du terme M est un appel par nom vers le préfixe γ quand elle se conforme au schéma suivant :

1. La dérivation A commence par calculer la forme normale de tête P du terme M, en suivant la stratégie de l'appel par nom.
2. Si le terme M est normalisable de tête, alors ce calcul réussit. On appelle $B\colon M \to^* P$ le début de la dérivation A calculé jusqu'ici et on considère les deux cas possibles :
   (a) Si les symboles de tête (étiquetés) du préfixe γ et de la forme normale de tête P sont distincts, alors échouer.

(b) Sinon, le terme P est de la forme $P = f^l(P_1, \ldots, P_a)$ et le préfixe γ est de la forme $\gamma = f^l(\gamma_1, \ldots, \gamma_a)$. Soit I l'ensemble des indices autorisés à partir du symbole f qui sont tels que le fils γ_i n'est pas une variable ($I = \{\, i \in \mathrm{dom}(f) \mid \gamma_i \notin \mathcal{X}\,\}$). Dans le cas où l'ensemble I n'est pas vide, on poursuit le calcul par une dérivation interne C, dont aucune étape n'est vide et qui est telle que, pour tout indice $i \in I$, la dérivation argument C/i est un appel par nom vers le préfixe γ_i, abstraction faite d'éventuelles étapes vides ; tandis que, pour tout indice $i \notin I$, la dérivation argument C/i est nulle.

On montre facilement par induction sur le préfixe γ que, si la dérivation A: $M{\to}^*N$ termine normalement, alors le terme N est une forme normale relativement au préfixe γ.

Lemme 4.4.23 *Soit un terme M qui admet une forme normale relativement à un préfixe γ. Alors, tous les appels par nom vers γ issus du terme M sont des dérivations finies. En outre, pour toute dérivation A: $M{\to}^*N$ qui est telle que le terme N est en forme normale relativement au préfixe γ et tout appel par nom vers le préfixe γ, on a l'inclusion $B \sqsubseteq A$.*

Démonstration : La preuve est par induction sur le préfixe à atteindre. Soit donc B un appel par nom vers le préfixe γ. Si γ est une variable, alors on a nécessairement $B = O$. Sinon, on pose $\gamma = f^l(\gamma_1, \ldots, \gamma_a)$. Par hypothèse, le terme M possède une forme normale de tête $P = f^l(P_1, \ldots, P_a)$ et, en raison de la correction de l'appel par nom, la dérivation B commence par un appel par nom fini B': $M{\to}^*P$. Ensuite, on suppose qu'il existe des fils autorisés γ_i qui ne sont pas des variables et on considère la suite B'' de B. Si la dérivation B'' pouvait échouer, alors il existerait un de ces fils γ_{i_0} et un appel par nom vers le préfixe γ_{i_0} qui échouerait (il s'agit de la dérivation B''/i_0, moins ses étapes vides). Par hypothèse d'induction, le fils P_{i_0} ne serait donc pas donc pas normalisable relativement au préfixe γ_{i_0}. Or, on montre facilement le contraire, en considérant la dérivation interne $A \setminus B'$ et son argument $(A \setminus B')/i_0$. Si la dérivation B'' pouvait ne pas terminer, alors, puisqu'aucune des étapes de B'' n'est vide, il existerait une dérivation argument B''/i_0 qui comprendrait un nombre infini d'étapes non-vides. Il y aurait donc un fils dur et autorisé γ_{i_0}, ainsi qu'un appel par nom vers ce préfixe qui ne terminerait pas. Par induction, ce cas est impossible comme le précédent. Par conséquent, la dérivation B termine et est de la forme B: $M{\to}^*P$.

On prouve ensuite l'inclusion $B \sqsubseteq A$, par induction sur le préfixe γ et en reprenant les notations du paragraphe précédent. Par la proposition 4.4.21, on a l'inclusion $B' \sqsubseteq A$. Par hypothèse d'induction, pour tout indice i du domaine de f tel que γ_i n'est pas une variable, on a l'inclusion $B''/i \sqsubseteq (A \setminus B')/i$. Pour tout autre indice i à partir du symbole f, la dérivation argument B''/i est nulle et on a également $B''/i \sqsubseteq (A \setminus B')/i$. D'où, par la propriété $C \sqsubseteq D \Leftrightarrow \exists C',\ CC' \equiv D$ et la caractérisation des dérivations équivalentes par les étiquettes complètes, il vient facilement $B'' \sqsubseteq A \setminus B'$. Soit finalement $B \sqsubseteq A$, en considérant la dérivation $B \setminus A = (B' \setminus A)\,(B'' \setminus (A \setminus B'))$ qui est nulle.

Enfin, par un argument de symétrie évident, tous les appels par noms vers le préfixe γ sont équivalents. □

Ce dernier lemme met bien l'accent sur la différence entre l'appel par nom et l'appel par nécessité. L'appel par nécessité n'est sûr que pour le calcul de la forme normale, tandis que l'appel par nom est sûr pour le calcul de n'importe quel préfixe stable. Par exemple, soient les trois règles $F(x,y) \to G(y,x)$, $A \to B$ et $C \to C$. On considère les deux dérivations infinies :

$$A_1\colon F(A,C) \xrightarrow{\epsilon} G(C,A) \xrightarrow{2} G(C,B) \xrightarrow{1} \ldots \qquad\qquad A_2\colon F(A,C) \xrightarrow{1} F(B,C) \xrightarrow{2} \ldots$$

Si elle repousse indéfiniment la contraction du radical à la racine la dérivation A_2 n'atteint pas une forme normale de tête. Il s'agit cependant d'un appel par nécessité, car aucun radical ne peut ici être effacé. En revanche, les deux premières étapes de la dérivation A_1 constituent un appel par nom vers le préfixe stable $G(x, B)$. L'appel par nom est donc une stratégie plus précise que l'appel par nécessité dans le cas des termes non-normalisables.

Chapitre 5

Dérivations complètes

La dernière section du chapitre précédent a semblé disqualifier l'appel par nécessité, comme une stratégie seulement utile pour calculer les formes normales. Ce chapitre va le réhabiliter, en montrant que c'est l'appel par nécessité qui possède la propriété d'optimalité la plus simple et la plus générale.

5.1 Sous-treillis des dérivations complètes

À toute dérivation A, on associe as *complétion*, qui est une dérivation complète qui contracte les familles complètes des radicaux contractés par A.

Définition 5.1.1 *Soit une réduction A: $M \stackrel{U}{\to} N$. Soit $\{\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_n\}$ la partition par familles de l'ensemble U. Pour toute famille $\mathcal{F}_i$ de radicaux de nom α_i, on note $\overline{\mathcal{F}_i}$, la complétion de la famille $\mathcal{F}_i$, c'est à dire la famille complète qui regroupe tous les radicaux de nom α_i. La complétion $\overline{A}$ de A est alors la réduction issue de M qui contracte l'ensemble complet de radicaux : $\overline{U} = \overline{\mathcal{F}}_1 \cup \overline{\mathcal{F}}_2 \cup \ldots \cup \overline{\mathcal{F}}_n$.*

Remarquons que l'on a $A \sqsubseteq \overline{A}$, puisque $U \subseteq \overline{U}$.

Définition 5.1.2 (Complétion d'une dérivation) *Soit une dérivation A: $M {\to}^* N$. Si A est la dérivation vide O, alors on pose $\overline{A} = O$. Sinon, il existe une réduction B: $M \to P$ et une dérivation C: $P {\to}^* N$ telles que $A = BC$ et on pose :*

$$\overline{A} = \overline{B}\,(\overline{C \backslash (\overline{B} \backslash B)})$$

Comme le montre la figure 5.1, on peut concaténer la réduction $\overline{B}$ et la dérivation $C \backslash (\overline{B} \backslash B)$, parce que l'on a $B \sqsubseteq \overline{B}$.

Lemme 5.1.3 *Pour toute dérivation A, on a l'inclusion $A \sqsubseteq \overline{A}$.*

Démonstration : On montre plutôt l'existence d'une dérivation A', avec $AA' \equiv \overline{A}$ (lemme 4.2.7). Le cas de base $A = \overline{A} = O$ est immédiat. Dans le cas général $|A| \geq 1$, il existe une réduction B et une dérivation C, telles que l'on a $A = BC$. Par définition de la complétion des réductions, on a $B \sqsubseteq \overline{B}$. Soit (lemme 4.2.7), $B\,(\overline{B} \setminus B) \equiv \overline{B}$, que l'on note $BB' \equiv \overline{B}$. Par le lemme des déplacements parallèles généralisés, on a $C \sqcup B' \equiv B' \sqcup C$, que l'on note $CB'' \equiv B'C'$ (NB, on a $C' = C \setminus (\overline{B} \setminus B)$, soit encore $\overline{A} = \overline{B}\,\overline{C'}$) En outre, par hypothèse d'induction ($|C'| = |C \setminus (\ldots)| = |C| < |A|$), il existe

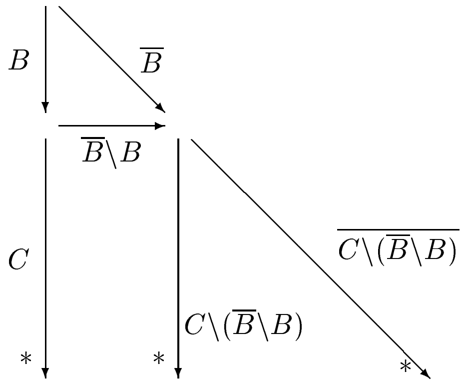


Figure 5.1 : Complétion de la dérivation BC

une dérivation C'', avec $C'C'' \equiv \overline{C'}$ Soit, la relation $\equiv$ étant une congruence pour la concaténation, il existe une dérivation $A' = B''C''$, avec :

$$AA' = B(CB'')C'' \equiv (BB')(C'C'') \equiv \overline{B}\,\overline{C'} = \overline{A}$$

□

Du point de vue des diagrammes, ce lemme se traduit par la fermeture par le bas de la figure 5.1, déduite de la fermeture du rectangle et du triangle de sa partie inférieure.

Puisque l'on a $A \sqsubseteq \overline{A}$ dans le cas général, on peut envisager de faire abstraction de la contrainte $|B| = 1$ de la définition de la complétion.

Lemme 5.1.4 *Soit une dérivation $A = BC$. On a l'égalité, $\overline{A} = \overline{B}\,(\overline{C\backslash(\overline{B}\backslash B)})$.*

Démonstration : Le cas $B = O$ est immédiat : on a $\overline{O}\,(\overline{C\backslash(\overline{O}\backslash O)}) = C$, puisque $\overline{O} = O, O \setminus O = O$, $C \setminus O = C$ et $OC = C$. Le cas $|B| = 1$ est traité par la définition. Reste a prouver le cas $|B| > 1$. On procède par induction sur $|B| + |A|$, en posant $B = B_1B_2$ avec $|B_1| = 1$.

$$\begin{aligned}
\overline{A} &= \overline{B_1}\,(\overline{(B_2C)\setminus(\overline{B_1}\setminus B_1)}) && \text{définition de } \overline{A} \\
&= \overline{B_1}\,(\overline{(B_2\setminus(\overline{B_1}\setminus B_1))\,(C\setminus((\overline{B_1}\setminus B_1)\setminus B_2))}) && \text{décomposition 4.1.3-4.2} \\
&= \overline{B_1}\,(\overline{(B_2\setminus(\overline{B_1}\setminus B_1))\,(C\setminus(\overline{B_1}\setminus B))}) && \text{recomposition 4.1.3-4.4}
\end{aligned}$$

Posons $B_2' = B_2\setminus(\overline{B_1}\setminus B_1)$ et $C' = C\setminus(\overline{B_1}\setminus B)$ et continuons. On peut appliquer l'hypothèse d'induction à la dérivation $B_2'C'$ ($|B_2'| + |B_2'C'| < |B| + |A|$), soit :

$$\overline{A} = \overline{B_1}\,\overline{B_2'}\,(\overline{C'\setminus(\overline{B_2'}\setminus B_2')})$$

Or $\overline{B_1}\,\overline{B_2'} = \overline{B_1}\,(\overline{B_2\setminus(\overline{B_1}\setminus B_1)})$. Et donc, par une nouvelle application de l'hypothèse d'induction ($|B_1| + |B_1B_2| < |B| + |A|$), on a $\overline{B_1}\,\overline{B_2'} = \overline{B}$ (1). Par ailleurs, on a :

$$\begin{aligned}
\overline{C'\setminus(\overline{B_2'}\setminus B_2')} &= \overline{(C\setminus(\overline{B_1}\setminus B))\setminus(\overline{B_2'}\setminus B_2')} \\
&= \overline{C\setminus((\overline{B_1}\setminus B)\,(\overline{B_2'}\setminus B_2'))} && \text{recomposition 4.1.3-4.4}
\end{aligned}$$

Par ailleurs, (décomposition 4.2 et $B_1 \sqsubseteq \overline{B_1}$), on a $B \setminus \overline{B_1} = (B_1 \setminus \overline{B_1})\,(B_2 \setminus (\overline{B_1} \setminus B_1) = \emptyset B_2'$. On a donc $\overline{B_2'} \setminus B_2' = \overline{B_2'} \setminus (\emptyset B_2') = \overline{B_2'} \setminus (B \setminus \overline{B_1})$ (rappelons que, pour toute dérivation D, on a $D \setminus \emptyset = D$). Soit finalement :

$$\begin{array}{rcll} \overline{C' \setminus (\overline{B_2'} \setminus B_2')} & = & \overline{C \setminus ((\overline{B_1} \setminus B)\,(\overline{B_2'} \setminus (B \setminus \overline{B_1})))} & \\ & = & \overline{C \setminus ((\overline{B_1}\,\overline{B_2'}) \setminus B)} & \text{recomposition 4.1.3-4.2} \\ & = & \overline{C \setminus (\overline{B} \setminus B)} & \text{cf. (1)} \end{array}$$

□

Les dérivations complètes ont déjà été définies à la section 2.6, comme les dérivations dont toutes les étapes élémentaires contractent des familles complètes. Il est clair qu'avec une telle définition, les dérivations complètes sont stables par concaténation. Mieux, pour toutes dérivations A: $M \to^* N$ et B: $N \to^* P$ concaténables, la concaténation AB est complète, si et seulement si les dérivations A et B sont complètes.

On peut aussi définir les dérivations complètes à partir des complétions.

Lemme 5.1.5 *Une dérivation est complète, si et seulement si elle est égale à sa complétion.*

Démonstration : La preuve est par induction sur la longueur de A. Le cas $|A| = 0$ est trivial ($A = \overline{A} = O$). Le cas $|A| = 1$ est évident à partir des définitions. Enfin, dans le cas $|A| > 1$, on pose $A = BC$, avec $|B| > 0$ et $|C| > 0$. Par stabilité des dérivations complètes par concaténation et par hypothèse d'induction, on a les équivalences :

$$A \text{ complète} \Leftrightarrow B \text{ complète et } C \text{ complète} \Leftrightarrow B = \overline{B} \text{ et } C = \overline{C}$$

Par ailleurs, par le lemme 5.1.4, on a :

$$\overline{A} = \overline{B}\,\overline{C \setminus (\overline{B} \setminus B)}$$

Supposons tout d'abord $B = \overline{B}$ et $C = \overline{C}$. On en tire, d'une part $\overline{B} = B$, et d'autre part $\overline{B} \setminus B = \emptyset^{|B|}$, $C \setminus (\overline{B} \setminus B) = C$, puis $\overline{C \setminus (\overline{B} \setminus B)} = \overline{C}$. Soit finalement, $\overline{A} = BC = A$. Réciproquement, si on suppose $A = \overline{A}$, alors on a $BC = \overline{B}\,(\overline{C \setminus (\overline{B} \setminus B)})$. On montre facilement par récurrence qu'une dérivation et sa complétion ont la même longueur, soit ici $|B| = |\overline{B}|$. On tire donc $B = \overline{B}$ de l'hypothèse, que l'on peut ensuite réécrire comme $BC = B\,(\overline{C \setminus \emptyset^{|B|}})$, d'où il vient finalement, $C = \overline{C}$. □

Ce lemme permet de noter toute dérivation complète A comme une complétion $\overline{B}$, puisqu'on pourra toujours poser $B = A$.

Les dérivations complètes sont stables par l'opération de résidu. En fait on montre une propriété un petit peu plus générale.

Lemme 5.1.6 *Soient un terme cohérent M, une dérivation complète $\overline{A}$: $M \to^* N$ et une dérivation quelconque B: $M \to^* P$. Alors la dérivation $\overline{A} \setminus B$ est une dérivation complète.*

Démonstration : La preuve est par induction sur $|\overline{A}|$. Le cas de base $\overline{A} = O$ est évident : $O \setminus B = O$. Dans le cas général, on décompose la dérivation A, sous la forme $\overline{A} = \overline{A}_1\,\overline{A}_2$, où $\overline{A}_1$: $M \to Q$ est une réduction complète qui contracte l'ensemble complet de radicaux $\overline{U} = \overline{\mathcal{F}}_1 \cup \overline{\mathcal{F}}_2 \cup \ldots \cup \overline{\mathcal{F}}_n$, tandis que $\overline{A}_2$: $Q \to^* N$ est une dérivation complète. Par définition, on a l'égalité $\overline{A} \setminus B = (\overline{A}_1 \setminus B)\,(\overline{A}_2 \setminus (B \setminus \overline{A}_1))$. Par définition, la réduction $\overline{A}_1 \setminus B$ contracte celles des familles $\overline{\mathcal{F}}_i \setminus B$

qui ne sont pas vides. Or, par le lemme 2.5.3 et pour tout entier i compris entre 1 et n, les familles $\overline{\mathcal{F}}_i \setminus B$ sont vides ou complètes. C'est à dire que la réduction $\overline{A}_1 \setminus B$ est complète. En outre, par le lemme 2.6.2, le terme Q est cohérent, on peut donc appliquer l'hypothèse d'induction à la dérivation complète $\overline{A}_2$ pour prouver que la dérivation $\overline{A}_2 \setminus (B \setminus \overline{A}_1)$ est complète. Soit finalement, $\overline{A} \setminus B$ est une dérivation complète. □

Il faut noter que nous avons eu besoin de la cohérence du terme initial M. Ce n'est pas très étonnant, car la cohérence du terme M est associée à des familles de radicaux "raisonnables" dans les termes dérivés du terme M. Par exemple, le lemme 2.5.3 assure que, dans le cas où le terme initial M de la dérivation B: $M \to^* P$ est cohérent, alors les résidus des familles complètes de radicaux du terme M forment des familles complètes de radicaux dans le terme P. En un sens, le lemme 5.1.6 ci-dessus généralise ce résultat de conservation des familles au cas des radicaux créés.

On peut insister sur la structure de l'ensemble des dérivations complètes issues d'un terme donné.

Proposition 5.1.7 *Étant donné un terme cohérent M, on note $\overline{\mathcal{D}}(M)$ l'ensemble des dérivations complètes et finies issues du terme M. Alors, $<\overline{\mathcal{D}}(M)/\equiv, \sqsubseteq, \sqcup>$ est un sous-treillis du sup-demi-treillis des dérivations finies issues du terme M.*

Démonstration : Il s'agit d'une conséquence directe de la structure de sup-demi-treillis de l'ensemble $\mathcal{D}(M)/\equiv$ et de la fermeture de l'ensemble $\overline{\mathcal{D}}(M)$ par les opérations de concaténation et de résidu. □

Les dérivations complètes contractent des familles de radicaux et non plus des radicaux isolés. C'est pourquoi on révise certaines définitions. Tout d'abord, on note $\overline{\mathcal{R}}(M)$ l'ensemble des familles complètes de radicaux du terme M.

Définition 5.1.8 *Soient un terme cohérent M et une dérivation complète $\overline{A}$: $M \to^* N$. On définit l'ensemble $\overline{\mathcal{R}}(\overline{A})$ des familles de radicaux contractées par la dérivation $\overline{A}$ de la façon suivante :*

1. $\overline{\mathcal{R}}(O) = \emptyset$
2. *Si la dérivation $\overline{A}$ se décompose comme $\overline{A} = \overline{B}\,\overline{C}$, avec :*

$$\overline{B}\colon M \overset{\overline{\mathcal{F}}_1 \cup \overline{\mathcal{F}}_2 \cup \ldots \cup \overline{\mathcal{F}}_n}{\longrightarrow} P \qquad \overline{C}\colon P \to^* N$$

Alors, on pose $\overline{\mathcal{R}}(\overline{A}) = \{\overline{\mathcal{F}}_1, \overline{\mathcal{F}}_2, \ldots, \overline{\mathcal{F}}_n\} \cup \{\, \overline{\mathcal{F}} \in \overline{\mathcal{R}}(M) \mid \overline{\mathcal{F}} \setminus \overline{B} \in \overline{\mathcal{R}}(\overline{C})\,\}$

Cette définition est correcte, puisque par le lemme 2.5.3 les ensembles $\overline{\mathcal{F}} \setminus \overline{B}$ sont bien des familles complètes du terme P, qui est bien un terme cohérent par le lemme 2.6.2. Comme dans le cas des radicaux contractés et pour toutes dérivations $\overline{A}$ et $\overline{B}$ concaténables, on a l'équivalence :

$$\overline{\mathcal{F}} \in \overline{\mathcal{R}}(\overline{A}\,\overline{B}) \Leftrightarrow \overline{\mathcal{F}} \in \overline{\mathcal{R}}(A) \text{ ou } \overline{\mathcal{F}} \setminus \overline{A} \in \overline{\mathcal{R}}(\overline{B})$$

Les familles contractées sont en fait les familles dont au moins un des membres est un radical contracté.

Lemme 5.1.9 *Soit M un terme cohérent. Pour toute dérivation complète $\overline{A}$: $M \to^* N$, on a l'équivalence :*

$$\overline{\mathcal{F}} \in \overline{\mathcal{R}}(\overline{A}) \Leftrightarrow \overline{\mathcal{F}} \cap \mathcal{R}(\overline{A}) \neq \emptyset$$

Démonstration : La preuve est par induction sur la longueur $|\overline{A}|$. Si la dérivation $\overline{A}$ est vide, alors on a $\overline{\mathcal{R}}(\overline{A}) = \mathcal{R}(\overline{A}) = \emptyset$. Sinon, la dérivation $\overline{A}$ se décompose comme la concaténation d'une réduction complète $\overline{B}$: $M \to P$ avec $\overline{\mathcal{R}}(\overline{B}) = \overline{\mathcal{F}}_1, \overline{\mathcal{F}}_2, \ldots, \overline{\mathcal{F}}_n$ et d'une dérivation complète $\overline{C}$: $P \to^* N$. On a alors les équivalences successives :

$$\overline{\mathcal{F}} \in \overline{\mathcal{R}}(\overline{A}) \Leftrightarrow \left\{ \begin{array}{ccc} \overline{\mathcal{F}} \in \overline{\mathcal{R}}(\overline{B}) & \Leftrightarrow & \overline{\mathcal{F}} \subseteq \mathcal{R}(\overline{B}) \\ & \text{ou bien} & \\ \overline{\mathcal{F}} \setminus \overline{B} \in \overline{\mathcal{R}}(\overline{C}) & \Leftrightarrow & \overline{\mathcal{F}} \setminus \overline{B} \cap \mathcal{R}(\overline{C}) \neq \emptyset \end{array} \right\} \Leftrightarrow \overline{\mathcal{F}} \cap \mathcal{R}(\overline{A}) \neq \emptyset$$

□

Il ne faut pas conclure de ce dernier résultat que les familles contractées sont une notion de peu d'intérêt. Bien au contraire, elles capturent ce qui fait l'essence de comportement raisonnable des dérivations complètes. En particulier, un résultat “intuitif” qui fait défaut dans le cas ces radicaux contractés est vrai pour les familles contractées.

Lemme 5.1.10 *Soit M un terme cohérent. Pour toute dérivation complète $\overline{A}$: $M \to^* N$ on a l'implication :*

$$\overline{\mathcal{F}} \in \overline{\mathcal{R}}(\overline{A}) \Rightarrow \overline{\mathcal{F}} \setminus \overline{A} = \emptyset$$

Démonstration : La propriété se démontre par induction sur $|A|$. Le cas de base ($\overline{A} = O$) est trivial. Soit ensuite une dérivation complète $\overline{A}$ qui se décompose comme $\overline{A} = \overline{B}\,\overline{C}$, où $\overline{B}$ est une réduction complète qui contracte les familles complètes $\overline{\mathcal{F}}_1, \overline{\mathcal{F}}_2, \ldots, \overline{\mathcal{F}}_n$. Soit une famille complète $\overline{\mathcal{F}}$ dans $\overline{\mathcal{R}}(\overline{A})$. Si la famille $\overline{\mathcal{F}}$ est une des familles $\overline{\mathcal{F}}_i$, alors tous les radicaux membres de $\overline{\mathcal{F}}$ sont contractés par la réduction $\overline{B}$ et n'ont pas de résidu (lemme 1.2.11) on a donc $\overline{\mathcal{F}} \setminus \overline{B} = \emptyset$, soit finalement $\overline{\mathcal{F}} \setminus \overline{A} = (\overline{\mathcal{F}} \setminus \overline{B}) \setminus \overline{C} = \emptyset$. Sinon, la famille complète $\overline{\mathcal{F}} \setminus \overline{B}$ est contractée par la dérivation $\overline{C}$ et on a $\overline{\mathcal{F}} \setminus \overline{A} = (\overline{\mathcal{F}} \setminus \overline{B}) \setminus \overline{C} = \emptyset$, par hypothèse d'induction. □

Il y a un rapport étroit entre les radicaux contractés par une dérivation et les familles complètes contractées par sa complétion.

Lemme 5.1.11 *Soit un terme cohérent M. Soit une dérivation A: $M \to^* N$ et soit $\overline{A}$: $M \to^* P$ sa complétion. Alors, on a l'implication :*

$$\overline{\mathcal{F}} \in \overline{\mathcal{R}}(\overline{A}) \Rightarrow \overline{\mathcal{F}} \cap \mathcal{R}(A) \neq \emptyset$$

Démonstration : La preuve est par induction sur la longueur de la dérivation A. Si la dérivation A est la dérivation vide, alors on a $\mathcal{R}(A) = \overline{\mathcal{R}}(\overline{A}) = \emptyset$. Sinon, la dérivation A se décompose comme $A = BC$, avec B: $M \to Q$ et C: $Q \to^* N$. Par définition, on a alors l'égalité $\overline{A} = \overline{B}\,(\overline{C \setminus (\overline{B} \setminus B)})$. Soit $\overline{\mathcal{F}}$ une famille de $\overline{\mathcal{R}}(\overline{A})$. Si $\overline{\mathcal{F}}$ est contractée par la réduction complète $\overline{B}$, c'est que, par définition, il existe un membre u de la famille $\overline{\mathcal{F}}$ qui est contracté par la réduction B. Sinon, on a $\overline{\mathcal{F}} \setminus \overline{B} \in \overline{\mathcal{R}}(\overline{C \setminus (\overline{B} \setminus B)})$. À l'aide de l'hypothèse d'induction ($|\overline{C} \setminus \ldots| = |C| < |A|$) on prouve l'existence d'au moins une occurrence de radical u dans l'ensemble $(\overline{\mathcal{F}} \setminus \overline{B}) \cap \mathcal{R}(C \setminus (\overline{B} \setminus B))$. Il existe donc dans la famille $\overline{\mathcal{F}}$, une occurrence de radical v, avec $u \in v \setminus \overline{B}$. De l'inclusion $B \sqsubseteq \overline{B}$, exprimée sous la forme $\overline{B} \equiv B \sqcup \overline{B}$ et de l'égalité $v \setminus \overline{B} = v \setminus (B \sqcup \overline{B})$ (lemme 4.2.11), on déduit l'existence dans le terme Q d'une occurrence de radical w, telle que l'on a $w \in v \setminus B$ et $u \in w \setminus (\overline{B} \setminus B)$, avec $u \in \mathcal{R}(C \setminus (\overline{B} \setminus B))$. Soit, par le lemme 4.3.2, $w \in \mathcal{R}(C)$, puis $v \in \mathcal{R}(A)$. □

5.2 Mise en œuvre des dérivations complètes

On sait déjà que, au prix d'un certain nombre de restrictions (les membres droits des règles sont des graphes acycliques, les réductions ne contractent que des termes partagés et sont issues de graphes acycliques), la réécriture simplement étiquetée est un modèle de la réécriture de graphes acycliques. Dans cette section, on va voir que cette même réécriture simplement étiquetée (et donc la réécriture de graphes acycliques) peut être utilisée pour mettre en œuvre les dérivations complètes.

Commençons par considérer un terme complètement étiqueté M quelconque. L'image $S(M)$ du terme M par le morphisme de simplification S (voir la section 3.1.2) est un terme simplement étiqueté, que l'on appelle la *mise en œuvre* du terme M dans les graphes acycliques. Pour toute occurrence u dans M l'étiquette simple $\tau(S(M)/u)$ est *l'adresse* du sous-terme M/u dans la mise en œuvre de M. On note $\&(M/u)$ cette adresse. De même, étant donné un système de réécriture complètement étiqueté Σ^l, on a appelle *mise en œuvre* de Σ^l, son image $S(\Sigma^l)$ par le morphisme de simplification S. Si le terme simplement étiqueté $S(M)$ est un graphe acyclique au sens de la définition 1.2.2, alors tous les radicaux du terme M qui sont membres d'une même famille ont même adresse dans la mise en œuvre du terme M sous forme d'un graphe acyclique $S(M)$. Pour qu'une famille complète de radicaux dans le terme M, corresponde exactement à un radical partagé de $S(M)$, il faut imposer la condition supplémentaire que deux radicaux dans M qui on même adresse ont aussi même nom. En fait, on prend des conditions un peu plus fortes, afin qu'elles restent stables dérivation.

Définition 5.2.1 (Quasi-égalité) *Soient deux termes complètement étiquetés M_1 et M_2, par définition, les termes M_1 et M_2 sont quasi-égaux et l'on note $M_1 \simeq M_2$, si et seulement si on se trouve dans l'un des deux cas suivants :*

1. $M_1 = x_1^{l_1.e_1}$, $M_2 = x_2^{l_2.e_2}$, $x_1 = x_2$ *et* $e_1 = e_2$
2. $$\left\{\begin{array}{l} M_1 = f_1^{l_1.e_1}(N_1,\ldots,N_{a_1}) \\ M_2 = f_2^{l_2.e_2}(P_1,\ldots,P_{a_2}) \end{array}\right. ,\ e_1 = e_2,\ f_1 = f_2 \textit{ et } \forall i \in [1\ldots a_1]\ N_i = P_i$$

C'est à dire que les termes quasi-égaux ne peuvent se distinguer que par les parties gauches de leurs étiquettes de têtes.

Définition 5.2.2 (Graphe acyclique dans les étiquettes complètes) *Un terme M complètement étiqueté est un graphe acyclique, si et seulement si les deux conditions suivantes sont vérifiées :*

1. *Le terme simplement étiqueté $S(M)$ est un graphe acyclique cohérent au sens des étiquettes simples.*
2. *Pour tous sous-termes M/u_1 et M/u_2, on a l'implication*

$$\&(M/u_1) = \&(M/u_2) \Rightarrow M/u_1 \simeq M/u_2$$

Remarquons que les deux parties de cette définition ne sont pas indépendantes. En effet, si les sous-termes M/u_1 et M/u_2 sont quasi-égaux, alors leurs mises en œuvre $S(M)/u_1$ et $S(M)/u_2$ sont égales, un résultat que l'on peut tirer directement de la condition *1* ci-dessus. Le point important est que les radicaux partagés sont la mise en œuvre exacte des familles complètes.

Corollaire 5.2.3 *Soit un terme M complètement étiqueté qui est un graphe acyclique. Quelles que soient les occurrences u_1 et u_2 dans M, les sous-termes M/u_1 et M/u_2 sont des radicaux de même nom α, si et seulement si les sous-termes $S(M)/u_1$ et $S(M)/u_2$ sont deux instances d'un même radical partagé d'adresse $\&(\alpha)$.*

Démonstration : Il suffit de revenir aux définitions en remarquant que le sous-terme M/u est un radical de nom α pour le système Σ^l, si et seulement si le sous-terme $S(M)/u$ est un radical dont l'étiquette de tête est $S(\tau(M)) = S(\tau(\alpha)) = \&(\alpha)$. □

La condition *2* de la définition des graphes acycliques est superflue dans un cas particulier significatif. On peut estimer qu'une situation naturelle pour le terme de départ d'une dérivation est de n'avoir que des étiquettes alphabétiques, puisque ces étiquettes résultent d'un étiquetage initial conscient, et non d'une suite de contractions de radicaux. Or, les étiquettes alphabétiques sont à la fois des étiquettes simples et des étiquettes complètes, mieux, pour tout terme M dont tous les sous-termes sont décorés par des étiquettes alphabétiques, on a $M = S(M)$ et les deux définitions des graphes acycliques (1.2.2 et 5.2.2) se confondent. Malheureusement, la condition *1* seule n'est pas conservée par réduction complète. C'est à dire qu'il est réellement nécessaire d'aller au delà de la définition intuitive des graphes acycliques, pour pouvoir envisager la mise en œuvre des dérivations complètes.

Comme dans le cas des systèmes simplement étiquetés, des restrictions sur l'étiquetage des membres droits de règles sont nécessaires.

Définition 5.2.4 (Réécriture de graphes) *Un système complètement étiqueté Σ^l est un système de réécriture de graphes, si et seulement si il obéit aux deux conditions suivantes :*

1. *Les membres droits de règles sont des graphes acycliques.*
2. *Les seules réductions autorisées sont les réductions de graphes, c'est à dire les réductions complètes $\overline{A}$: $M \to N$ telles que M est un graphe acyclique.*

On montre facilement, compte tenu du mode d'étiquetage des membres droits de règles, que vérifier la condition *1* ci-dessus revient exactement à la contrôler lors de la première étape de l'étiquetage des membres droits, quand toutes les étiquettes qui décorent les sous-termes des membres droits sont des étiquettes alphabétiques. D'ou l'on peut conclure que la procédure de vérification de la condition *1* est commune au système de réécriture Σ^l et à sa mise en œuvre $S(\Sigma^l)$.

Lemme 5.2.5 *Soit un système de réécriture complètement étiqueté Σ^l, tel que Σ^l est un système de réécriture de graphes. Alors les réductions complètes issues d'un graphe acyclique M sont exactement isomorphes aux réductions de graphes issues du graphe acyclique cohérent $S(M)$.*

Démonstration : Nous venons de voir que les conditions sur les membres droits de règles 1.3.5-1 pour le système $S(\Sigma^l)$ et 5.2.4-1 pour le système Σ^l sont équivalentes. On considère ensuite une réduction A: $M \overset{U}{\to} N$ et la réduction isomorphe par le morphisme de simplification S, notée $S(A)$: $S(M) \overset{U}{\to} S(N)$:

$$\begin{array}{ccc} M & \xrightarrow{\quad S \quad} & S(M) \\ \Big\downarrow{\scriptstyle U} & & \Big\downarrow{\scriptstyle U} \\ N & \xrightarrow{\quad S \quad} & S(N) \end{array}$$

Par le corollaire 5.2.3, l'ensemble d'occurrences U est une réunion de familles complètes de radicaux du terme M, si et seulement si c'est une réunion de radicaux partagés dans le terme $\mathcal{S}(M)$. Autrement dit, la réduction A est complète, si et seulement si la réduction $S(A)$ est une réduction de graphes. □

Il reste, pour aller plus loin, à vérifier que l'on peut définir les dérivations de graphes dans le système complètement étiqueté.

Lemme 5.2.6 *Soit un système de réécriture complètement étiqueté Σ^l, tel que Σ^l est un système de réécriture de graphes. Soit ensuite un terme M complètement étiqueté, tel que M est un graphe acyclique. Soit enfin une réduction complète $\overline{A}$: $M \to N$. Alors, le terme N est un graphe acyclique.*

Démonstration: On reprend les notations de la preuve du lemme précédent, en considérant cette fois une réduction complète $\overline{A}$ et sa mise en œuvre $S(\overline{A})$. Par ce lemme précédent, la réduction $S(\overline{A})$: $S(M) \to S(N)$ est une réduction de graphes au sens des étiquettes simples. Donc, par le lemme 1.3.6, le terme $S(N)$ est un graphe acyclique cohérent. Considérons ensuite deux sous-termes du terme N, N/u_1 et N/u_2, avec $\&(N/u_1) = \&(N/u_2)$. On distingue trois cas :

1. Il existe deux occurrences v_1 et v_2 dans le terme M, telles que l'on a $u_1 \in v_1 \setminus \overline{A}$ et $u_1 \in v_2 \setminus \overline{A}$. Alors (lemme 2.2.1), il existe deux étiquettes éventuellement vides l_1 et l_2, telles que l'on a, $\tau(N/u_1) = l_1.\tau(M/v_1)$ et $\tau(N/u_2) = l_2.\tau(M/v_2)$. D'où il vient les deux égalités entre adresses, $\&(N/u_1) = \&(M/v_1)$ et $\&(N/u_2) = \&(M/v_1)$. Puisque le terme complètement étiqueté M est un graphe acyclique, on peut déduire $M/v_1 \simeq M/v_2$. La réduction $\overline{A}$ complète réduit exactement les mêmes radicaux dans les sous-termes quasi-égaux M/v_1 et M/v_2, en outre toute réduction qui a lieu au dessus de ces sous-termes ne peut qu'intervenir en ajoutant des étiquettes élémentaires à gauche de $\tau(M/v_1)$ ou de $\tau(M/v_2)$ et donc, sans intervenir sur l'étiquette élémentaire la plus à droite, commune aux quatre sous-termes M/v_1, M/v_2, N/u_1 et N/u_2. Soit finalement, $N/u_1 \simeq N/u_2$.
2. Les deux occurrences u_1 et u_2 sont créées par la réduction $\overline{A}$. Il existe donc deux radicaux M/v_1 et M/v_2 dont la contraction par $\overline{A}$ crée les occurrences u_1 et u_2 respectivement. Soient $\alpha_1 \to \beta_1$ et $\alpha_2 \to \beta_2$ les règles de réécriture utilisées pour réduire les radicaux M/v_1 et M/v_2 respectivement. On a donc, $M_1 = l_1.\sigma_1(\alpha_1)$ et $M_2 = l_2.\sigma_2(\alpha_2)$. En outre, Il existe un héritier v'_1 de v_1 par $\overline{A}$, avec $v_1 = v'_1.w_1$ et $\tau(N/u_1) = m_1.<\alpha_1,a_{w_1}>$, où m_1 est une étiquette qui peut être vide et a_{w_1} est l'étiquette alphabétique utilisée pour décorer l'occurrence w_1 du membre droit β_1. De même, on a $u_2 = v'_2.w_2$ et $\tau(N/u_2) = m_2.<\alpha_2,a_{w_2}>$. Par conséquent, on a $\&(N/u_1) = \&(M/v_1).a_{w_1}$ et $\&(N/u_2) = \&(M/v_2).a_{w_2}$. Soit, de l'hypothèse $\&(N/u_1) = \&(N/u_2)$, on tire $\&(M/v_1) = \&(M/v_2)$ et $a_{w_1} = a_{w_2}$. C'est à dire, puisque M est un graphe acyclique, que l'on a $M/v_1 \simeq M/v_2$, $\alpha_1 = \alpha_2 = \alpha$ (par l'absence de paire critique) et $\forall x \in \mathcal{V}(\alpha)$, $\sigma_1(x) = \sigma_2(x)$. La réduction $\overline{A}$ complète contracte exactement les même radicaux dans les sous-termes M/v_1 et M/v_2, il existe donc une substitution ρ telle que l'on a : $N/v'_1 = m_1.\rho(\beta)$ et $N/v'_2 = m_2.\rho(\beta)$. Par ailleurs, Le membre droit β de la règle de réécriture $\alpha \to \beta$ est aussi un graphe acyclique, de $a_{w_1} = a_{w_2}$ on peut donc tirer la quasi égalité $\beta/w_1 \simeq \beta/w_2$, qui est ici l'égalité $\beta/w_1 = \beta/w_2$ (les membres droit ne portent que des étiquettes élémentaires). Soit finalement, $\rho(\beta/w_1) = \rho(\beta/w_2)$ et donc $N/v_1 \simeq N/v_2$. —Notons que l'on a pas forcément $N/v_1 = N/v_2$ dans le cas $w_1 = w_2 = \epsilon$—.
3. Enfin, il est impossible que l'une des occurrences u_1 ou u_2 soit un résidu tandis que l'autre est créée par la réduction $\overline{A}$. Pour voir ceci, il suffit de se placer du point de vue de la réduction simplement étiquetée $S(\overline{A})$ pour laquelle une telle situation est exclue —cf. les cas 3. et 4. de la démonstration du lemme 1.3.6—. □

Theorème 5.2.7 *Soit un système de réécriture complètement étiqueté Σ^l, tel que Σ^l est un système de réécriture de graphes. Soit ensuite un terme M complètement étiqueté, tel que M est un graphe acyclique. Alors, on a les deux résultats suivants.*

1. *Le système $S(\Sigma^l)$ est un système de réécriture de graphes.*
2. *Les dérivations complètes issues de M sont isomorphes aux dérivations de graphes issues de $S(M)$.*

Démonstration : Ce théorème n'est autre que la fermeture transitive du lemme 5.2.5 par le lemme 5.2.6. □

5.3 Coût des dérivations

Définition 5.3.1 *Soit une réduction A: $M \xrightarrow{U} N$. Par définition, le coût $\mathcal{C}(A)$ de la réduction A est le nombre de familles de radicaux dans la partition en familles de l'ensemble U.*

$$\mathcal{C}(A) = n, \quad \textit{avec} \quad U = \mathcal{F}_1 \cup \mathcal{F}_2 \cup \ldots \cup \mathcal{F}_n$$

Pour une dérivation composite AB, on pose $\mathcal{C}(AB) = \mathcal{C}(A) + \mathcal{C}(B)$. Le choix de la mesure de coût $\mathcal{C}$ revient à supposer que le coût unitaire correspond à la contraction d'une famille de radicaux. Du point de vue de la mise en œuvre des dérivations, cela veut dire qu'une famille de radicaux doit être représentée par une structure de données partagée. Or, le principal résultat de la section précédente a été de montrer que l'on peut choisir les graphes acycliques pour cette structure de données. On peut donc dire que la mesure du coût des dérivations est relative aux graphes acycliques, et comme telle raisonnable.

Il faut bien noter qu'un raisonnement aussi direct n'est pas possible dans le cas du λ-calcul. En effet, dans le λ-calcul, les résidus d'un même radical ne sont pas nécessairement disjoints et l'on conçoit bien qu'il n'est pas possible de mettre en œuvre les familles de radicaux à l'aide des graphes acycliques —Pour une démonstration complète de ce résultat, voir [Fie 90, Gen 90]—. Ce fait a deux conséquences :

1. La mise en œuvre des réductions complètes dans le λ-calcul est un problème difficile qui a fait l'objet d'études particulièrement compliquées [Kat 90, Lam 90].
2. Puisqu'elle ne correspond plus à une mise en œuvre intuitive, la définition des familles de radicaux à partir des étiquettes semble tout à fait arbitraire. C'est pourquoi, dans [Lév 78], une notion de radicaux "équivalents par réduction" est introduite et il est montré que les classes d'équivalences de cette relation sont les familles définies en suivant l'étiquetage.

Dans le cas des systèmes orthogonaux, je pense qu'une mise en œuvre des familles qui est à la fois facile et conforme à l'intuition me dispense d'en dire plus. D'une certaine façon, la simplicité de la mise en œuvre des familles de radicaux est leur justification.

Un certain nombre de résultats faciles vont maintenant illustrer la pertinence du coût.

Lemme 5.3.2 *Soient deux dérivations A: $M \to^* N$ et B: $M \to^* P$, alors, on a l'inégalité :*

$$\mathcal{C}(A \setminus B) \leq \mathcal{C}(A)$$

Démonstration : Si la dérivation A est en fait une réduction A: $M \stackrel{\mathcal{F}_1 \cup \mathcal{F}_2 \cup \ldots \cup \mathcal{F}_n}{\longrightarrow} N$, où l'entier n est le coût de la réduction A, alors la réduction $A \setminus B$ contracte les familles de radicaux $\mathcal{F}_1 \setminus B$, $\mathcal{F}_2 \setminus B$,..., $\mathcal{F}_n \setminus B$. Certaines de ces familles résidus peuvent être vides, d'où l'inégalité $\mathcal{C}(A \setminus B) \leq \mathcal{C}(A)$. Si maintenant la dérivation A est une concaténation $A = A_1 A_2$, alors, par la décomposition 4.1.3-4.2, on a $\mathcal{C}(A) = \mathcal{C}(A_1 \setminus B) + \mathcal{C}(A_2 \setminus (B \setminus A_1))$. On peut ensuite conclure par application directe de l'hypothèse d'induction aux dérivations A_1 et A_2. □

Ce résultat est tout à fait logique. Les étapes élémentaires de la dérivation $A \setminus B$ contractent moins de familles que celles de la dérivation A, puisque certains des radicaux contractés au cours de la dérivation A peuvent être contractés ou effacés au cours de la dérivation B. Par ailleurs, le choix d'une mesure de coût liée aux familles fait qu'il n'y a pas d'inconvénient à contracter les résidus d'un radical plutôt que ce radical, même dans le cas où le radical initial est dupliqué.

Lemme 5.3.3 *Soit une dérivation quelconque A, alors on a l'inégalité :*

$$\mathcal{C}(\overline{A}) \leq \mathcal{C}(A)$$

Démonstration : Dans le cas où A est une réduction qui contracte les familles $\mathcal{F}_1$, $\mathcal{F}_2$,..., $\mathcal{F}_n$, alors $\overline{A}$ est une réduction qui contracte les familles complètes $\overline{\mathcal{F}_1}$, $\overline{\mathcal{F}_2}$,..., $\overline{\mathcal{F}_n}$. On a donc $\mathcal{C}(A) = \mathcal{C}(\overline{A}) = n$. Sinon, la dérivation A est une concaténation $A = A_1 A_2$, avec $\mathcal{C}(\overline{A}) = \mathcal{C}(\overline{A_1}) + \mathcal{C}(\overline{A_2 \setminus (\overline{A_1} \setminus A_1)})$. Par hypothèse d'induction appliquée aux dérivations A_1 et $A_2 \setminus \ldots$, on a $\mathcal{C}(A) \leq \mathcal{C}(A_1) + \mathcal{C}(A_2 \setminus (\overline{A}_1 \setminus A_1))$. Il reste à conclure par le lemme 5.3.2, $\mathcal{C}(A_2 \setminus \ldots) \leq \mathcal{C}(A_2)$ et $\mathcal{C}(\overline{A}) \leq \mathcal{C}(A_1) + \mathcal{C}(A_2) = \mathcal{C}(A)$. □

Réunie avec l'inclusion $A \sqsubseteq \overline{A}$, la propriété $\mathcal{C}(\overline{A}) \leq \mathcal{C}(A)$ exposée ci-dessus montre que l'on tire un bénéfice en utilisant la complétion $\overline{A}$ à la place de la dérivation A. En effet, la complétion $\overline{A}$ prolonge la dérivation A, et ce pour un coût qui qui peut être moindre.

5.4 Stratégies dans les dérivations complètes

5.4.1 Familles nécessaires

Définition 5.4.1 *Soit un terme M complètement étiqueté. Par définition, une famille complète de radicaux $\overline{\mathcal{F}}$ dans M est une famille nécessaire et on note $\overline{\mathcal{F}} \in \overline{\mathcal{N}}(M)$, si et seulement si, pour toute dérivation complète $\overline{A}$: $M \rightarrow^* N$, on a $\overline{\mathcal{F}} \in \overline{\mathcal{R}}(\overline{A})$ ou $\overline{\mathcal{F}} \setminus \overline{A} \neq \emptyset$.*

Les familles nécessaires peuvent être caractérisées par une propriété de leur nom. Pour ce faire, on doit tout d'abord examiner précisément l'origine des noms qui contribuent au terme final d'une dérivation.

Lemme 5.4.2 *Soit une réduction A: $M \rightarrow N$. Soit α un nom de radical qui contribue au terme N ($\alpha <^+ N$). Alors, on a deux possibilités :*

1. *Le nom α contribue au terme M ($\alpha <^+ M$).*
2. *Il existe un radical de nom α qui est contracté par la réduction A.*

Démonstration: Si le nom α contribue au terme N, alors, par définition, il existe un sous-terme N/u dont l'étiquette de tête $\tau(N/u)$ est de la forme $\tau(N/u) = l_1.{<}\gamma,a{>}.l_2$, où l_1 et l_2 sont des étiquettes éventuellement vides, a est une étiquette alphabétique et γ est un nom de radical avec $\alpha <^* \gamma$. Par le lemme 5.4.2, on a deux possibilités :

1. Il y a un un sous-terme M/v dont l'étiquette de tête est de la forme $\tau(M/v) = m_1.<\gamma,a>.m_2$. Auquel cas, le nom α contribue au terme M.
2. Ou bien, le nom γ est le nom d'un radical contracté par la réduction A. Auquel cas, le nom α contribue au terme M (si on a $\alpha <^+ \gamma$), ou bien un radical de nom α est contracté par la réduction A (si on a $\alpha = \gamma$). □

Lemme 5.4.3 *Soit une dérivation A: $M \to^* N$ et soit un nom de radical α qui contribue au terme N ($\alpha <^+ N$). Alors, il y a deux possibilités :*

1. *Le nom α contribue au terme initial M ($\alpha <^+ M$).*
2. *Il existe un radical d'occurrence u dans le terme M, qui est contracté par la dérivation A ($u \in \mathcal{R}(A)$), et dont le nom γ est égal ou contribue au nom α ($\gamma <^* \alpha$)*

Démonstration: On raisonne par induction sur $|A|$. Si la dérivation A est la dérivation vide, alors la propriété *1* est toujours vérifiée. Sinon, on décompose la dérivation A comme $A = BC$, avec B: $M \to P$ et C: $P \to^* N$. Par hypothèse d'induction appliquée à la dérivation C, on a deux possibilités :

1. Le nom α contribue au terme P. Alors, le résultat vient directement par le lemme 5.4.2 ci-dessus et par l'inclusion $\mathcal{R}(B) \subseteq \mathcal{R}(A)$.
2. Il existe dans le terme P, un radical R de nom $\delta <^* \alpha$ qui est contracté par la dérivation C. Alors, par le lemme 2.4.3, il y a deux possibilités mutuellement exclusives :
   (a) Le radical R est le résidu par la réduction B d'un radical S d'occurrence u dans le terme M. Par définition, on a $u \in \mathcal{R}(A)$ et, par le lemme 2.3.8, le radical S est de nom δ. On pose donc ici $\gamma = \delta$.
   (b) Il existe un radical S d'occurrence u et de nom γ dans le le terme M qui est un créateur du radical R. On a bien ici $u \in \mathcal{R}(A)$, puisque le radical S est contracté par la première étape B de la dérivation A. En outre, par le lemme 2.4.5, il vient $\gamma < \delta$. Soit finalement, on a $\gamma <^+ \alpha$ par définition de la relation $<^+$. □

Lemme 5.4.4 *Soit un terme cohérent M. Pour toute famille complète $\overline{\mathcal{F}}$ de radicaux de nom α dans le terme M, on a les équivalences :*

$$\forall \overline{A}\colon M \to^* N,\ \overline{\mathcal{F}} \backslash\!\backslash \overline{A} \neq \emptyset \Leftrightarrow \forall \overline{A}\colon M \to^* N,\ \alpha <^* N \Leftrightarrow \overline{\mathcal{F}} \in \overline{\mathcal{N}}(M)$$

Démonstration : On numérote (1), (2) et (3) les propositions ci-dessus. L'implication (1) $\Rightarrow$ (2) est une conséquence directe des lemmes 2.3.8 et 2.5.4 sur l'étiquetage des résidus et des héritiers. On montre ensuite l'implication (2) $\Rightarrow$ (3). On suppose donc que, pour toute dérivation complète $\overline{A}$: $M \to^* N$, on a $\alpha <^* N$. Si le nom α est le nom d'un radical R dans le terme N, alors, puisque le terme M est cohérent, le radical R est un résidu d'un membre de la famille complète $\overline{\mathcal{F}}$ (lemme 2.5.3), c'est à dire que l'on a $\overline{\mathcal{F}} \setminus \overline{A} \neq \emptyset$. Sinon, on a $\alpha <^+ N$. Dès lors, par le lemme 5.4.3, il existe dans le terme initial M un radical R de nom $\gamma <^* \alpha$ qui est contracté par la dérivation $\overline{A}$. Par la cohérence de M, on a nécessairement $\gamma = \alpha$. Enfin, par le lemme 5.1.9, la famille $\overline{\mathcal{F}}$, dont un des membres est contracté par la dérivation complète $\overline{A}$, est une famille contractée par $\overline{A}$. Enfin, on montre la contraposée $\neg(1) \Rightarrow \neg(3)$. Soit donc $\overline{\mathcal{F}}$, une famille complète de radicaux du terme M, telle qu'il existe une dérivation complète $\overline{A}$: $M \to^* N$, avec $\overline{\mathcal{F}} \backslash\!\backslash \overline{A} = \emptyset$. On considère un équivalent standard B de la dérivation $\overline{A}$. Par l'invariance des descendants par dérivations équivalentes, on a $\overline{\mathcal{F}} \backslash\!\backslash B = \emptyset$. Puisque la dérivation B est standard, on montre comme dans la preuve du lemme 4.4.13 que pour toute occurrence u prise dans $\overline{\mathcal{F}}$, on a $u \setminus B = \emptyset$ et $u \notin \mathcal{R}(B)$. On considère ensuite la complétion $\overline{B}$ de la dérivation B. Par l'inclusion $B \sqsubseteq \overline{B}$, il existe une dérivation C, avec $BC \equiv \overline{B}$. Soit (lemme 4.2.11), $\overline{\mathcal{F}} \setminus \overline{B} = (\overline{\mathcal{F}} \setminus B) \setminus C = \emptyset$. En outre, par le lemme 5.1.11 et la condition $\forall u \in \overline{\mathcal{F}},\ u \notin \mathcal{R}(B)$, on a $\overline{\mathcal{F}} \notin \overline{B}$. Soit finalement, la famille $\overline{\mathcal{F}}$ n'est pas nécessaire à partir du terme M. $\square$

Enfin, une famille complète de radicaux est nécessaire, si et seulement si elle contient un radical nécessaire.

Lemme 5.4.5 *Soit un terme M. Pour toute familles complète $\overline{\mathcal{F}}$ de radicaux du terme M, on a l'équivalence :*

$$\overline{\mathcal{F}} \in \overline{\mathcal{N}}(M) \Leftrightarrow \overline{\mathcal{F}} \cap \mathcal{N}(M) \neq \emptyset$$

Démonstration : Si il existe une occurrence de radical u qui est à la fois membre de la famille $\overline{\mathcal{F}}$ et nécessaire à partir du terme M, alors, par définition de $\mathcal{N}$ et pour toute dérivation complète $\overline{A}$: $M \to^* N$, on a $u \setminus \overline{A} \neq \emptyset$ —auquel cas on a $\overline{\mathcal{F}} \setminus \overline{A} \neq \emptyset$—, ou bien $u \in \mathcal{R}(\overline{A})$ —auquel cas on a $\overline{\mathcal{F}} \cap \mathcal{R}(\overline{A}) \neq \emptyset$ et donc $\overline{\mathcal{F}} \in \overline{\mathcal{R}}(\overline{A})$ (lemme 5.1.9)—. Réciproquement, soit une famille non-vide et complète $\overline{\mathcal{F}}$ qui ne contient pas de radical nécessaire à partir du terme M. Par le lemme 4.4.13, pour toute occurrence de radical qui n'est pas dans $\mathcal{N}(M)$, il existe une dérivation A_u, telle que l'on a $u \setminus\!\!\setminus A_u = \emptyset$. On pose $\overline{\mathcal{F}} = \{u_1, u_2, \ldots, u_n\}$. Par hypothèse, il existe n dérivations $A_1, A_2, \ldots, A_n$, telles que, pour tout entier i compris entre 1 et n, on a l'égalité $u_i \setminus\!\!\setminus A_i = \emptyset$. On considère ensuite la dérivation $A = A_1 \sqcup A_2 \sqcup \ldots \sqcup A_n$. Par le lemme des déplacements parallèles généralisés, pour tout entier i compris entre 1 et n, on a $u_i \setminus\!\!\setminus A = \emptyset$. C'est à dire que l'on a $\overline{\mathcal{F}} \setminus\!\!\setminus A = \emptyset$. On considère ensuite la complétion $\overline{A}$ de la dérivation A. De par l'inclusion $A \sqsubseteq \overline{A}$, il existe une dérivation B, avec $AB \equiv \overline{A}$. Soit finalement, par invariance des descendants selon des dérivations équivalentes, $\overline{\mathcal{F}} \setminus\!\!\setminus \overline{A} = (\overline{\mathcal{F}} \setminus\!\!\setminus A) \setminus\!\!\setminus B = \emptyset$ et donc $\overline{\mathcal{F}} \notin \overline{\mathcal{N}}(M)$ (lemme 5.4.4). $\square$

Par ce dernier lemme, si on connaît un radical nécessaire à partir d'un terme M donné, on connaît également une famille nécessaire à partir du terme M. Par conséquent, il existe une famille nécessaire dans tout terme qui n'est pas en forme normale.

Ce lemme est un peu surprenant car il revient à remplacer une quantification universelle sur les dérivations quelconques par une quantification universelle sur les dérivations complètes. Il justifie la définition suivante des familles standards.

Définition 5.4.6 *Soit un terme cohérent M. Par définition, une famille complète $\overline{\mathcal{F}}$ du terme M est standard à partir du terme M, et on note $\overline{\mathcal{F}} \in \overline{\mathcal{E}}(M)$, si et seulement si la famille $\overline{\mathcal{F}}$ contient un radical standard à partir du terme M.*

5.4.2 Appel complet par nécessité, appel paresseux

Définition 5.4.7 *Soit un terme M cohérent. Par définition, une dérivation complète $\overline{A}$: $M \to^* N$ suit la stratégie de l'appel complet par nécessité, si et seulement si la dérivation $\overline{A}$ est la dérivation vide, ou si la dérivation $\overline{A}$ est la concaténation $\overline{A} = \overline{B}\,\overline{C}$, où $\overline{C}$: $P \to^* N$ est un appel complet par nécessité et $\overline{B}$ est une réduction complète, avec :*

$$\overline{B}\colon M \stackrel{\overline{\mathcal{F}_1} \cup \overline{\mathcal{F}_2} \cup \ldots \cup \overline{\mathcal{F}_n}}{\longrightarrow} N \text{ où } \overline{\mathcal{F}_1},\ \overline{\mathcal{F}_2}, \ldots, \overline{\mathcal{F}_n} \text{ sont des familles de } \overline{\mathcal{N}}(M)$$

Il faut noter que, contrairement aux appels par nécessité, les appels complets par nécessité peuvent comprendre des étapes vides. Par ailleurs, il n'est pas non plus obligatoire d'avoir atteint une forme normale pour pouvoir arrêter la stratégie complète. Ces différences sont purement techniques, elles sont indispensables pour pouvoir structurer l'ensemble des appels complets par nécessité issus d'un terme donné.

Il est clair que les appels complets par nécessité sont stables par concaténation.

Corollaire 5.4.8 *Soient un terme cohérent M, ainsi que deux dérivations complètes $\overline{A}$: $M \to^* N$ et $\overline{B}$: $N \to^* P$. Alors, la dérivation $\overline{A}\,\overline{B}$ est un appel complet par nécessité, si et seulement si les dérivations $\overline{A}$ et $\overline{B}$ sont des appels complets par nécessité.*

Les appels complets par nécessité sont aussi stables par l'opération de résidu, pourvu que l'on considère les résidus par une dérivation complète.

Lemme 5.4.9 *Soient $\overline{A}$: $M \to^* N$ un appel complet par nécessité, et $\overline{B}$: $M \to^* P$ une dérivation complète quelconque, tous deux issus du même terme cohérent M. Alors, la dérivation résidu $\overline{A} \setminus \overline{B}$ est un appel complet par nécessité.*

Démonstration : La preuve est par induction sur la longueur $|\overline{A}|$. Si $\overline{A} = O$, alors la dérivation $\overline{A} \setminus \overline{B} = O$ est un appel complet par nécessité. Sinon $\overline{A}$ est de la forme $\overline{A} = \overline{A}_1 \overline{A}_2$, où la réduction $\overline{A_1}$: $M \to Q$ est telle que $\overline{\mathcal{R}}(\overline{A}_1) \subseteq \overline{\mathcal{N}}(M)$; tandis que la dérivation $\overline{A}_2$: $Q \to^* N$ est un appel complet par nécessité. Par la décomposition 4.1.3-4.2, on a :

$$\overline{A} \setminus \overline{B} = (\overline{A}_1 \setminus \overline{B})\,(\overline{A}_2 \setminus (\overline{B} \setminus \overline{A}_1))$$

Le terme Q est cohérent (lemme 2.6.2) et la dérivation $\overline{B} \setminus \overline{A}_1$ est complète (lemme 5.1.6), on peut donc appliquer l'hypothèse d'induction à l'appel complet par nécessité $\overline{A}_2$ et prouver que la dérivation $\overline{A}_2 \setminus (\overline{B} \setminus \overline{A}_1)$ est un appel complet par nécessité. Reste à montrer que la réduction $\overline{A}_1 \setminus \overline{B}$ contracte des familles nécessaires à partir du terme P. Soit une famille complète $\overline{\mathcal{F}}$, contractée par la réduction $\overline{A}_1 \setminus \overline{B}$. Par le lemme 5.1.6, la famille $\overline{\mathcal{F}}$ est bien complète et il existe une famille complète $\overline{\mathcal{F}}'$ contractée par la réduction $\overline{A}_1$, avec $\overline{\mathcal{F}} = \overline{\mathcal{F}}' \setminus \overline{B}$. Par hypothèse, on a $\overline{\mathcal{F}}' \in \overline{\mathcal{N}}(M)$. Pour toute dérivation complète $\overline{C}$: $P \to^* R$ issue du terme P, on a donc $\overline{\mathcal{F}} \in \overline{\mathcal{R}}(\overline{B}\,\overline{C})$. Or, on ne peut pas avoir $\overline{\mathcal{F}}' \in \overline{\mathcal{R}}(\overline{B})$, puisque l'on a $\overline{\mathcal{F}}' \setminus \overline{B} \neq \emptyset$ (lemme 5.1.10). On a donc nécessairement $\overline{\mathcal{F}}' \setminus \overline{A} \in \overline{\mathcal{R}}(\overline{C})$. Soit finalement, pour toute dérivation complète $\overline{C}$ issue du terme P, on a $\overline{\mathcal{F}} \in \overline{\mathcal{R}}(\overline{C})$. C'est à dire que $\overline{\mathcal{F}}$ est une famille nécessaire à partir du terme P. □

En conclusion, les classes de l'équivalence $\equiv$ restreinte aux appels complets par nécessité issus d'un terme M cohérent donné, forment un sous-treillis du sup-demi treillis des dérivations complètes issues du terme M.

On définit également la stratégie d'appel paresseux qui est la version complète de l'appel par nom.

Définition 5.4.10 (Appel paresseux) *Soit un terme M cohérent. Par définition, une dérivation complète $\overline{A}$: $M \to^* N$ suit la stratégie de l'appel paresseux, si et seulement si la dérivation $\overline{A}$ est la dérivation vide, ou si la dérivation $\overline{A}$ est la concaténation $\overline{A} = \overline{B}\,\overline{C}$, où $\overline{C}$: $P \to^* N$ est un appel paresseux et $\overline{B}$ est une réduction complète, avec :*

$$\overline{B}\colon M \xrightarrow{\overline{\mathcal{F}_1} \cup \overline{\mathcal{F}_2} \cup \ldots \cup \overline{\mathcal{F}_n}} N \text{ où } \overline{\mathcal{F}_1},\ \overline{\mathcal{F}_2}, \ldots, \overline{\mathcal{F}_n} \text{ sont des familles de } \overline{\mathcal{E}}(M)$$

Il est clair, par l'inclusion $\mathcal{E}(M) \subseteq \mathcal{N}(M)$, et le lemme 5.4.5, que l'appel paresseux est un appel complet par nécessité particulier. La dénomination "appel paresseux" est conforme à l'usage qui y voit la mise en œuvre avec partage de l'appel par nom. En effet, rappelons que les familles de radicaux s'identifient exactement aux radicaux partagés dans la mise en œuvre de la réécriture de termes à l'aide des graphes acycliques.

5.5 Optimalité

5.5.1 Forme générale de la propriété d'optimalité

La structure de demi-treillis des appels complets par nécessité permet une preuve élégante d'une forme très générale de la propriété d'optimalité.

Lemme 5.5.1 (Optimalité) *Soient un appel complet par nécessité $\overline{A}$: $M \to^* N$ et une dérivation complète $\overline{B}$: $M \to^* P$, tous deux issus du même terme cohérent M. Alors, on a l'inégalité suivante :*

$$\mathcal{C}(\overline{A} \sqcup \overline{B}) \leq \mathcal{C}(\overline{B} \sqcup \overline{A})$$

Démonstration : La preuve est par induction sur la somme $|\overline{A}| + |\overline{B}|$. Si l'une des dérivations $\overline{A}$ ou $\overline{B}$ est vide, alors on a l'égalité $\mathcal{C}(\overline{A} \sqcup \overline{B}) = \mathcal{C}(\overline{B} \sqcup \overline{A})$. Sinon, la dérivation $\overline{A}$ se décompose en $\overline{A}_1\overline{A}_2$, où la réduction $\overline{A}_1$ contracte les familles complètes $\overline{\mathcal{F}}_1, \overline{\mathcal{F}}_2, \ldots, \overline{\mathcal{F}}_n$; tandis que la dérivation $\overline{B}$ se décompose en $\overline{B}_1\overline{B}_2$, où la réduction $\overline{B}_1$ contracte possiblement certaines des familles contractées par la réduction $\overline{A}_1$, ainsi que d'autres familles. On suppose que la réduction $\overline{B}_1$ contracte les k familles $\overline{\mathcal{F}}_1, \overline{\mathcal{F}}_2, \ldots, \overline{\mathcal{F}}_k$, plus m autres familles $\overline{\mathcal{G}}_1, \overline{\mathcal{G}}_2, \ldots, \overline{\mathcal{G}}_m$. On a naturellement $\mathcal{C}(\overline{A}_1) = n$ et $\mathcal{C}(\overline{B}_1) = k + m$. On pose ensuite (cf. la figure 4.3).

$$\begin{array}{llll}
\overline{A}'_1 = \overline{A}_1 \setminus \overline{B}_1 & \overline{B}'_1 = \overline{B}_1 \setminus \overline{A}_1 & \overline{A}'_2 = \overline{A}_2 \setminus \overline{B}'_1 & \overline{B}'_2 = \overline{B}_2 \setminus \overline{A}' \\
\overline{A}''_1 = \overline{A}'_1 \setminus \overline{B}_2 & \overline{A}''_2 = \overline{A}'_2 \setminus \overline{B}'_1 & \overline{B}''_1 = \overline{B}'_1 \setminus \overline{A}_1 & \overline{B}''_2 = \overline{B}'_2 \setminus \overline{A}'_1
\end{array}$$

On considère tout d'abord le carré limité par les quatre réductions $\overline{A}_1$, $\overline{B}'_1$, $\overline{B}_1$ et $\overline{A}'_1$. Pour tout entier i compris entre 1 et k, on a $\overline{\mathcal{F}}_i \setminus \overline{A}_1 = \emptyset$, car la réduction $\overline{A}_1$ contracte la famille $\overline{\mathcal{F}}_i$. Par conséquent, seules les familles du type $\overline{\mathcal{G}}_i \setminus \overline{A}_1$ peuvent être non-vides et contractées par la réduction résidu $\overline{B}'_1$. En d'autres termes, on a l'inégalité $\mathcal{C}(\overline{B}'_1) \leq m$. Soit maintenant un entier i compris entre $k+1$ et n. De par l'identité des résidus par dérivations équivalentes et puisque la réduction $\overline{A}_1$ contracte la famille $\overline{\mathcal{F}}_i$, on a $\overline{\mathcal{F}}_i \setminus (\overline{B}_1\overline{A}'_1) = \overline{\mathcal{F}}_i \setminus (\overline{A}_1\overline{B}'_1) = (\overline{\mathcal{F}}_i \setminus \overline{A}_1) \setminus \overline{B}'_1 = \emptyset$. Or, la famille $\overline{\mathcal{F}}_i$ est nécessaire à partir du terme M, elle doit donc être contractée par la dérivation $\overline{B}_1\overline{A}'_1$. Par hypothèse ($i > k$), la famille $\overline{\mathcal{F}}_i$ n'est pas contractée par la réduction $\overline{B}_1$, les résidus $\overline{\mathcal{F}}_i \setminus \overline{B}_1$ forment donc une famille non-vide qui est contractée par la réduction $\overline{A}'_1$. En revanche, la réduction $\overline{A}'_1$ ne peut pas contracter de résidu de l'une des k premières familles $\overline{\mathcal{F}}_i$, puisque celles-ci sont contractées par la réduction $\overline{B}$. Du point de vue des coûts, on a $\mathcal{C}(\overline{A}'_1) = n - k$. En regroupant les résultats déjà vus, on a :

$$\mathcal{C}(\overline{A}_1) + \mathcal{C}(\overline{B}'_1) \leq n + m \quad \text{et} \quad \mathcal{C}(\overline{B}_1) + \mathcal{C}(\overline{A}'_1) = (m + k) + (n - k)$$

Soit finalement :

$$\mathcal{C}(\overline{A}_1\overline{B}'_1) = \mathcal{C}(\overline{A}_1 \sqcup \overline{B}_1) \leq \mathcal{C}(\overline{B}_1 \sqcup \overline{A}_1) = \mathcal{C}(\overline{B}'_1\overline{A}_1)$$

La dérivation $\overline{A}_2$ est un appel complet par nécessité par définition ; tandis que les dérivations $\overline{A}'_1$ et $\overline{A}'_2$ sont des appels complets par nécessité par le lemme 5.4.9. On peut donc utiliser l'hypothèse d'induction pour prouver les trois inéquations :

$$\begin{array}{l}
\mathcal{C}(\overline{A}'_1\overline{B}'_2) = \mathcal{C}(\overline{A}'_1 \sqcup \overline{B}_2) \leq \mathcal{C}(\overline{B}_2 \sqcup \overline{A}'_1) = \mathcal{C}(\overline{B}_2\overline{A}''_1) \\
\mathcal{C}(\overline{A}_2\overline{B}''_1) = \mathcal{C}(\overline{A}_2 \sqcup \overline{B}'_1) \leq \mathcal{C}(\overline{B}'_1 \sqcup \overline{A}_2) = \mathcal{C}(\overline{B}'_1\overline{A}'_2) \\
\mathcal{C}(\overline{A}'_2\overline{B}''_2) = \mathcal{C}(\overline{A}'_2 \sqcup \overline{B}'_2) \leq \mathcal{C}(\overline{B}'_2 \sqcup \overline{A}'_2) = \mathcal{C}(\overline{B}'_2\overline{A}''_2)
\end{array}$$

Il reste à utiliser les regroupements adéquats des étapes des dérivations $\overline{A} \sqcup \overline{B}$ et $\overline{B} \sqcup \overline{A}$:

$$\begin{array}{rcl}
\mathcal{C}(\overline{A} \sqcup \overline{B}) & = & \mathcal{C}(\overline{A}_1) + \mathcal{C}(\overline{A}_2\overline{B}''_1) + \mathcal{C}(\overline{B}''_2) \\
& \leq & \mathcal{C}(\overline{A}_1\overline{B}'_1) + \mathcal{C}(\overline{A}'_2\overline{B}''_2) \\
& \leq & \mathcal{C}(\overline{B}_1) + \mathcal{C}(\overline{A}'_1\overline{B}'_2) + \mathcal{C}(\overline{A}''_2) \\
& \leq & \mathcal{C}(\overline{B}_1\overline{B}_2) + \mathcal{C}(\overline{A}''_1\overline{A}''_2) \\
& = & \mathcal{C}(\overline{B} \sqcup \overline{A})
\end{array}$$

□

En fait, la propriété d'optimalité reste valable quand la dérivation non-optimale (la dérivation $\overline{B}$) n'est pas complète. J'ai donné une version particulière du théorème d'optimalité dans le cas des dérivations complètes, car sa preuve exploite pleinement les familles de radicaux et la structure du treillis des dérivations complètes.

Theorème 5.5.2 *Soit un terme M cohérent. Soient un appel complet par nécessité $\overline{A}$: $M \to^* N$ et une dérivation B: $M \to^* P$. Alors, on a l'inégalité suivante :*

$$\mathcal{C}(\overline{A} \sqcup B) \leq \mathcal{C}(B \sqcup \overline{A})$$

Démonstration : On utilise ici l'induction sur la longueur de la dérivation $\overline{A}$. Si la dérivation $\overline{A}$ est vide, alors on a $\overline{A} \sqcup B = B \sqcup \overline{A} = B$. Sinon, on décompose la dérivation $\overline{A}$ comme $\overline{A} = \overline{A}_1\overline{A}_2$, ou $\overline{A}_1$ est une réduction qui contracte les familles complètes nécessaires $\overline{\mathcal{F}}_1, \overline{\mathcal{F}}_2, \ldots, \overline{\mathcal{F}}_n$. On peut toujours numéroter les $\overline{\mathcal{F}}_i$, afin que les k premières familles $\overline{\mathcal{F}}_1, \overline{\mathcal{F}}_2, \ldots, \overline{\mathcal{F}}_k$ aient un de leurs membres contracté par la dérivation B. On décompose la dérivation B en étapes élémentaires $B = B_1 B_2 \ldots B_b$. Chaque étape élémentaire B_i contracte des radicaux qui sont des résidus de membres d'une des familles $\mathcal{F}_j$ et d'autres radicaux. Ce qui correspond, en regroupant les radicaux contractés par familles, à un coût $\mathcal{C}(B_i) = k_i + m_i$. On a bien entendu l'égalité $\mathcal{C}(B) = \sum_{i=1}^{b} k_i + \sum_{i=1}^{b} m_i$. On pose $m = \sum_{i=1}^{b} m_i$. Par définition même de l'entier k, on a $\sum_{i=1}^{b} k_i \geq k$ (il n'y a pas forcément égalité, car des étapes élémentaire distinctes peuvent contracter des résidus de membres d'une même famille initiale $\overline{\mathcal{F}}_j$). C'est à dire que l'on a l'inégalité $\mathcal{C}(B) \geq k + m$. Considérons maintenant la dérivation résidu $B' = B \setminus \overline{A}_1$. Par définition et pour toute étape élémentaire B'_i de la dérivation B', on a $\mathcal{R}(B'_i) = \mathcal{R}(B_i) \setminus (\overline{A}_1 \setminus B[1, i-1])$. On a donc à priori, $\mathcal{C}(B'_i) \leq m_i + k_i$. Cette décomposition correspondant à k_i familles incluses dans une famille complète de la forme $(\overline{\mathcal{F}}_j \setminus B[1, i-1]) \setminus (\overline{A}_1 \setminus B[1, i-1]) = \overline{\mathcal{F}}_j \setminus (B[1, i-1] \sqcup \overline{A}_1)$ et à m_i autres familles. Or, il n'est pas possible que la réduction B'_i contracte un résidu par la dérivation $B[1, i-1] \sqcup \overline{A}_1$ d'un membre d'une famille $\overline{\mathcal{F}}_j$, car on a $\overline{\mathcal{F}}_j \setminus (B[1, i-1] \sqcup \overline{A}_1) = \overline{\mathcal{F}}_j \setminus (\overline{A}_1 \sqcup B[1, i-1]) = (\overline{\mathcal{F}}_j \setminus \overline{A}_1) \setminus B'[1, i-1] = \emptyset$. Du point de vue des coûts, on a donc forcément $\mathcal{C}(B'_i) \leq m_i$. Soit globalement, $\mathcal{C}(B') \leq m$. Considérons ensuite la réduction $\overline{A}'_1 = \overline{A}_1 \setminus B$. Pour toute entier j de l'intervalle $[k+1 \ldots n]$, il existe une occurrence de radical $u_j \in \overline{\mathcal{F}}_j$ qui est dans $\mathcal{N}(M)$ ($\overline{\mathcal{F}}_j \in \overline{\mathcal{N}}(M)$ et lemme 5.4.5). Or, on a $u_j \setminus (B \sqcup \overline{A}_1) = u_j \setminus (\overline{A}_1 \sqcup B) = \emptyset$. Par définition des radicaux nécessaires, on a donc $u_j \in \mathcal{R}(B \sqcup \overline{A}_1)$. Par hypothèse ($j > k$), le radical d'occurrence initiale u_j n'est pas contracté par la dérivation B. Un de ses résidus est donc nécessairement contracté par la réduction $\overline{A}'_1$ et on a $\mathcal{C}(\overline{A}'_1) \geq n - k$ (il n'y a pas nécessairement égalité, car tous les résidus des familles $\overline{\mathcal{F}}_j$ pour $j \in [1 \ldots k]$ ne sont pas forcément contractés ou effacés par la dérivation B). En regroupent les résultats obtenus jusqu'ici, on a :

$$\mathcal{C}(\overline{A}_1 \sqcup B) = \mathcal{C}(\overline{A}_1) + \mathcal{C}(B') \leq n + m \quad \text{et} \quad \mathcal{C}(B \sqcup \overline{A}_1) = \mathcal{C}(B) + \mathcal{C}(\overline{A}'_1) \geq (m + k) + (n - k)$$

Soit finalement, $\mathcal{C}(\overline{A}_1 \sqcup B) \leq \mathcal{C}(B \sqcup \overline{A}_1)$ (1). On a ensuite les décompositions :

$$\overline{A} \sqcup B = \overline{A}_1\overline{A}_2((B \setminus \overline{A}_1) \setminus \overline{A}_2) \quad \text{et} \quad B \sqcup \overline{A} = B(\overline{A}_1 \setminus B)\,(\overline{A}_2 \setminus (B \setminus \overline{A}_1))$$

Soit, du point de vue des coûts :

$$\begin{array}{l} \mathcal{C}(\overline{A} \sqcup B) = \mathcal{C}(\overline{A}_1 \sqcup B) - \mathcal{C}(B \setminus \overline{A}_1) + \mathcal{C}(\overline{A}_2 \sqcup (B \setminus \overline{A}_1)) \\ \mathcal{C}(B \sqcup \overline{A}) = \mathcal{C}(B \sqcup \overline{A}_1) + \mathcal{C}((B \setminus \overline{A}_1) \sqcup \overline{A}_2) - \mathcal{C}(B \setminus \overline{A}_1) \end{array}$$

Il reste pour conclure, à appliquer l'inégalité (1) et l'hypothèse d'induction à l'appel complet par nécessité $\overline{A}_2$. □

Le théorème d'optimalité exprime que l'on a toujours intérêt à commencer les dérivations par des étapes qui contractent des familles complètes et nécessaires, plutôt que tout autre radical ou famille de radicaux. La section suivante décrit les conséquences bénéfiques de ce principe dans le cas des dérivations issues d'un terme normalisable, ou qui possède une forme normale de tête.

5.5.2 Dérivations optimales particulières

La stratégie d'appel complet par nécessité (un peu modifiée, pour éviter les séquences infinies d'étapes vides et ne terminer que sur une forme normale) est une stratégie correcte pour le calcul de la forme normale, car c'est presque une stratégie d'appel par nécessité. En effet, par le lemme 4.4.9, un appel complet par nécessité contracte au moins un radical nécessaire à chaque étape. Mais la stratégie de l'appel complet par nécessité est surtout une stratégie optimale pour le calcul de la forme normale.

Proposition 5.5.3 *Soit un terme M cohérent et normalisable, alors tout appel complet par nécessité est une dérivation normalisante qui a le plus petit coût possible.*

Démonstration : On sait déjà que tout appel complet par nécessité issu de M est une dérivation normalisante $\overline{A}$: $M \to^* N$. Soit une autre dérivation normalisante B: $M \to^* N$. Les dérivations $\overline{A}$ et B normalisantes sont équivalentes (lemme 4.2.13). Autrement dit, les dérivations $\overline{A} \setminus B$ et $B \setminus \overline{A}$ sont nulles et on a $\mathcal{C}(\overline{A}) = \mathcal{C}(\overline{A} \sqcup B)$ et $\mathcal{C}(B) = \mathcal{C}(B \sqcup \overline{A})$. Soit, par le théorème 5.5.2, $\mathcal{C}(\overline{A}) \leq \mathcal{C}(B)$. □

La stratégie d'appel paresseux est optimale dans un sens plus fin. On la précise légèrement, en interdisant les séquences infinies d'étapes vides et en imposant de poursuivre jusqu'à atteindre une forme normale de tête.

Proposition 5.5.4 *Soit un terme M cohérent et possédant une forme normale de tête. Alors, toute dérivation qui suit la stratégie de l'appel paresseux calcule une forme normale de tête de M, et ce pour un coût minimum.*

Démonstration : Tout d'abord, les appels paresseux qui sont presque des appels par nom terminent et sont de la forme $\overline{A}$: $M \to^* N$, où N est un terme en forme normale de tête. On montre ensuite par induction sur la longueur de la dérivation $\overline{A}$ qu'elle est incluse dans toute dérivation $\overline{B}$: $M \to^* P$ complète et normalisante de tête. Le cas de base $\overline{A} = O$ est trivial. Dans le cas général, on pose $\overline{A} = \overline{A}_1\overline{A}_2$, où $\overline{A}_1$ est une réduction qui contracte les familles standards $\overline{\mathcal{F}}_1, \overline{\mathcal{F}}_2, \ldots, \overline{\mathcal{F}}_n$ et $\overline{A}_2$ est un appel paresseux. Puisque la dérivation $\overline{A}$ est non-nulle, le terme M n'est pas en forme normale de tête. Par conséquent le lemme 4.4.20, s'applique à la la dérivation $\overline{B}$ qui relie un terme qui n'est pas en forme normale de tête à une forme normale de tête. On a donc l'inclusion $\mathcal{E}(M) \subseteq \mathcal{E}xt(\overline{B})$. C'est à dire que toute les radicaux standards à partir du terme M sont contractés par la dérivation $\overline{B}$. Soit, par définition des familles standards et par le lemme 5.1.9, la dérivation complète $\overline{B}$ contracte toutes les familles $\overline{\mathcal{F}}_i$. Soit finalement, $\overline{A}_1 \setminus \overline{B} = \emptyset$. Par hypothèse d'induction on a également $\overline{A}_2 \setminus (\overline{B} \setminus \overline{A}_1) = \emptyset^{|\overline{A}_2|}$ et donc, $\overline{A} \setminus \overline{B} = (\overline{A}_1 \setminus \overline{B})\,(\overline{A}_2 \setminus (\overline{B} \setminus \overline{A}_1)) = \emptyset^{|A|}$. Soit encore, $\overline{A} \sqsubseteq \overline{B}$.

Les appels paresseux sont un cas particulier d'appel complet par nécessité et vérifient l'inéquation d'optimalité. Compte tenu de la situation particulière $\overline{A} \sqsubseteq \overline{B}$, on a ici :

$$\mathcal{C}(\overline{A}) + \mathcal{C}(\overline{B} \setminus \overline{A}) = \mathcal{C}(\overline{A} \sqcup \overline{B}) \leq \mathcal{C}(\overline{B} \sqcup \overline{A}) = \mathcal{C}(\overline{B}) + 0$$

Soit l'inéquation particulière $\mathcal{C}(\overline{A}) \leq \mathcal{C}(\overline{B})$.

Considérons ensuite une dérivation normalisante de tête B: $M \to^* Q$ qui n'est pas forcément complète. Alors, sa complétion $\overline{B}$: $M \to^* R$ est aussi une dérivation normalisante de tête. En effet, puisque l'on a l'inclusion $B \sqsubseteq \overline{B}$, il existe une dérivation C: $Q \to^* R$ et il ne peut y avoir de dérivation non-interne issue du terme R. En outre, par le lemme 5.3.3, on a $\mathcal{C}(\overline{B}) \leq \mathcal{C}(B)$. Soit, par le résultat du paragraphe précédent, tout appel paresseux $\overline{A}$ et toute dérivation normalisante de tête B vérifient l'inéquation $\mathcal{C}(\overline{A}) \leq \mathcal{C}(B)$ □

Accessoirement, on a prouvé que tous les appels paresseux issus de M sont des dérivations équivalentes. A fortiori, tous ces appels paresseux aboutissent au même terme N. Le terme N est la forme normale de tête complète du terme M. C'est à dire que pour toute dérivation complète $\overline{B}$: $M \to^* P$ qui est telle que le terme P est en forme normale de tête, il existe une dérivation $\overline{C}$: $N \to^* P$. Le résultat principal du lemme ci-dessus peut donc être rapidement formulé en disant que la stratégie d'appel paresseux est correcte et optimale pour le calcul de la forme normale de tête dans les graphes acycliques.

Enfin, les stratégies d'appel paresseux vers un certain préfixe stable se définissent en considérant l'appel paresseux au lieu de l'appel par nom dans la description de la section 4.4.4. C'est à dire que l'appel paresseux vers un certain préfixe est une stratégie qui contracte des familles standards appropriées, jusqu'à atteindre le préfixe stable visé. Ces stratégies sont sûres et optimales pour le calcul du préfixe stable.

Proposition 5.5.5 *Soit un terme M cohérent et possédant une forme normale relativement à un préfixe γ. Alors, toute dérivation qui suit la stratégie de l'appel paresseux vers le préfixe γ calcule une forme normale de M relativement à γ, et ce pour un coût minimum.*

Démonstration : En reprenant exactement le schéma de la démonstration du lemme 4.4.23 de correction de l'appel par nom vers le préfixe γ, on montre, à partir du résultat précédent sur le calcul optimal de la tête (proposition 5.5.4), que tous les appels paresseux vers γ terminent et qu'ils sont des dérivations équivalentes incluses dans toute dérivation complète $\overline{A}$: $M \to^* N$ telle que le terme N est une forme normale relativement au préfixe γ. Quels que soient $\overline{B}$, un appel paresseux vers γ et A, une dérivation normalisante relativement à γ, on a donc les inégalités $\mathcal{C}(\overline{B}) \leq \mathcal{C}(\overline{A}) \leq \mathcal{C}(A)$, par les mêmes arguments que pour le calcul optimal de la tête. □

5.6 Systèmes gauches

La stratégie paresseuse se ramène une stratégie bien connue pour une classe significative de systèmes de réécriture. Les systèmes gauches ont été introduits pour la première fois dans [O'D 77]. En voici une définition aménagée pour la réécriture conditionnelle.

Définition 5.6.1 *Par définition, un système de réécriture Σ est un système gauche, si et seulement si, pour tout membre gauche de règle α et tout sous-terme autorisé α/u qui est une variable, tous les sous-termes autorisés qui sont à droite de α/u sont aussi des variables. En symboles et pour tout nom de radical α, on a :*

$$\forall u \in \mathcal{O}(\alpha),\ \alpha/u = x \Rightarrow (\forall v \in \mathcal{O}(\alpha),\ u \leq_{lo} v \Rightarrow \alpha/v = y)$$

Il est très facile de décider si un système est gauche ou non. Il suffit de faire la liste selon l'ordre gauche de tous les sous-termes autorisés de chaque nom de radical. Par exemple, le système $\lambda\sigma$ faible est un système gauche.

$$\begin{aligned}
\text{(WeakBeta)} &: \{(\lambda M)[s]\ N, (\lambda M)[s], s, N\} \\
\text{(App)} &: \{(M\ N)[s], s\} \\
\text{(FVar)} &: \{\mathtt{1}[M.s], M.s, M, s\} \\
\text{(RVar)} &: \{\mathtt{n+1}[M.s], M.s, M, s\}
\end{aligned}$$

De même, tous les systèmes à combinateurs et en particulier les règles de la logique combinatoire sont des systèmes gauches. Les schémas de programmes récursifs [Vui 74] sont un autre exemple.

Étant donné un terme M, le radical le plus à gauche dans le terme M est celui dont l'occurrence est minimale pour l'ordre total $\leq_{lo}$ parmi les occurrences des radicaux du terme M. Grâce à la forme spécifique des noms de radicaux dans les systèmes gauches, le radical le plus à gauche reste le radical le plus à gauche au cours des dérivations.

Lemme 5.6.2 *Soit une réduction A: $M \to N$ dans un système gauche. Soit R le radical le plus à gauche dans le terme M. Si le radical R n'est pas contracté par la réduction A, alors il admet un unique résidu qui est le radical le plus à gauche dans le terme M.*

Démonstration: On désigne par u l'occurrence de R dans le terme M. Le radical R qui est préservé par la dérivation A admet un unique résidu d'occurrence u dans le terme N. Il existe donc au moins un radical dans le terme N. Par conséquent il existe un radical S qui est le plus à gauche dans le terme N, soit v son occurrence. On montre ensuite l'égalité $u = v$ par induction sur la structure de la réduction A.

1. Si A est du type (id), alors il n'y a pas de radical dans $M = x^l$.
2. Si A est du type (rec), avec :

$$\frac{A_i\colon M_i \to N_i \text{ si } i \in \mathrm{dom}(f) \quad A_i\colon M_i = N_i \text{ si } i \notin \mathrm{dom}(f)}{A\colon f^l(M_1, M_2, \ldots, M_a) \to f^l(N_1, N_2, \ldots, N_a)}\ (\text{rec})$$

   Si $u = \epsilon$, alors on a $\epsilon \setminus A = \{\epsilon\}$ et le radical d'occurrence $v = \epsilon$ dans le terme N est à la fois l'unique résidu de R par la réduction A le radical le plus à gauche.

   Sinon, il existe un indice i dans le domaine de f, avec $u = i.u_i$. Montrons tout d'abord que l'on a pas $v = \epsilon$. Si tel était le cas, le radical S serait nécessairement un radical créé par la réduction A. Donc (cf. la preuve du lemme 2.5.3), il existerait un radical extérieur pour la réduction A et créateur de S. Soit w l'occurrence de ce radical dans le terme M. Rappelons que l'on aurait alors $w\setminus\!\!\setminus\!\!\setminus A = \{w\}$ et que l'occurrence w serait autorisée et dure dans le nom α du radical S. Par la condition de non-recouvrement des noms de radicaux appliquée aux radicaux d'occurrences ϵ et u dans le terme N, il existerait une occurrence autorisée u_x d'une variable x du nom α et une occurrence u', avec $u = u_x.u'$. Par définition des systèmes gauches, on aurait alors $w <_{lo} u$. Soit, il existerait dans le terme M un radical d'occurrence w strictement au dessus ou plus à gauche que le radical R, ce qui est exclu par hypothèse. L'occurrence v est donc de la forme $v = j.v_j$, où j est un indice du domaine de f. Par l'hypothèse $v \leq_{lo} u$, on a $j \leq i$. Si l'on avait l'inégalité stricte $j < i$, alors, par le lemme 2.4.3 appliqué à la réduction A_j: $M_j \to N_j$, il existerait au moins un radical dans le terme M_j. Dès lors, il existerait un radical à la gauche de R dans M. On a donc forcément $v = i.v_i$. Il est clair que u_i est l'occurrence du radical le plus à gauche dans le terme M_i, que v_i est l'occurrence du radical le plus à gauche dans dans le terme N_i et que le radical d'occurrence u_i n'est pas contracté par la réduction A_i. Soit, par induction, $u_i = v_i$ et finalement $u = v$.

3. Si la réduction A est de type (red), alors on a nécessairement $u = \epsilon$ et le radical R est contracté. □

Corollaire 5.6.3 *Soit Σ un système gauche. Soit un terme M qui n'est pas en forme normale pour Σ. Alors, le radical le plus à gauche dans le terme M est un radical standard à partir du terme M.*

Démonstration : Soit u l'occurrence du radical le plus à gauche. Pour toute dérivation A: $M \to^* N$ et par le lemme 5.6.2, le radical R est préservé par la dérivation A toute entière ou bien il est préservé jusqu'à être un des radicaux extérieurs pour une étape élémentaire de la dérivation A. Soit finalement $u \in \mathcal{E}xt'(A)$. D'où il vient $u \in \mathcal{E}(M)$ par le lemme 4.4.8. □ □

Pour tout système gauche, la stratégie de réduction qui consiste à toujours réduire le radical partagé dont une des instances est le radical le plus à gauche est donc une stratégie paresseuse. On appelle *stratégie extérieure-gauche dans les graphes* cette stratégie, qui est donc correcte et optimale pour le calcul de la tête puis de la forme normale. Ce résultat est valable dans deux modélisations différentes du λ-calcul faible, qui sont $\lambda\sigma$ faible et les systèmes à combinateurs.

Cet exemple permet aussi de comprendre l'intérêt de la forme normale relativement à un préfixe. Dans le cas, par exemple, d'un système à combinateurs, certains préfixes stables sont aisément reconnaissables. Il s'agit des préfixes de la forme $\gamma = F\ x_1\ x_2 \ldots\ x_n$, où l'entier n est strictement plus petit que l'arité du combinateur F. On appelle *formes normales de tête faibles* les formes normales relativement à de tels préfixes. Alors, la stratégie gauche complète est sûre et optimale pour la mise en forme normale de tête faible.

Proposition 5.6.4 *Dans un système à combinateurs, soit un terme M cohérent, tel qu'il existe une dérivation A: $M \to^* N$, avec N en forme normale de tête faible. Soit ensuite $\overline{B}$ la dérivation gauche complète issue de M. Alors la dérivation $\overline{B}$ atteint un terme en forme normale de tête faible en un nombre minimum d'étapes.*

Démonstration : On pose $N = F\ N_1\ N_2 \ldots\ N_n$. On montre facilement par induction sur l'entier n que la dérivation $\overline{B}$ suit la stratégie paresseuse vers le préfixe $\gamma = F\ x_1\ x_2 \ldots\ x_n$ (cf la définition de la section 4.4.4). Le point clé est que, pour tout terme de la forme $P = @(P_1, P_2)$ qui est en forme normale de tête et donc n'admet pas, *a fortiori*, de radical à la racine, les radicaux les plus à gauche dans les terme P et P_1 sont les mêmes. On applique ensuite la proposition 5.5.5. Un point remarquable est qu'il n'est en fait pas utile d'identifier la première étape de $\overline{B}$ qui aboutit à une forme normale de tête, il suffit de calculer normalement jusqu'à atteindre une forme normale de tête faible. □

L'optimalité de la stratégie extérieure-gauche avec partage dans les graphes acycliques est un résultat connu, dans le cas des schémas de programme récursifs [Ber 79], ainsi que dans celui de la logique combinatoire [Ken 84, Sta 80]. Le cadre de cette thèse est plus large, non seulement, parce qu'il s'applique à toute la classe des systèmes orthogonaux, mais aussi par ma notion d'optimalité (telle que donnée dans le théorème 5.5.2). En effet, les définitions de l'optimalité selon les auteurs cités ne s'appliquent simplement qu'aux dérivations normalisantes et nécessitent un traitement sémantique particulier pour être étendues aux dérivations les plus générales. Le théorème d'optimalité, quoique simple, n'en est pas moins tout à fait général, puisqu'il a pour corollaires les divers résultats d'optimalité des stratégies complètes. Je pense qu'il traduit bien l'essence de l'optimalité, comme le lemme des déplacements parallèles traduit l'essence du déterminisme.

Chapitre 6

Appel par filtrage ambigu

Comme le λ-calcul peut être compilé vers les différentes formes de λ-calcul faible, le langage ML doit pouvoir être compilé, de sorte que tout programme écrit en ML soit transformé en un système de réécriture. Dans la pratique, la compilation doit bien évidemment traiter des programmes ML complets, cependant la théorie n'a jusqu'ici été développée qu'en considérant le noyau de ML qui est le λ-calcul augmenté de la récursion et éventuellement —voir [Tur 79] pour la compilation de ML vers les combinateurs de la logique combinatoire et [Pey 87] pour la compilation vers les systèmes de supercombinateurs—.

Dans ce chapitre, j'utilise la plus moderne de ces techniques, les supercombinateurs, pour montrer qu'il est possible de traiter en plus l'appel par filtrage. On parvient ainsi à la compilation de la totalité de ML vers les systèmes de réécriture.

6.1 Systèmes à combinateurs étendus par filtrage

Les systèmes de réécriture considérés dans ce chapitre sont les produits de la compilation de programmes GAML [Mar 91]. GAML est une version paresseuse de ML et est donc statiquement typé. Les *constructeurs* sont des symboles fonctionnels d'arité quelconque introduits par des déclarations de types. Par exemple, voici la déclaration du type des booléens :

$$\texttt{type bool = true | false}$$

Cette déclaration définit les constructeurs zéroaires ou *constants* `true` et `false`. L'ensemble formé de ces deux constructeurs est la *signature* du type des booléens. Outre les booléens, le compilateur GAML reprend tous les types prédéfinis de CAML (voir l'introduction). En fait, les déclarations de type font plus que définir de nouveaux constructeurs et spécifier leur arité. Elles donnent aussi les règles de typage qui s'appliquent à chaque constructeur. Soit, par exemple, la déclaration du type des arbres polymorphes :

$$\texttt{type arbre } \alpha \texttt{ = Feuille } \alpha \texttt{ | Noeud (arbre } \alpha\texttt{) } \alpha \texttt{ (arbre } \alpha\texttt{)}$$

Cette déclaration définit les constructeurs `Feuille` et `Noeud` d'arités respectives 1 et 3. Chacun de ces constructeurs possède un type le plus général, qui est rendu par un *schéma de type polymorphe* :

$$\texttt{Feuille: } \forall\alpha.\alpha \rightarrow \texttt{arbre } \alpha \qquad \texttt{Noeud: } \forall\alpha.\texttt{arbre } \alpha \rightarrow \alpha \rightarrow \texttt{arbre } \alpha \rightarrow \texttt{arbre } \alpha$$

C'est à dire que le terme $\texttt{Noeud}(M_1, M_2, M_3)$ est du type `arbre` α, pourvu que les terme M_1 et M_3 soient aussi du type `arbre` α et que le terme M_2 soit du type α.

Si un terme n'est pas clos, alors il n'est typable que dans le cadre d'un *environnement de types* qui assigne un type à chacune de ses variables. Par exemple, on peut assigner le type `arbre bool` au terme $M = \texttt{Noeud}(x, y, \texttt{Feuille}(\texttt{false}))$, à condition de se placer dans l'environnement de types $\Gamma = [x{:}\,\texttt{arbre bool}, y{:}\,\texttt{bool}]$. Pour plus de précisions, je renvoie à l'article fondateur sur le typage polymorphe de ML [Dam 82] ou au chapitre introductif de la thèse [Ler 92].

Les *motifs* sont des termes particuliers qui vont servir à définir les règles de réécriture. Ce sont des termes construits bien typés et linéaires.

$$\textit{Motifs} \quad p \quad ::= \quad x \mid c(p_1, p_2, \ldots, p_a) \quad \textit{le terme } p \textit{ est linéaire et typable}$$

Les *types fonctionnels* sont les types de la forme $\tau = \tau_1 \to \tau_2$. Les autres types sont les *types de base*. Le type d'un motif qui n'est pas une variable est toujours un type de base. Enfin, un type qui ne contient aucune flèche est un *type de résultat*.

On se donne ensuite un ensemble de symboles de combinateurs, qui sont des symboles fonctionnels d'arité nulle. Chaque symbole de combinateur possède un type éventuellement polymorphe représenté par un schéma de type. Par exemple, le type polymorphe du combinateur K de la logique combinatoire est $\forall \alpha\beta.\alpha \to \beta \to \alpha$. Les *termes applicatifs* sont les termes bien typés construits à partir de variables (notées x), de symboles de combinateurs (notés F), de symboles de constructeurs (notés c d'arité a) et de l'application binaire @ :

$$\textit{Termes applicatifs} \quad e \quad ::= \quad x \mid F \mid c(e_1, e_2, \ldots, e_a) \mid @(e_1, e_2) \quad \textit{Le terme } e \textit{ est typable}$$

À chaque symbole de combinateur F, on associe un nombre fini de règles. La forme imposée de ces règles est décrite comme suit :

$$\begin{array}{cccccc} F & p_1^1 & p_2^1 & \cdots & p_n^1 & \to & e^1 \\ F & p_1^2 & p_2^2 & \cdots & p_n^2 & \to & e^2 \\ & & & & & \vdots & \\ F & p_1^m & p_2^m & \cdots & p_n^m & \to & e^m \end{array}$$

Pour tout entier i compris entre 1 et m, la règle $F\ p_1^i\ p_2^i \ldots\ p_n^i \to e^i$ est la *clause* numéro i de la définition de F. L'entier n est l'arité du combinateur F, toutes les clauses qui définissent le combinateur F comprennent n motifs disposés d'une façon identique comme les arguments de l'application du symbole de combinateur F à n arguments. C'est à dire que, la i-ème clause qui définit F est exactement :

$$@(\ldots @(@(F, p_1^i), p_2^i) \ldots, p_n^i) \to e^i$$

On retrouve le cas plus simple des combinateurs sans motifs de la fin de la section 1.1.2 en prenant des motifs toujours réduits à des variables.

Une représentation compacte d'une définition par filtrage ou définition de combinateur est donc un triplet composé d'un symbole de combinateur F, d'une *matrice de motifs* (P) et d'un *vecteur colonne* de termes applicatifs (E) :

$$F, \begin{pmatrix} p_1^1 & p_2^1 & \cdots & p_n^1 \\ p_1^2 & p_2^2 & \cdots & p_n^2 \\ & \vdots & & \\ p_1^m & p_2^m & \cdots & p_n^m \end{pmatrix}, \begin{pmatrix} e^1 \\ e^2 \\ \vdots \\ e^m \end{pmatrix}$$

De façon abrégée, on note $D = F, (P), (E)$ les définitions de combinateurs.Chacune des lignes de la matrice (P) est composée d'un vecteur de motifs de taille n, que l'on note $\vec{p}^{\,i}$ ou encore

$(p_1^i\ p_2^i\ \ldots\ p_n^i)$. Enfin, dans l'écriture du motif composante p_j^i, l'entier i est le *numéro de ligne* et l'entier j est l'*indice de colonne*.

Quand on veut faire abstraction du symbole de combinateur F, on parle de clauses, notées $(\vec{p}, e)$, ainsi que d'*ensembles de clauses*, notés $(P), (E)$.

Un ensemble fini de définitions de combinateurs constitue un *programme* ou *système d'appel par filtrage*, généralement noté $\mathcal{P}$. On ne considère que les programmes qui obéissent aux deux contraintes suivantes :

1. Les règles introduites sont linéaires à gauche.
2. Les règles introduites sont bien typées.

Les systèmes d'appel par filtrage peuvent être simplement étiquetés par le procédé introduit dans la section 1.2.3, on obtient ainsi des *programmes simplement étiquetés* que l'on note généralement $\mathcal{P}^g$. Pourvu que l'on s'impose les conditions de la réécriture de graphes (définition 1.3.5), les programmes simplement étiquetés s'assimilent aux systèmes d'appel par filtrage portant sur les graphes acycliques.

Les techniques du λ-lifting et de la compilation vers une machine virtuelle à réduction de graphes, ou G-machine, permettent de transformer tout programme ML en un système d'appel par filtrage portant sur les graphes acycliques [Pey 87, Aug 87, Joh 87]. Je renvoie à [Aug 87] pour une preuve de la correction de ces transformation, qui n'est malheureusement effectuée que dans le cadre d'une sémantique dénotationnelle de ML (paresseux). Un résultat plus général relativement à une sémantique de ML par réécriture [Plo 77, How 90, Wri 91], fondée sur la β réduction classique dans le λ-calcul peut sans doute être montré.

L'existence de ces transformations encourage la représentation des règles de réécriture des systèmes d'appel par filtrage comme une suite de définitions GAML.

Motifs :

p	$::=$	x	variable
	$\mid$	`_`	attrape-tout
	$\mid$	$c\ p_1\ p_2 \ldots\ p_a$	motif construit

Expressions :

e	$::=$	x	variable
	$\mid$	$e_1\ e_2$	application (sous-entendue)
	$\mid$	$c\ e_1\ e_2 \ldots\ e_a$	terme construit
	$\mid$	`let` x_1 `=` e_1 `and` $\ldots x_n$ `=` e_n `in` e	définition locale

Clauses :

$cl\ ::=\ p_1\ p_2 \ldots\ p_a \to e$

Définitions de combinateurs :

$d\ ::=$ `let rec` F `= function` $cl^1 \mid cl^2 \mid \ldots \mid cl^m$

La principale restriction, par rapport à GAML complet, est l'inexistence de définitions de fonctions locales. En effet, il n'y pas de moyen d'introduire les définitions de combinateurs (classe d) à l'intérieur des termes applicatifs (classe e).

Les expressions de la classe e ci-dessus sont équivalentes à l'ensembles des graphes acycliques dans les termes applicatifs simplement étiqueté, le partage pouvant être rendu par des liaisons `let` locales ou des étiquettes. Dans la mise en œuvre de ML par la G-machine [Joh 87, Aug 87, Mar 91] ces expressions sont traduites en un code spécialisé (le G-code), dont l'exécution par une machine virtuelle adaptée, produit le graphe équivalent à l'expression e. Dans la formalisation du langage fonctionnel Id [Ari 90], les graphes sont représentés directement sous une forme proche de celles des expressions e.

Un exemple d'utilisation de la syntaxe GAML des combinateurs est la définition suivante du combinateur somme qui fait la "somme" de deux arbres d'entiers :

```
let rec somme = function
  (Noeud t1 i t2) (Noeud u1 j u2)  →  Noeud (somme t1 u1) (i+j) (somme t2 u2)
| (Noeud t1 i t2) (Feuille j)      →  Noeud t1 (i+j) t2
| (Feuille i) (Noeud u1 j u2)      →  Noeud u1 (i+j) u2
| (Feuille i) (Feuille j)          →  Feuille (i+j)
```

On suppose ici qu'il existe un combinateur plus de type int → int → int défini par une infinité de clauses de la forme plus $i\ j \to k$, où k est la somme des entiers i et j. (L'utilisateur qui écrit un programme fini ne peut donner qu'un nombre fini de clauses, mais le système GAML a d'autres moyens pour définir des primitives). Dès lors, on peut définir le combinateur somme par le triplet :

$$\mathtt{somme}, \begin{pmatrix} \mathtt{Noeud}(t_1,i,t_2) & \mathtt{Noeud}(u_1,j,u_2) \\ \mathtt{Noeud}(t_1,i,t_2) & \mathtt{Feuille}(j) \\ \mathtt{Feuille}(i) & \mathtt{Noeud}(u_1,j,u_2) \\ \mathtt{Feuille}(i) & \mathtt{Feuille}(j) \end{pmatrix}, \begin{pmatrix} \mathtt{Noeud}(\mathtt{somme}\ t_1\ u_1, \mathtt{plus}\ i\ j, \mathtt{somme}\ t_2\ u_2) \\ \mathtt{Noeud}(t_1, \mathtt{plus}\ i\ j, t_2) \\ \mathtt{Noeud}(u_1, \mathtt{plus}\ i\ j, u_2) \\ \mathtt{Feuille}(\mathtt{plus}\ i\ j) \end{pmatrix}$$

Enfin, le combinateur somme est bien du type arbre int → arbre int → arbre int.

Il n'y a fondamentalement aucune différence entre une définition de combinateur sous forme d'un programme GAML ou d'une série de clauses. C'est pourquoi on adopte désormais la première de ces écritures dans les exemples. En particulier, le terme construit $c(M_1, \ldots, M_a)$ est noté c $\mathtt{M_1}$ ... $\mathtt{M_a}$ c'est à dire que les parenthèses et les virgules sont implicites. Il s'agit là d'une commodité inspirée par la syntaxe de GAML, elle n'implique pas l'utilisation implicite de l'application binaire @. De fait, la règle de l'arité doit être respectée et il n'y a pas "d'application partielle" des constructeurs. On utilise aussi la notation "_" du motif attrape-tout, pour remplacer les variables qui ne sont pas reprises dans les membres droits des clauses. Par exemple, le programme suivant ignore les poids placés aux nœuds intermédiaires d'un arbre d'entiers :

```
let rec somme_feuilles = function
  (Noeud t1 _ t2)  →  somme_feuilles t1+somme_feuilles t2
| (Feuille i)      →  i
```

Jusqu'ici, aucune condition particulière n'a encore été imposée aux définitions par filtrage dans le but d'éviter les paires critiques. En fait, comme nous allons le voir, cette condition est très contraignante et il est hors de question de l'imposer dans un langage de programmation réaliste.

6.2 Termes partiellement connus

6.2.1 Termes partiels

En raison de la linéarité à gauche des règles, les noms des variables d'un membre gauche n'ont d'importance que relativement au membre droit correspondant. Il est commode de pouvoir les oublier. En particulier, quand on cherche à savoir si un terme M est un radical de nom α, seules importent les occurrences dures de α. Dans ce contexte,les sous-termes de M qui sont situés à d'autres occurrences peuvent être ignorés, voire considérés comme inconnus. Dans ce but, on se donne on nouveau symbole nullaire Ω que l'on appelle le *terme indéfini.* Nous verrons dans la section suivante une autre utilisation du terme indéfini Ω pour représenter les résultats des réductions.

L'ensemble $\mathcal{T}_\Omega$ des termes partiels ou Ω-termes est l'ensemble des termes clos augmenté des termes qui contiennent le terme indéfini Ω.

$$\textit{Termes partiels} \quad \mathcal{T}_\Omega\text{:} \quad M \quad ::= \quad \Omega \mid f(M_1, M_2, \ldots, M_a)$$

À tout terme M, on associe le terme partiel M_Ω qui est égal à M, après remplacement des variables de M par le terme indéfini Ω. Un Ω-terme est donc parfois noté M_Ω.

Les Ω-termes sont naturellement ordonnés par l'ordre préfixe.

Définition 6.2.1 (Ordre préfixe) *Soient deux termes partiels M et N. La relation d'ordre préfixe, notée $M \preceq N$ est définie par l'équivalence :*

$$M \preceq N \Leftrightarrow \begin{cases} M = \Omega \\ \quad ou \\ M = f(M_1, M_2, \ldots, M_a), N = f(N_1, N_2, \ldots, N_a) \ \textit{et}\ \forall i \in [1 \ldots a]\ M_i \preceq N_i \end{cases}$$

Par convention, étant donnés deux termes M et N, on note aussi $M \preceq N$ la relation $M_\Omega \preceq N_\Omega$. La relation stricte associée à $\preceq$ est notée $\prec$. C'est à dire que l'on pose $M \prec N \Leftrightarrow M \preceq N$ et $N \not\preceq M$. La relation $\preceq$ est un ordre bien fondé sur l'ensemble $\mathcal{T}_\Omega$. En fait, on a ici une propriété plus forte : on montre aisément par induction sur la structure des termes partiels que, pour tout Ω-terme M, il y a un nombre fini de termes partiels qui sont des préfixes de M.

Comme annoncé, Ω-termes et ordre préfixe servent à reconnaître les radicaux.

Définition 6.2.2 (Relation d'instanciation) *Soient un terme linéaire α et un terme M. Le terme M est une instance du terme α, si et seulement si il existe une substitution σ, avec $\sigma(\alpha) = M$. On note alors $\alpha \preceq_\sigma M$ et $\sigma = [\alpha \backslash M]$*

Lemme 6.2.3 *Soient un terme linéaire α et un terme M. Alors, on a l'équivalence :*

$$\alpha \preceq_\sigma \mathcal{M} \Leftrightarrow \alpha \preceq M$$

Démonstration : La preuve est une induction facile sur le terme α dont elle utilise la linéarité. □

Deux noms de radicaux α et γ peuvent donner lieu à une paire critique quand ils ont des instances communes, on dit alors qu'ils sont *compatibles* et on note $\alpha \uparrow_\sigma \gamma$. De même, deux termes partiels M et N sont compatibles et on note $M \uparrow N$, quand il existe un terme partiel P qui est tel que $M \preceq P$ et $N \preceq P$. Deux Ω-termes M et N qui n'ont pas de majorant commun sont *incompatibles* et on note $M \# N$. Par le lemme 6.2.3 et pour tout couple de noms de radicaux (α, γ), on a l'équivalence $\alpha \uparrow_\sigma \gamma \Leftrightarrow \alpha \uparrow \gamma$.

La compatibilité est aisément décidable.

Lemme 6.2.4 *Pour tous termes partiels M et N, on a l'équivalence :*

$$M \uparrow N \Leftrightarrow \begin{cases} M = \Omega \ \textit{ou}\ N = \Omega \\ \quad ou \\ M = f(M_1, M_2, \ldots, M_a), N = f(N_1, N_2, \ldots, N_a) \ \textit{et}\ \forall i \in [1 \ldots a],\ M_i \uparrow N_i \end{cases}$$

Démonstration : La preuve est une induction facile sur la structure des Ω-termes M et N. □

Le résultat correspondant pour l'incompatibilité vient immédiatement par négation :

$$M \# N \Leftrightarrow \begin{cases} M = f(M_1, M_2, \ldots, M_a) \text{ et } N = f'(N_1, N_2, \ldots, N_{a'}), \text{ avec } f \neq f' \\ \text{ou} \\ M = f(M_1, M_2, \ldots, M_a), N = f(N_1, N_2, \ldots, N_a) \text{ et } \exists i \in [1 \ldots a],\ M_i \# N_i \end{cases}$$

Les propriétés suivantes se démontrent à l'aide de la caractérisation inductive de la compatibilité :

Proposition 6.2.5 *Soient M et N deux Ω-termes compatibles.*

1. *Les termes partiels M et N admettent un plus petit commun majorant ou borne supérieure noté $M \sqcup N$ avec :*

$$\begin{cases} \Omega \sqcup N &= N \\ M \sqcup \Omega &= M \\ f(M_1, M_2, \ldots, M_a) \sqcup f(N_1, N_2, \ldots, N_a) &= f(M_1 \sqcup N_1, M_2 \sqcup N_2, \ldots, M_a \sqcup N_a) \end{cases}$$

2. *On a l'équivalence :*

$$M \preceq N \Leftrightarrow M \sqcup N = N$$

Deux noms de radicaux compatibles α et γ sont aussi dits *unifiables* [Rob 65]. Par convention, on pose que les variables des noms α et γ sont deux à deux distinctes. Leur instance commune minimale est un terme $\delta = \sigma(\alpha) = \sigma(\gamma)$ qui est tel que $\delta_\Omega = \alpha_\Omega \sqcup \gamma_\Omega$. La substitution σ est *l'unificateur principal* des termes α et γ. Et, pour toute instance commune $\phi(\alpha) = \phi(\gamma)$, il existe une substitution θ définie sur les variables de δ, avec $\theta(\delta) = \phi(\alpha) = \phi(\gamma)$, soit $\phi = \theta \circ \sigma$. Ici, on peut toujours choisir les noms des variables de δ de sorte que le terme δ soit linéaire et que toutes ses variables soient nouvelles. Dès lors, pour toute variable y d'occurrence u_y dans δ, l'occurrence u_y est une occurrence molle d'au moins un des termes α ou γ. En outre, si l'occurrence u_y n'est pas une occurrence de variable dans le terme α, alors il existe une variable x d'occurrence u_x dans le terme α, avec $u_y = u_x.v_y$, où v_y est l'occurrence de la variable y dans le terme dur $\sigma(x)$, tandis que les termes $\sigma(x)$ et γ/u_x sont égaux aux variables près ($\sigma(x) \preceq \gamma/u_x$ et $\gamma/u_x \preceq \sigma(x)$, que l'on peut aussi écrire $(\gamma/u_x)_\Omega = (\sigma(x))_\Omega$).

6.2.2 Valeurs partielles

Dans le cas des systèmes d'appel par filtrage, un sous-ensemble de termes partiels revêt un intérêt particulier :

$$\textit{Valeurs partielles} \quad \mathcal{V}_\Omega\!: \quad V \quad ::= \quad \Omega \mid c(V_1, V_2, \ldots, V_a)$$

Une *constante* est une valeur partielle qui ne contient pas de sous-terme indéfini. Les valeurs partielles sont donc des termes partiels fabriqués à l'aide des seuls constructeurs. Ils traduisent le point de vue de l'observateur des résultats de l'exécution d'un programme GAML.

1. Seuls comptent les constructeurs.
2. Tous les termes dont le type contient au moins une flèche sont "des fonctions" qui ne sont pas distinguables à ce niveau.

La fonction d'*approximation* ω traduit tout terme (partiel) en une valeur partielle et permet de donner une valeur à tout terme applicatif :

$$\omega(\Omega) = \Omega \qquad \omega(x) = \Omega \qquad \omega(c(M_1, \ldots, M_a)) = c(\omega(M_1), \ldots, \omega(M_a))$$

$$\omega(F) = \Omega \qquad \omega(@(M_1, M_2)) = \Omega$$

La fonction d'approximation est monotone ($M \prec N \Rightarrow \omega(M) \prec \omega(N)$) et introduit une nouvelle relation d'ordre entre les termes (partiels).

Définition 6.2.6 *La relation d'ordre de définition sur l'ensemble des termes (partiels) est définie par l'équivalence suivante :*

$$M \preceq_\omega N \Leftrightarrow \omega(M) \preceq \omega(N)$$

La monotonie de l'approximation ω entraine immédiatement l'implication $M \preceq N \Rightarrow M \preceq_\omega N$. Il faut remarquer que le terme indéfini Ω a la double utilité de représenter les sous-termes inconnus (i.e., non-évalués) et indifférents (i.e., de type fonctionnel). Cette confusion se justifie de par la forme particulière des noms de radicaux dans les systèmes d'appel par filtrage.

Lemme 6.2.7 *Un terme (partiel) M est une instance d'un motif p, si et seulement si la valeur partielle $\omega(p)$ est un préfixe de la valeur partielle $\omega(M)$. Autrement dit, dans le cas d'un motif p et d'un terme M, on a les équivalences :*

$$p \preceq_\sigma M \Leftrightarrow p \preceq M \Leftrightarrow p \preceq_\omega M$$

Démonstration : Le première équivalence est le lemme 6.2.3 et la seconde est immédiate en considérant que les motifs ne contiennent, ni le symbole @, ni aucun symbole de combinateur. □

Un terme $M = F\ M_1\ M_2 \ldots\ M_n$ est donc un appel réductible du combinateur F, quand n est l'arité de F et qu'il existe une ligne de motifs $\vec{p}$ telle que, pour tout entier i compris entre 1 et n, on a la condition $p_i \preceq_\omega M_i$. C'est à dire que l'on peut adopter le parti d'ignorer les valeurs fonctionnelles, quand il s'agit de savoir si les arguments d'un appel de combinateur sont suffisamment réduits pour autoriser un appel par filtrage.

Le lemme suivant, en montrant que l'approximation d'un terme en cours de réduction ne peut que s'améliorer, justifie de considérer les valeurs partielles comme des approximations successives du résultat d'un calcul.

Lemme 6.2.8 *Soit un programme qui définit un système orthogonal, alors pour toute dérivation A: $M \rightarrow^* N$ dans ce système, on a $M \preceq_\omega N$.*

Démonstration: Il suffit de montrer l'inégalité $\omega(M) \preceq \omega(N)$ pour toute réduction A: $M \rightarrow N$.

1. Si A est du type (id), alors $\omega(M) = \omega(N) = \Omega$.
2. Si A est du type (rec), avec $M = f(M_1, \ldots, M_a)$ et $N = f(N_1, \ldots, N_a)$, alors il y a deux cas. Si le symbole f est égal à @ ou à un combinateur d'arité nulle, on a directement $\omega(M) = \omega(N) = \Omega$. Sinon f est un constructeur c et le résultat vient par induction.
3. Si A est du type (red), alors, compte tenu de la forme des règles, on a tout de suite $\omega(M) = \Omega$, puisque le symbole de tête du terme M est une application binaire ou un combinateur d'arité nulle. □

Pour des raisons techniques, on étend l'approximation aux vecteurs de termes et aux matrices de motifs, en posant :

$$\omega(\vec{M}) = (\omega(M_1)\ \omega(M_2)\ \dots\ \omega(M_n)) \qquad \omega(P) = \begin{pmatrix} \omega(p_1^1) & \omega(p_2^1) & \dots & \omega(p_n^1) \\ \omega(p_1^2) & \omega(p_2^2) & \dots & \omega(p_n^2) \\ & \vdots & & \\ \omega(p_1^m) & \omega(p_2^m) & \dots & \omega(p_n^m) \end{pmatrix}$$

Les relations $\preceq$, $\preceq_\sigma$ et $\preceq_\omega$ s'étendent naturellement aux vecteurs de motifs et de termes, on a par exemple :

$$(p_1\ p_2\ \dots\ p_n) \preceq (M_1\ M_2\ \dots\ M_n) \Leftrightarrow \forall i \in [1 \dots n],\ p_i \preceq M_i$$

De même, compatibilité et incompatibilité ont un sens pour les vecteurs de motifs, comme ils en ont un pour les termes linéaires :

$$\begin{array}{rcl} (p_1\ p_2\ \dots\ p_n) \# (q_1\ q_2\ \dots\ q_n) & \Leftrightarrow & \exists i \in\ [1 \dots n],\ p_i \# q_i \\ (p_1\ p_2\ \dots\ p_n) \uparrow (q_1\ q_2\ \dots\ q_n) & \Leftrightarrow & \forall i \in\ [1 \dots n],\ p_i \uparrow q_i \end{array}$$

Lemme 6.2.9 *Un programme définit un système orthogonal, si et seulement si, pour toute matrice de motifs (P) de ce programme, tous les vecteurs de motifs qui composent (P) sont deux à deux incompatibles.*

Démonstration : Soient deux membres gauches $\alpha = F\ p_1\ p_2 \dots\ p_a$ et $\alpha' = F'\ p'_1\ p'_2 \dots\ p'_{a'}$, qui forment une paire critique. Par définition, il existe deux substitutions σ et σ', ainsi qu'une occurrence u dure dans le nom α, avec $\sigma(\alpha/u) = \sigma'(\alpha')$. Par conséquent, le symbole de tête du sous-terme dur α/u est soit @, soit un combinateur d'arité nulle. Dès lors, l'occurrence u est de la forme $1.1 \dots 1$ et on a $F = F'$ puis $a = a'$. Soit finalement, les deux termes $F\ p_1\ p_2 \dots\ p_a$ et $F\ p'_1\ p'_2 \dots\ p'_a$ ont une instance commune. C'est à dire que les vecteurs de motifs $\vec{p}$ et $\vec{p}'$ sont compatibles. □

6.3 Le filtrage en GAML

6.3.1 Filtrage et réécriture avec priorité

On se rend facilement compte qu'il est trop contraignant d'imposer l'incompatibilité des motifs au programmeur GAML. On interdirait alors des programmes parmi les plus traditionnels. En effet, il est souvent commode d'éviter d'énumérer tous les cas de filtrage. Par exemple, on est légitimement tenté d'écrire les primitives de la logique booléenne en n'utilisent que le motif `true` (figure 6.1). L'interprétation de telles définitions n'est pas immédiate. Informellement, le motif attrape-tout `_`

```
let and = function
  true true →  true
| _    _    →  false
```

```
let or = function
  _ true  →  true
| true _  →  true
| _    _  →  false
```

Figure 6.1 : Les primitives booléennes.

(ou une variable dans un motif) représente les cas par défaut des définition de fonctions. Par exemple, la dernière clause de la fonction `or` s'applique aux vecteurs de valeurs dont une au moins des composantes n'est pas le booléen `true`. Parfois, les cas par défaut ne sont pas contraints et représentent le cas général. Par exemple, dans la première clause de la fonction `or`, le motif `_` n'est pas contraint et représente n'importe quel booléen. Il est même des cas où l'utilisation des motifs attrape-tout est obligatoire, quand on considère des signatures infinies comme celle du type des entiers `int`.

```
let rec fib = function 0 → 1 | 1 → 1 | n → fib (n-1)+fib (n-2)
```

Dans la discussion qui va suivre, on se donne une fonction `F` plus simple que la fonction de Fibonacci dont elle reprend les membres gauches.

```
let  F = function 0 → "une" | 1 → "deux" | _ → "ni une, ni deux"
```

La déclaration de la fonction `F` ne peut pas être considérée directement comme définissant les règles d'un système de réécriture, car on perdrait la propriété de Church-Rosser :

$$A\colon \texttt{F 0} \xrightarrow{\epsilon} \texttt{"une"} \qquad B\colon \texttt{F 0} \xrightarrow{\epsilon} \texttt{"ni une, ni deux"}$$

Pour la même raison, on ne peut pas non plus utiliser directement la définition des systèmes avec priorité de [Bae 87]. Dans cette approche, on ordonne les règles de réécriture selon un ordre de *priorité*. Un choix naturel est l'ordre dans lequel le programmeur écrit ces règles. Par exemple, dans les cas de la fonction `F`, l'ordre retenu est :

$$\texttt{F 0} \rightarrow \texttt{"une"} \ll \texttt{F 1} \rightarrow \texttt{"deux"} \ll \texttt{F n} \rightarrow \texttt{"ni une, ni deux"}$$

On pose ensuite qu'une règle de réécriture ne peut s'appliquer effectivement que quand il n'existe pas de règle plus prioritaire qui s'applique aussi. Par exemple, la réduction selon la troisième règle $A\colon \texttt{F (0+0)} \xrightarrow{\epsilon} \texttt{"ni une, ni deux"}$ est valide, puisque le terme `F (0+0)` n'est pas un radical pour les deux règles les plus prioritaires. Il suffit ensuite de considérer la dérivation $B\colon \texttt{F (0+0)} \xrightarrow{2} \texttt{F 0} \xrightarrow{\epsilon} \texttt{"une"}$, pour constater que la propriété de Church-Rosser n'est pas satisfaite.

Pourtant, il est en général admis que la définition de la fonction `fib` a un sens qui est que, pour tout *entier* n, différent de 0 et de 1, l'appel de fonction `fib n` se calcule par la dernière clause de la définition de la fonction `fib`. C'est à dire que l'on utilise bien l'idée des priorités en exigeant en plus que les arguments d'une fonction soient suffisamment réduits, afin que toute décision d'appel soit irréversible. En effet, le typage des expression garantit ici que, si l'argument M de l'appel de fonction `fib M` est en forme normale (de tête), alors le terme `M` est un entier. Dans sa thèse [Lav 88], A. Laville étend cette intuition aux motifs arbitrairement profonds.

Définition 6.3.1 *Soit un motif p un terme M. Par définition, le motif p et le terme M sont incompatibles et on note $p \#_\omega M$, si et et seulement si les valeurs partielles $\omega(p)$ et $\omega(M)$ sont incompatibles ($\omega(p) \# \omega(M)$).*

C'est à dire que le motif p et le terme M sont incompatibles, quand le terme M est suffisamment défini pour que l'on soit certain qu'il n'est pas et ne deviendra jamais (lemme 6.2.8) une instance du motif p. En étendant la définition ci-dessus aux vecteurs, il vient naturellement :

$$(p_1\ p_2\ \ldots\ p_n) \#_\omega (M_1\ M_2\ \ldots\ M_n) \Leftrightarrow \exists i \in [1 \ldots n],\ p_i \#_\omega M_i$$

Définition 6.3.2 (Filtrage avec priorité textuelle) *Soit une matrice (P) composée de m vecteurs de motifs de taille n. Soit i un numéro de clause de la matrice (P). Le prédicat de filtrage selon la i-ème ligne de motifs de la matrice (P), noté $match_i[(P)]$, est une fonction définie de l'ensemble des vecteurs de termes de taille n vers l'ensemble des valeurs de vérité par l'équivalence :*

$$match_i[(P)](\vec{M}) \quad \Leftrightarrow \quad (\vec{p}^{\,i} \preceq_\omega \vec{M} \text{ et } \forall j < i,\ \vec{p}^{\,j} \#_\omega \vec{M})$$

Si le vecteur $\vec{M}$ est en fait un vecteur de valeurs strictes, c'est à dire que tous les symboles de tous les termes M_i sont des constructeurs, alors on a $\omega(\vec{M}) = \vec{M}$, l'incompatibilité $\vec{p}^{\,j} \#_\omega \vec{M}$ se ramène à la négation $\vec{p}^{\,j} \not\preceq \vec{M}$. Autrement dit, on retrouve le filtrage strict de langages tels que CAML, on peut alors se contenter de la présentation simple de la réécriture avec priorité.

Dans le cas général, la condition $\vec{p}^{\,i} \preceq_\omega \vec{M}$ entraîne que le vecteur de termes $\vec{M}$ est une instance du vecteur de motifs $\vec{p}^{\,i}$. Soit, il existe une substitution $\sigma = [\vec{p}^{\,i} \backslash \vec{M}]$, avec $\sigma(\vec{p}^{\,i}) = \vec{M}$. Lorsque la valeur de vérité $\text{match}_i[(P)](\vec{M})$ est égale à vrai, on dit que la i-ème ligne de la matrice (P) filtre le vecteur $\vec{M}$. Quand on ne souhaite pas préciser le numéro de ligne i, on dit simplement que la matrice (P) filtre le vecteur $\vec{M}$ et on note $\text{match}[(P)]$ le *prédicat de filtrage* correspondant qui est donc la disjonction de tous les prédicats $\text{match}_i[(P)]$.

On définit ensuite la relation de réduction simple avec priorité

Définition 6.3.3 *Soit un terme M. Par définition le sous-terme $N = M/u$ est un radical pour le programme $\mathcal{P}$, si et seulement si le sous-terme N est de la forme $N = F\ N_1\ N_2 \ldots\ N_n$ et qu'il existe une définition $F, (P), (E)$ dans le programme $\mathcal{P}$ et un numéro de clause i de la matrice (P), tels que le prédicat $match_i[(P)](N)$ est vérifié. Le terme M se réécrit alors selon la règle de réécriture avec priorité $\to'$ comme suit :*

$$M \quad \xrightarrow[\mathcal{P}]{u}{}' \ M[u \leftarrow e^i[\vec{p}^{\,i} \backslash \vec{N}]]$$

Remarquons que la définition de la réécriture avec priorité se ramène à celle de la réécriture classique dans le cas où les définitions de combinateurs ne sont pas ambiguës. En effet, si deux lignes de motifs de numéros i et j sont incompatibles ($\vec{p}^{\,i} \# \vec{p}^{\,j} \Leftrightarrow \vec{p}^{\,i} \#_\omega \vec{p}^{\,j}$), alors, pour tout vecteur de termes tel que l'on a $\vec{p}^{\,i} \preceq_\omega \vec{M}$, on a automatiquement $\vec{p}^{\,j} \#_\omega \vec{M}$, par définition de l'incompatibilité.

La définition de la réécriture avec priorité est conforme à une certaine intuition, mais il faut tout de même se méfier. Par exemple, la fonction `or` de la figure 6.1, n'est pas le "parallel-or" `por`. Pour le voir, on se donne la définition ML supplémentaire `let rec ⊥ = ⊥`. Par définition, la fonction `por` doit au moins donner lieu aux deux réductions `por ⊥ true` $\to'$ `true` et `por ⊥ true` $\to'$ `true`. Or, si l'on a bien la réduction `or ⊥ true` $\to'$ `true`, le terme `or true ⊥` n'est pas réductible en `true` selon la deuxième clause de la définition de la fonction `or`, car les vecteurs $\vec{p}^{\,1} =$ (`_ true`) et $\vec{M} =$ (`true` $\bot$) sont compatibles. Néanmoins, le parallel-or peut être défini dans le cadre de la réécriture $\to'$, en posant :

```
let por = function
  false false  →  false
| _     _      →  true
```

Sans changer la définition du prédicat de filtrage, on peut considérer qu'il s'agit aussi d'un prédicat sur les vecteurs de valeurs partielles indexé sur les matrices de valeurs partielles. Soit, étant donnés une matrice de valeurs partielles (U) de taille n par m et un vecteur de valeurs partielles $\vec{V}$ de taille n, on pose :

$$\text{match}_i[(U)](\vec{V}) \Leftrightarrow \vec{U}^i \preceq \vec{V} \text{ et } \forall j < i,\ \vec{U}^j \# \vec{V}$$

En outre, pour toute matrice de motifs (P), tout numéro de ligne i et tout vecteur de termes $\vec{M}$, on a l'équivalence $\mathrm{match}_i[(P)](\vec{M}) \Leftrightarrow \mathrm{match}_i[\omega(P)](\omega(\vec{M}))$, par le lemme 6.2.7 et la définition 6.3.1.

Le prédicat sur les valeurs partielles a les propriétés suivantes.

Proposition 6.3.4 *Soit une matrice de valeurs partielles (U). Alors, les prédicats de filtrages par les diverses lignes de la matrice (U) sont mutuellement exclusifs et monotones.*

$$\begin{array}{rcl} i \neq j \text{ et } match_i[(U)](\vec{V}) & \Rightarrow & match_j[(U)](\vec{V}) \text{ n'est pas vérifié} \\ \vec{V} \preceq \vec{V'} \text{ et } match_i[(U)](\vec{V}) & \Rightarrow & match_i[(U)](\vec{V'}) \end{array}$$

Démonstration : On remarque tout d'abord que, si le vecteur $\vec{V}$ est mieux défini que le vecteur $\vec{U^i}$, alors ces deux vecteurs sont compatibles, puisqu'ils admettent le vecteur $\vec{V}$ comme majorant commun. Soit, on a l'implication et la contraposée suivantes :

$$\vec{U^i} \preceq \vec{V} \Rightarrow \vec{U^i} \uparrow \vec{V} \qquad \vec{U^i} \# \vec{V} \Rightarrow \vec{U^i} \not\preceq \vec{V}$$

Par conséquent, que l'on ait $i < j$ ou $i > j$, si le prédicat $\mathrm{match}_i[(U)](\vec{V})$ est vérifié, le prédicat $\mathrm{match}_j[(U)](\vec{V})$ ne saurait l'être.

Pour la monotonie, on pose les hypothèses $\mathrm{match}_i[(U)](\vec{V})$ et $\vec{V} \preceq \vec{V'}$. Par transitivité de $\preceq$, on a directement $\vec{U^i} \preceq \vec{V'}$. Soit un numéro de ligne j strictement plus petit que i, si on avait $\vec{U^j} \uparrow \vec{V'}$, alors, par définition, il existerait un vecteur de valeurs $\vec{W}$, avec $\vec{U^j} \preceq \vec{W}$ et $\vec{V'} \preceq \vec{W}$. Dès lors, par transitivité de $\preceq$, le vecteur $\vec{W}$ serait un majorant commun des vecteurs $\vec{U^j}$ et $\vec{V}$. Ce qui est impossible, puisque, par hypothèse, on a $\vec{U^j} \# \vec{V}$. □

Grâce au caractère mutuellement exclusif des prédicats de filtrage, une réduction $M \xrightarrow{u}{}' N$ est entièrement caractérisée par la seule donnée du couple (M, u). En outre, un predicat de filtrage qui ne possèderait pas cette propriété ne saurait fonder une relation de réécriture non-ambiguë sans de sérieuses contraintes sur les membres droits des règles.

De même, une *réduction parallèle avec priorité*, notée $M \xrightarrow{U}{}' N$ est caractérisée par un terme initial M et un ensemble U d'occurrences de radicaux pour le programme $\mathcal{P}$. En effet, pour chaque occurrence u prise dans U, il existe une et une seule règle de réécriture avec priorité dans le programme $\mathcal{P}$ qui fait un radical du sous-terme M/u et on peut reconstituer un unique arbre de preuve de réduction parallèle à partir du couple (M, U) (cf le lemme 1.2.6). Par ailleurs, on voit facilement que le lemme 6.2.8 reste valable dans le cas de la réécriture $\rightarrow'$ et donc, que l'ordre de définition reste une mesure de la progression du calcul.

L'intérêt de la monotonie va apparaître dans le lemme suivant. Étant donné un ensemble E de vecteurs de valeurs partielles, on note $inf_{\preceq}(E)$ l'ensemble des point minimaux de E pour l'ordre $\preceq$. C'est à dire que l'on pose :

$$inf_{\preceq}(E) = \{ \vec{V} \in E \mid \forall \vec{U} \in E,\ \vec{U} \preceq \vec{V} \Rightarrow \vec{U} = \vec{V} \}$$

Lemme 6.3.5 *Soit un ensemble E de vecteurs de valeurs partielles, tel que l'ensemble E est monotone (i.e., le prédicat $\vec{V} \in E$ est monotone). Alors, l'ensemble E est entièrement décrit par la donnée de ses points minimaux :*

$$\vec{V} \in E \Leftrightarrow \exists \vec{U} \in inf_{\preceq}(E),\ \vec{U} \preceq \vec{V}$$

Démonstration : Le sens réciproque $\Leftarrow$ est une conséquence immédiate de la monotonie de l'ensemble E. Pour montrer le sens direct $\Rightarrow$, on se donne un vecteur $\vec{V}$ dans E. Soit l'ensemble $F = \{ \vec{U} \mid \vec{U} \preceq \vec{V} \}$. On sait que l'ensemble F est de cardinal fini et donc l'ensemble

$F \cap E$ est aussi de cardinal fini. Par conséquent, l'ensemble $inf_{\preceq}(F \cap E)$ est non-vide, comme ensemble des points minimaux d'un ensemble fini non-vide ($\vec{V} \in F \cap E$). C'est à dire qu'il existe au moins un vecteur $\vec{U}$ dans $inf_{\preceq}(F \cap E)$, avec $\vec{U} \preceq \vec{V}$ par définition de F. On montre ensuite l'inclusion $inf_{\preceq}(F \cap E) \subseteq inf_{\preceq}(E)$. Soit un vecteur $\vec{W}$ dans E, tel que l'on a $\vec{W} \preceq \vec{U}$. On a les relations, $\vec{W} \preceq \vec{U} \preceq \vec{V}$ et donc, $\vec{W} \in F \cap E$ par transitivité de $\preceq$. D'où il vient, $\vec{W} = \vec{U}$ par définition de $inf_{\preceq}(F \cap E)$ Soit finalement, $\vec{U} \in inf_{\preceq}(E)$. □

Corollaire 6.3.6 *Soit une matrice de valeurs partielles (U) à n colonnes et m lignes. Alors, pour tout numéro de ligne i de la matrice (P), il existe un ensemble de valeurs partielles $VEP_i[(U)]$ de valeurs partielles deux à deux incomparables, tel que, pour tout vecteur $\vec{V}$, on a l'équivalence :*

$$match_i[(U)](\vec{V}) \Leftrightarrow \exists \vec{W} \in VEP_i[(U)],\ \vec{W} \preceq \vec{V}$$

Démonstration : L'ensemble $\mathrm{VEP}_i[(U)]$ est l'ensemble des points minimaux de l'ensembles des vecteurs de valeurs partielles qui satisfont le prédicat monotone $\mathrm{match}_i[(U)]$. □

L'ensemble $\mathrm{VEP}_i[(U)]$ est l'ensemble des *valeurs étendues principales* relativement à la i-ème ligne de la matrice (U). Dans [Lav 88] l'ensemble $\mathrm{VEP}_i[(U)]$ est appelé ensemble des motifs étendus principaux et n'est défini que dans le cas des signatures finies. Dans ce cas, l'ensemble $\mathrm{VEP}_i[(U)]$ est fini et une méthode est donnée le calculer. Dans [Pue 90], cette méthode est étendue au cas général des signatures infinies. Le principe de cette dernière approche est que les ensembles de termes infinis en jeu peuvent être représentés à l'aide de leurs complémentaires qui sont toujours finis. Par exemple, dans le cas de la fonction de Fibonacci, on a la matrice (U) et les ensemble de valeurs étendues principales suivants :

$$(U) = \begin{pmatrix} \mathtt{1} \\ \mathtt{2} \\ \Omega \end{pmatrix} \quad \text{et} \quad \left\{ \begin{array}{lcl} \mathrm{VEP}_1[(U)] & = & \{\mathtt{0}\} \\ \mathrm{VEP}_2[(U)] & = & \{\mathtt{1}\} \\ \mathrm{VEP}_3[(U)] & = & \{\, V\!:\!\mathtt{int} \mid V \neq \mathtt{0} \wedge V \neq \mathtt{1}\,\} \end{array} \right.$$

Quand les types des motifs de la matrice (U) sont tels qu'il n'y a qu'un nombre fini de valeurs partielles qui les possèdent, on calcule facilement les valeurs étendues principales par énumération. Dans le cas des fonction `and` et `or`, on a :

$$(U) = \begin{pmatrix} \mathtt{true} & \mathtt{true} \\ \Omega & \Omega \end{pmatrix} \quad \text{et} \quad \left\{ \begin{array}{lcl} \mathrm{VEP}_1[(U)] & = & \{(\mathtt{true}\ \mathtt{true})\} \\ \mathrm{VEP}_2[(U)] & = & \{(\mathtt{false}\ \Omega), (\Omega\ \mathtt{false})\} \end{array} \right.$$

$$(U) = \begin{pmatrix} \Omega & \mathtt{true} \\ \mathtt{true} & \Omega \\ \Omega & \Omega \end{pmatrix} \quad \text{et} \quad \left\{ \begin{array}{lcl} \mathrm{VEP}_1[(U)] & = & \{(\Omega\ \mathtt{true})\} \\ \mathrm{VEP}_2[(U)] & = & \{(\mathtt{true}\ \mathtt{false})\} \\ \mathrm{VEP}_3[(U)] & = & \{(\mathtt{false}\ \mathtt{false})\} \end{array} \right.$$

L'approche qui va être présentée dans la suite de ce chapitre évite le calcul des valeurs étendues principales en mettant en œuvre directement le prédicat de filtrage avec priorité à l'aide d'automates finis. Dans ce cadre, seule importe l'existence des valeurs étendues principales et des motifs associés.

Définition 6.3.7 (Motifs étendus principaux) *Étant donnés une matrice de motifs (P) et un numéro de ligne i de la matrice (P), l'ensemble $MEP_i[(P)]$ des motifs principaux étendant la i-ème ligne de la matrice (P) est un ensemble de vecteurs de motifs de taille n qui est tel que, pour tout vecteur $\vec{q}$ de $MEP_i[(P)]$, il existe un et un seul vecteur de valeurs partielles $\vec{V}$ dans l'ensemble $VEP_i[\omega(P)]$, avec $\omega(\vec{q}) = \vec{V}$.*

Les motifs étendus principaux ne sont rien d'autre que les valeurs étendues principales dans lesquelles toutes les occurrences de Ω ont été remplacées par des variables choisies de façon normalisée. Remarquons que, étant donnés une matrice de motifs (P) et un vecteur de motifs $\vec{q}$ pris dans $\mathrm{MEP}_i[(P)]$, le vecteur $\vec{q}$ est lui même une instance de la i-ème ligne de la matrice (P). Par conséquent, il existe une substitution $\sigma = [\vec{p}^{\,i} \backslash \vec{q}]$, avec $\sigma(\vec{p}^{\,i}) = \vec{q}$.

Lemme 6.3.8 *Soit un programme $\mathcal{P}$. Il existe un système de réécriture $\Sigma_{\mathcal{P}}$, appelé système équivalent au programme $\mathcal{P}$, tel que, pour tout terme M, tout ensemble d'occurrences U dans M et tout terme N, on a l'équivalence :*

$$A{:}\, M \xrightarrow[\Sigma_{\mathcal{P}}]{U} N \Leftrightarrow A'{:}\, M \xrightarrow[\mathcal{P}]{U}{}' N$$

Démonstration : On considère le système $\Sigma_{\mathcal{P}}$, construit de la façon suivante. Pour chaque définition $F, (P), (E)$ du programme $\mathcal{P}$, chaque clause $F\ p_1^i\ p_2^i \ldots\ p_n^i \rightarrow' e^i$ est remplacée par ensemble de règles de la forme :

$$\{\, F\ q_1\ q_2 \ldots\ q_n \rightarrow e^i[\vec{p}^{\,i} \backslash \vec{q}] \mid \vec{q} \in \mathrm{MEP}_i[(P)]\,\}$$

Par les lemme 6.3.6 et 6.2.3 et pour toute occurrence u le sous-terme $M/u = F\ M_1\ M_2 \ldots\ M_n$ est un radical pour le programme $\mathcal{P}$, si et seulement si c'est un radical pour le système $\Sigma_{\mathcal{P}}$. On a alors les deux réductions simples :

$$A'{:}\, M \xrightarrow[\mathcal{P}]{u}{}' M[u \leftarrow e^i[\vec{p}^{\,i} \backslash \vec{M}]] \qquad A{:}\, M \xrightarrow[\Sigma_{\mathcal{P}}]{u} M[u \leftarrow (e^i[\vec{p}^{\,i} \backslash \vec{q}])[\vec{q} \backslash \vec{M}]]$$

Le même résultat d'équivalence vient facilement dans le cas des réductions parallèles. □

La définition du système équivalent s'applique aussi quand le programme initial est un programme simplement étiqueté $\mathcal{P}^g$. Il faut alors prendre note du fait que les membres droits des règles du système équivalent étiqueté $\Sigma_{\mathcal{P}}^g$ ne sont pas tout à fait étiquetés selon le procédé de la section 1.2.3. En effet, soit $\alpha \rightarrow \beta$ une règle de $\Sigma_{\mathcal{P}}^g$. Par construction, il existe une règle $\gamma \rightarrow' \delta$ du programme $\mathcal{P}^g$ et une substitution σ, avec $\alpha = \sigma(\gamma)$ et $\beta = \sigma(\delta)$. Tous les sous-termes durs à la fois dans β et δ portent bien une étiquette de tête de la forme $\tau(\beta/u) = \tau(\alpha).a_u$. Ce n'est plus le cas des sous-termes durs dans β dont l'occurrence u est de la forme $u = u_x.v$, où u_x est l'occurrence d'une variable x de δ et v une occurrence dure du terme $\sigma(x)$ qui est un sous-terme à la fois dans les termes β et α. Le sous-terme β/u est alors décoré par une étiquette $\tau(\beta/u) = \tau(\sigma(x)/v)$. C'est à dire que certaines des étiquettes du membres gauche α sont répétées sans changement dans le membre droit β, dont toutes les occurrences dures ne sont donc pas forcément créées. Cette subtilité n'est pas gênante, en particulier le lemme 1.3.6 reste valable. À savoir, en contractant simultanément toutes les occurrences d'un même radical partagé dans un graphe acyclique, on obtient toujours un graphe acyclique. Le point important est que toutes les étiquettes du terme final N d'une réduction A: $M \rightarrow N$ sont de l'une des deux formes $\tau(M/u) = l.a$ ou $\tau(M/u) = l$, où l est une étiquette qui décore un sous-terme du terme initial M.

Le système équivalent est unique aux noms des variables des motifs étendus principaux près. Par convention, on pose que toutes les variables de tous les membres gauches d'un système équivalent sont distinctes. Dans les exemples, on utilise aussi le symbole "_", avec son sens habituel. Par exemple, dans le cas des fonctions `and` et `or` de la figure 6.1, on a les deux systèmes équivalents :

$$\left\{\begin{array}{llll} \texttt{and} & \texttt{true} & \texttt{true} & \rightarrow\ \texttt{true} \\ \texttt{and} & \texttt{\_} & \texttt{false} & \rightarrow\ \texttt{false} \\ \texttt{and} & \texttt{false} & \texttt{\_} & \rightarrow\ \texttt{false} \end{array}\right. \quad \text{et} \quad \left\{\begin{array}{llll} \texttt{or} & \texttt{\_} & \texttt{true} & \rightarrow\ \texttt{true} \\ \texttt{or} & \texttt{true} & \texttt{false} & \rightarrow\ \texttt{true} \\ \texttt{or} & \texttt{false} & \texttt{false} & \rightarrow\ \texttt{false} \end{array}\right.$$

Le système équivalent à la fonction de fibonacci est composée des deux règles de base `fib 0` $\rightarrow$ `1` et `fib 1` $\rightarrow$ `1`, ainsi que d'une infinité de règles de la forme

$$\texttt{fib}\ i \rightarrow \texttt{plus (fib (moins}\ i\ \texttt{1)) (fib (moins}\ i\ \texttt{2))}$$

À raison d'une règle par constructeur `i` de la signature du type `int`, moins les constructeurs `0` et `1`.

6.3.2 Systèmes quasi-orthogonaux

Comme le montre l'exemple de la fonction `and`, les systèmes équivalents ne sont pas toujours orthogonaux, car ils peuvent admettre des paires critiques. Cependant, ces paires critiques se résolvent immédiatement et ne mettent pas en danger la confluence.

Définition 6.3.9 *Deux règles de réécriture $\alpha \rightarrow \beta$ et $\alpha' \rightarrow \beta'$ forment une paire critique triviale, si et seulement si les deux conditions suivantes sont satisfaites.*

1. *Il existe une substitution σ, avec $\sigma(\alpha) = \sigma(\alpha')$.*
2. *On a l'égalité $\sigma(\beta) = \sigma(\beta')$, pour toute substitution σ qui est telle que $\sigma(\alpha) = \sigma(\alpha')$.*

Autrement dit, les membres gauches α et α' sont unifiables et, en cas de paire critique, le résultat de la réécriture simple à partir du terme $\sigma(\alpha) = \sigma(\alpha')$ ne dépend pas de la règle choisie.

Lemme 6.3.10 *Soit un programme $\mathcal{P}$ et un système de réécriture $\Sigma_{\mathcal{P}}$ équivalent à $\mathcal{P}$. Alors, les éventuelles paires critiques du système $\Sigma_{\mathcal{P}}$ sont triviales.*

Démonstration : Soient $\alpha \rightarrow \beta$ et $\alpha' \rightarrow \beta'$ deux règles du système $\Sigma_{\mathcal{P}}$ qui donnent lieu à une paire critique. Le système $\Sigma_{\mathcal{P}}$ est un programme, à condition d'admettre que les définitions de fonctions peuvent comprendre un nombre infini de clauses. Le lemme 6.2.9 s'applique ici, c'est à dire que les paires critiques éventuelles du système $\Sigma_{\mathcal{P}}$ mettent en jeu deux règles distinctes $\alpha = F\ q_1\ q_2 \ldots\ q_n \rightarrow e$ et $\alpha' = F\ q'_1\ q'_2 \ldots\ q'_n \rightarrow e'$, ainsi qu'une substitution σ, avec $\sigma(\alpha) = \sigma(\alpha')$. Soit encore, $\sigma(\vec{q}) = \sigma(\vec{q}')$. Autrement dit, il existe une définition $F, (P), (E)$ dans le programme $\mathcal{P}$ et deux numéros de ligne i et i' dans la matrice (P), avec $\vec{q} \in \mathrm{MEP}_i[(P)]$ et $\vec{q}' \in \mathrm{MEP}_{i'}[(P)]$. Si les numéros i et i' étaient différents, alors par définition du filtrage avec priorité les vecteurs de motifs $\vec{q}$ et $\vec{q}'$ seraient incompatibles et il n'y aurait pas paire critique. On a a donc l'égalité $i = i'$. Soit, par construction des membres droits du système équivalent, il existe deux substitutions $\phi = [\vec{p}^{\,i} \backslash \vec{q}]$ et $\phi' = [\vec{p}^{\,i} \backslash \vec{q}']$, avec $e = \phi(e^i)$ et $e' = \phi'(e^i)$. Soit, en revenant aux notations en α et β, il existe un couple de termes (γ, δ) qui est tel que le terme γ est linéaire, que toutes les variables de δ sont des variables de γ et qu'il existe deux substitutions ϕ et ϕ', avec $\phi(\gamma) = \alpha$ et $\phi(\delta) = \beta$, $\phi'(\gamma) = \alpha'$ et $\phi'(\delta) = \beta'$. En outre, il existe une substitution σ, avec $\sigma(\alpha) = \sigma(\alpha')$. Soit, pour toute variable x du terme γ, on a l'égalité $(\sigma \circ \phi)(x) = (\sigma \circ \phi')(x)$. D'où il vient finalement, $(\sigma \circ \phi)(\delta) = (\sigma \circ \phi')(\delta)$, soit encore $\sigma(\beta) = \sigma(\beta')$. $\square$

Dans toute la suite de cette section on considère un système simplement étiqueté Σ^g *quasi-orthogonal*. C'est à dire que le système Σ^g est linéaire à gauche et que toutes ses paires critiques sont triviales. Sans perte de généralité, on suppose que toutes les variables de membres gauches sont distinctes. Pour simplifier, on suppose également que le système Σ^g n'est pas conditionnel (i.e., toutes les occurrences sont autorisées). La relation de réduction parallèle se définit indépendamment du caractère orthogonal du système Σ^g.

$$\frac{}{A\colon x^l \rightarrow x^l}\ (\mathrm{id}) \qquad \frac{\forall i \in [1 \ldots a],\ A_i\colon M_i \rightarrow N_i}{A\colon f^l(M_1, \ldots, M_a) \rightarrow f^l(N_1, \ldots, N_a)}\ (\mathrm{rec}) \qquad \frac{\forall x \in \mathcal{V}(\alpha),\ A_x\colon \sigma(x) \rightarrow \rho(x)}{A\colon \sigma(\alpha) \rightarrow \rho(\beta)}\ (\mathrm{red})$$

Dès lors, étant donnée une réduction parallèle A, présentée sous forme d'un arbre de preuve, on applique les définitions connues des radicaux contractés et des résidus (définitions 1.2.7 et 1.2.5).

Les lemmes de substitution 3.2.1, 2.3.6 et 3.2.2 sont encore valables (ces lemmes n'utilisent que la linéarité à gauche). Le lemme de préservation 2.3.7 des radicaux par résidus se montre facilement :

Lemme 6.3.11 *Soit une réduction A: $M \to N$ et soit un radical R de nom α et d'occurrence u dans M. Si il existe v dans $u \setminus A$, alors le sous-terme N/v est un radical de nom α.*

Démonstration: La preuve est par induction sur la structure de A. Je n'indique que les changements par rapport à la preuve du lemme 2.3.8.

1. Si A est du type (rec), avec :

$$\frac{\forall i \in [1 \ldots a],\ A_i\colon M_i \to N_i}{A\colon f^l(M_1, \ldots, M_a) \to f^l(N_1, \ldots, N_a)}\ \text{(rec)}$$

   Seul le cas $u = \epsilon$ est un peu nouveau. Supposons donc qu'il existe un radical à la racine $M = \sigma(\alpha)$. Puisque les paires critiques sont triviales, il existe pas d'occurrence de radical contracté par A qui est une occurrence dure du nom α. On peut donc appliquer le lemme 2.3.6 qui dit exactement que le terme N est un radical de nom α.

2. Si A est de type (red), avec :

$$\frac{\forall x \in \mathcal{V}(\alpha'),\ A_x\colon \sigma(x) \to \rho(x)}{A\colon \sigma(\alpha') \to \rho(\beta')}\ \text{(red)}$$

   Le seul recouvrement possible entre le radical à la racine contracté ici et le radical d'occurrence u est un recouvrement à la racine, c'est à dire que l'on a $u = \epsilon$, ou bien, $u = u_x.v$, où u_x est l'occurrence d'une variable du nom α. Dans ce dernier cas on raisonne par induction. Dans le cas $u = \epsilon$, il se peut que l'on ait $\alpha \neq \alpha'$, mais cela n'a pas d'importance, car l'occurrence ϵ n'a pas de résidu par A. □

Étant données deux réductions (i.e., deux arbres de preuve) A: $M \to N$ et B: $M \to P$, ce lemme permet de définir la réduction $B \setminus A$, comme la réduction issue du terme M et qui contracte les résidus par A des radicaux contractés par B selon les mêmes règles de réécriture que la réduction B. C'est à dire que, si la réduction B utilise la règle $\alpha \to \beta$ pour contracter le radical R, alors la réduction $B \setminus A$ utilise la même règle pour contracter les radicaux résidus de R par la réduction A.

Les paires critiques triviales donnent lieu à un troisième lemme de substitution, qui autorise la décomposition des réductions selon un préfixe qui est le plus petit commun majorant des deux noms de radicaux qui se recouvrent.

Lemme 6.3.12 *Soit un terme M, tel qu'il existe deux radicaux R et R' qui forment une paire critique triviale à la racine du terme M. Soient $\alpha \to \beta$ et $\alpha' \to \beta'$ les règles de réécritures associées aux radicaux R et R'. Soit σ l'unificateur principal des noms α et α'. Alors, pour toute réduction A: $M \to N$ qui contracte le radical R, il existe deux substitutions θ et ψ, avec $M = \theta(\sigma(\alpha))$ et $N = \psi(\sigma(\beta))$. En outre, pour toute variable y d'occurrence u_y dans le terme $\sigma(\alpha)$, il existe une réduction A_y: $\theta(y) \to \psi(y)$ et on a les l'égalités :*

$$\mathcal{R}(A) = \{\epsilon\} \cup (\bigcup_{y \in \mathcal{V}(\sigma(\alpha))} u_y.\mathcal{R}(A_y)) \qquad \forall u \in \mathcal{O}(\theta(y)),\ (u_y.u) \setminus A = \{\, v_y \mid \sigma(\beta)/v_y = y \,\}.(u \setminus A_y)$$

Démonstration: La réduction A est du type (red), avec :

$$\frac{\forall x \in \mathcal{V}(\alpha),\ A_x\colon \phi(x) \to \rho(x)}{A\colon \phi(\alpha) \to \rho(\beta)}\ \text{(red)}$$

On pose aussi $M = \phi(\alpha')$ (les variables des noms α et α' sont toutes distinctes). Par définition de l'unificateur principal, il existe une substitution θ, avec $\phi = \theta \circ \sigma$. Soit encore $M = \theta(\sigma(\alpha)) = \theta(\sigma(\alpha'))$.

La substitution ψ, si elle existe, est à définir sur les variables du terme $\sigma(\beta) = \sigma(\beta')$. Ces variables sont toutes des variables du terme $\sigma(\alpha) = \sigma(\alpha')$. Pour toute variable y d'occurrence u_y dans le terme $\sigma(\alpha) = \sigma(\alpha')$, il y deux possibilités :

1. Le sous-terme α/u_y existe et est une variable x, avec $\sigma(x) = y$. On pose alors, $u_x = u_y$, $\psi(y) = \rho(x)$ et $A_y = A_x$. On a bien $\mathcal{R}(A_y) = \mathcal{R}(A)/u_y$ par définition de $\mathcal{R}(A)$. On pose ensuite $V_x = \{\, v_x \in \mathcal{O}(\beta) \mid \beta/v_x = x \,\}$ et $V_y = \{\, v_y \in \mathcal{O}(\sigma(\beta)) \mid \sigma(\beta)/v_y = y \,\}$. Puisque l'unificateur principal σ est choisi afin que le terme $\sigma(\alpha)$ soit linéaire, on a l'implication $\sigma(\beta/v) = y \Rightarrow \beta/v = x$, d'où il vient naturellement $V_x = V_y$. Soit u une occurrence du terme $\theta(y) = \phi(x)$. On a l'égalité $(u_x.u) \setminus A = V_x.(u \setminus A_x)$ par définition des résidus. Soit finalement, $(u_y.u) \setminus A = V_y.(u \setminus A_y)$
2. L'occurrence u_y n'est pas une occurrence de α. Alors, il existe une variable x d'occurrence $u_x \prec u_y$ dans le nom α, telle que y est une variable d'occurrence w_y dans le terme $\sigma(x)$. Considérons ensuite la réduction $A_x\colon \phi(x) \to \rho(x)$. Par définition, de σ, on a l'égalité $\phi(x) = \theta(\sigma(x))$, c'est à dire que le terme linéaire $\sigma(x)$ est un préfixe du terme $\phi(x)$. Or, par définition de l'unificateur σ, les termes $\sigma(x)$ et $\sigma(\alpha'/u_x)$ sont égaux, en outre ils sont un préfixe du sous-terme $M/u_x = \phi(x)$. Dès lors, si il existait une occurrence u dure dans le terme $\sigma(x)$ et telle que le sous-terme $\phi(x)/u$ soit un radical, alors l'occurrence $u_x.u$ serait une occurrence de radical dans le terme M dure dans le nom α'. Ce qui donnerait lieu à une paire critique non-triviale ($u_x \succ \epsilon$, car $\alpha \neq x$). Autrement dit, le préfixe $\sigma(x)$ est préservé par la réduction A_x et, par le lemme 3.2.2, il existe une substitution ψ, avec $\rho(x) = \psi(\sigma(x))$. En outre, il existe une réduction $A_y\colon \theta(y) \to \psi(y)$, avec $\mathcal{R}(A_y) = \mathcal{R}(A_x)/w_y = (\mathcal{R}(A)/u_x)/w_y = \mathcal{R}(A)/u_y$. On pose ensuite $V_x = \{\, v_x \in \mathcal{O}(\beta) \mid \beta/v_x = x \,\}$ et $V_y = \{\, v_y \in \mathcal{O}(\sigma(\beta)) \mid \sigma(\beta)/v_y = y \,\}$. Par linéarité de $\sigma(\alpha)$, il vient $V_y = V_x.w_y$. Soit ensuite une occurrence u dans le sous-terme $\theta(y)$. Par définition des résidus, on a $(u_y.u) \setminus A = V_x.((w_y.u) \setminus A_x)$. Soit, par le lemme 3.2.2, $(u_y.u) \setminus A = V_x.(w_y.(u \setminus A_y))$. D'où il vient finalement $(u_y.u) \setminus A = V_y.(u \setminus A_y)$. □

En fait, quand il y effectivement paire critique, le choix du radical à contracter n'a pas grande importance, tant du point de vue du terme final d'une réduction que des résidus de radicaux.

Lemme 6.3.13 *Étant donnés un terme M et un ensemble d'occurrences de radicaux U dans le terme M. On a les deux résultats suivants :*

1. *Toutes les réductions de la forme $A\colon M \xrightarrow{U} N$ aboutissent au même terme.*
2. *Pour tout radical R d'occurrence u dans le terme M, l'ensemble $u \setminus A$ des occurrences des radicaux résidus de R est indépendant du choix des règles de réécriture utilisées dans la réduction $A\colon M \xrightarrow{U} N$.*

Démonstration : La preuve est par induction sur la structure du terme M. Si le terme M est une variable, alors il n'existe qu'une réduction vide et de type (id) qui en est issue. Sinon, le terme M est de la forme générale $M = f^l(M_1, \ldots, M_a)$. On distingue deux cas. Si on a $\epsilon \notin U$, alors toutes les réductions A avec $\mathcal{R}(A) = U$ sont du type (rec) et les propriétés *1* et *2* viennent facilement par induction appliquée aux arguments M_i. Si on a $\epsilon \in U$, soient alors deux réductions A: $M \to N$ et A': $M \to N'$, telles que l'on a $\mathcal{R}(A) = \mathcal{R}(A') = U$. Les réductions A et A' sont nécessairement du type (red) :

$$\frac{\forall x \in \mathcal{V}(\alpha),\ A_x\colon \phi(x) \to \rho(x)}{A\colon \phi(\alpha) \to \rho(\beta)}\ \text{(red)} \qquad \frac{\forall x' \in \mathcal{V}(\alpha'),\ A'_{x'}\colon \phi(x') \to \rho(x')}{A'\colon \phi(\alpha') \to \rho(\beta')}\ \text{(red)}$$

Soit σ l'unificateur principal des noms α et α'. Par définition, il existe une substitution θ, définie sur les variables du terme $\sigma(\alpha)$, avec $\phi = \theta \circ \sigma$. Par le lemme 6.3.12, il existe deux substitutions ψ et ψ', définies sur les variables de $\sigma(\alpha)$, avec $N = \psi(\sigma(\beta))$ et $N' = \psi'(\sigma(\beta'))$. Pour toute variable y du terme $\sigma(\alpha)$, il existe deux réductions A_y: $\theta(y) \to \psi(y)$ et A'_y: $\theta(y) \to \psi'(y)$, avec $\mathcal{R}(A_y) = \mathcal{R}(A'_y) = U/u_y$. Par induction il vient, $\psi(y) = \psi'(y)$. Or, on a l'égalité $\sigma(\beta) = \sigma(\beta')$ par définition des paires critiques triviales. Soit finalement, $N = \psi(\sigma(\beta)) = \psi'(\sigma(\beta')) = N'$.

Soit maintenant un radical R d'occurrence u dans le terme M. Si il y paire critique entre R et les radicaux à la racine de noms α et α', alors on a forcément $u = \epsilon$ et donc $u \setminus A = u \setminus A' = \emptyset$. Sinon, l'occurrence u n'est ni une occurrence dure de α, ni une occurrence dure de α'. Par conséquent, u n'est pas non plus une occurrence dure du terme $\sigma(\alpha)$, qui est le plus petit terme dont α et α' sont tous deux préfixes. Autrement dit, il existe une variable y d'occurrence u_y dans le terme $\sigma(\alpha)$ et une occurrence v dans le terme $\theta(y)$, avec $u = u_y.v$. Il vient ensuite :

$$\begin{array}{rcll} u \setminus A & = & \{\, v_y \mid \sigma(\beta) = y \,\}.(v \setminus A_y) & \text{lemme 6.3.12} \\ & = & \{\, v_y \mid \sigma(\beta') = y \,\}.(v \setminus A'_y) & \sigma(\beta') = \sigma(\beta) \text{ et induction} \\ & = & u \setminus A' & \text{lemme 6.3.12} \end{array}$$

□

Grâce à ce dernier lemme, on peut rectifier la définition des réductions dans les systèmes quasi-orthogonaux en posant que la réduction A: $M \stackrel{U}{\to} N$ est l'ensemble des arbres de preuve qui satisfont la propriété $\mathcal{R}(A) = U$. Cet ensemble est une classe d'équivalence des arbres de preuve construits à partir du terme M pour la relation $\mathcal{R}(A) = \mathcal{R}(B)$. C'est à dire que, ici encore, une réduction est caractérisé de façon non-ambiguë par la donnée de son terme de départ M et d'un ensemble U d'occurrences de radicaux à contracter dans le terme M.

De même, étant données deux réductions A: $M \to N$ et B: $M \to P$, la réduction résidu $B \setminus A$ est parfaitement définie comme la réduction issue du terme N et qui est telle que l'on a $\mathcal{R}(B \setminus A) = \mathcal{R}(B) \setminus A$.

Dans le cas des systèmes orthogonaux, on avait donné une description précise de la construction de l'arbre de preuve de la réduction $B \setminus A$. Ici c'est un représentant (i.e., un arbre de preuve) de la réduction $B \setminus A$ qui est construit en fonction de deux représentants des réductions A et B. Le seul cas réellement nouveau est celui où les deux réductions A et B sont de type (red), on le résoud par décomposition selon le lemme 6.3.12 et recomposition selon le lemme 3.2.1. Cette décomposition assure la preuve par induction du lemme des déplacements parallèles dans les systèmes quasi-orthogonaux.

Lemme 6.3.14 *Soient les réductions A: $M \to N$ et B: $M \to P$ dans un système quasi-orthogonal simplement étiqueté. Alors, on a les deux propriétés suivantes :*

1. *Les dérivations $A \sqcup B$ et $B \sqcup A$ aboutissent au même terme.*
2. *Pour tout radical R dans le terme M, on a l'égalité $R \setminus (A \sqcup B) = R \setminus (B \sqcup A)$.*

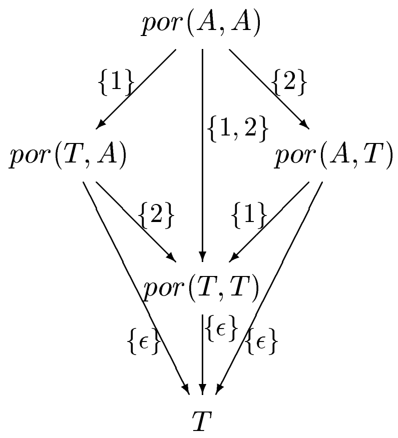


Figure 6.2 : Une classe de dérivations normalisantes dans un système quasi-orthogonal

Démonstration : La preuve de *1* est par induction sur la structure des réductions A et B. La preuve de *2* se fait par un réétiquetage adéquat. □

Cette version restreinte du lemme des déplacements parallèles suffit pour prouver une version pareillement limitée du lemme des déplacements parallèles généralisés, ainsi que le lemme du cube (cf. la preuve du lemme 4.1.6). C'est à dire que l'équivalence $\equiv$ est pertinente ($A \sqcup B \equiv B \sqcup A$) dans le cas des systèmes quasi-orthogonaux, comme elle l'était dans le cas des systèmes orthogonaux. En revanche, il est vain d'essayer de définir les radicaux nécessaires. On considère le système quasi-orthogonal formé de la définition du parallel-or $\{por(T, x) \to T, por(y, T) \to T\}$ et de la règle $A \to T$. La figure 6.2 donne les dérivations normalisantes qui sont issues du terme $por(A, A)$. Il existe, dans le terme initial $por(A, A)$, deux radicaux d'occurrences respectives 1 et 2. Comme le montrent la dérivation la plus à gauche et la dérivation la plus à droite dans la figure, aucun de ces radicaux n'est contracté par toutes les dérivations normalisantes. C'est à dire qu'il n'y a pas de radical nécessaire au sens de la définition 4.4.9 dans le terme $por(A, A)$. Par conséquent, il n'y pas non plus de radical standard à partir du terme $por(A, A)$, ni de dérivation standard dans la classe des dérivations normalisantes issues de ce terme.

Un aspect plus anecdotique des réductions de la figure 6.2 concerne la réduction A: $por(T, T) \xrightarrow{\epsilon} T$, qui admet deux arbres de preuve.

6.4 Compilation du filtrage

On va maintenant distinguer la classe des programmes qui définissent un prédicat de filtrage qui peut être calculé par un automate séquentiel.

Plutôt que de chercher à compiler le prédicat de filtrage, on définit une fonction de filtrage. Cette complication permet de traiter la construction du contractum —le terme $\sigma(\beta)$ dans la réécriture simple $\sigma(\alpha) \to \sigma(\beta)$—, en plus du filtrage proprement dit.

Définition 6.4.1 (Fonction de filtrage) *Soit un ensemble de clauses défini par le couple* $(P), (E)$, *où la matrice de motifs* (P) *possède* n *colonnes. La fonction de filtrage* $match[(P), (E)]$ *associe un terme (partiel) à tout vecteur de* n *termes (partiels)* $\vec{M}$ *de la manière suivante :*

1. *Si le vecteur* $\vec{M}$ *satisfait le prédicat de filtrage* $match_i[(P)]$ *pour un numéro de ligne* i *de la matrice* (P), *alors on pose* $match[(P), (E)](\vec{M}) = e^i[\vec{p}^{\,i} \backslash \vec{M}]$.

2. Sinon, on pose match$[(P),(E)](\vec{M}) = \Omega$.

On retrouve le prédicat de filtrage en posant (E) égal au vecteur colonne dont toutes les composantes sont la constante `true`. On continue de noter $\mathrm{match}[(P)]$ le prédicat de filtrage associé à la matrice (P). Il est clair qu'un vecteur de termes partiels $\vec{M}$ est filtré par une des lignes de la matrice de motifs (P), si et seulement si on a $\mathrm{match}[(P)](\vec{M}) = \mathtt{true}$ et que, dans le cas où $\vec{M}$ n'est pas filtré par la matrice (P), on a $\mathrm{match}[(P)](\vec{M}) = \Omega$. Cela revient à poser la valeur de vérité vrai égale à la constante `true` et la valeur de vérité faux égale au terme indéfini Ω, avec bien entendu $\Omega \preceq \mathtt{true}$.

Les primitives de la figure 6.1 définissent les deux fonctions de filtrage suivantes :

$\omega(\vec{M})$	$\mathtt{and}(\vec{M})$
$(\Omega\ \Omega)$ $(\Omega\ \mathtt{true})$ $(\mathtt{true}\ \Omega)$	Ω
(`true` `true`)	`true`
$(\Omega\ \mathtt{false})$ $(\mathtt{false}\ \Omega)$ (`false` `false`)	`false`

$\omega(\vec{M})$	$\mathtt{or}(\vec{M})$
$(\Omega\ \Omega)$ $(\Omega\ \mathtt{false})$ $(\mathtt{true}\ \Omega)$ $(\mathtt{false}\ \Omega)$	Ω
$(\Omega\ \mathtt{true})$ (`false` `true`) (`true` `true`) (`true` `false`)	`true`
(`false` `false`)	`false`

6.4.1 Filtrages simples

Un *automate séquentiel* est constitué de filtrages simples imbriqués. Les automates séquentiels sont représentés à l'aide d'expressions GAML, dont la syntaxe a été restreinte de façon appropriée :

Filtrages simples :

$F ::= \mathtt{match}\ x\ \mathtt{with}\ p \to A\ \{\mathtt{|}\ p \to A\}^*\ [\mathtt{|}\ \_ \ \to A]$

Motifs simples :

$p ::= c(x_1, x_2, \ldots, x_a)$ — et p est lineaire

Automates :

A	$::=$	F	filtrage imbriqué
	|	e	terme applicatif
	|	Ω	terme indéfini

(Dans cette description syntaxique par une grammaire BNF, les éléments optionnels sont entre crochets $[\ldots]$ et la répétition zéro fois ou plus est indiquée par une étoile en exposant $\{\ldots\}^*$.)

Les *motifs simples* employés ici sont particuliers, puisqu'ils sont de profondeur un. L'imbrication possible des filtrages simples est visible dans la règle de la classe F des expressions de filtrage simple : les résultats d'un filtrage simple sont des expressions de la classe générale A des automates. Les automates peuvent aussi être un terme applicatif, qui est un membre droit de règle de réécriture, ou le terme indéfini Ω qui traduira un échec du filtrage. La liste de clauses d'un filtrage simple F peut se terminer par une clause dont le motif est l'attrape-tout "`_`", on appelle cette clause la *clause par défaut*, les autres clauses sont les *clauses ordinaires* et les constructeurs de tête des motifs des clauses ordinaires sont les *constructeurs reconnus*. Par typage, les constructeurs reconnus par le filtrage simple F sont tous membres d'une même signature Σ. On pose que le filtrage simple F possède une clause par défaut dans le cas où certains des constructeurs de Σ ne sont pas reconnus et dans ce cas seulement.

Dans les définition qui suivent, on note F le filtrage simple ainsi décomposé :

$$F = \begin{cases} \texttt{match x with} & \\ \quad \texttt{c}_1\texttt{(x}_1\texttt{,x}_2\texttt{,}\ldots\texttt{,x}_{\texttt{a}_1}\texttt{)} & \rightarrow \; A_1 \\ | \qquad \vdots & \\ | \;\; \texttt{c}_\texttt{z}\texttt{(x}_1\texttt{,x}_2\texttt{,}\ldots\texttt{,x}_{\texttt{az}}\texttt{)} & \rightarrow \; A_z \\ | \;\; \_ & \rightarrow \; A_d \end{cases}$$

L'ensemble $\mathcal{L}(A)$ des *variables libres* d'un automate est défini comme suit :

$$\begin{array}{rcl} \mathcal{L}(F) & = & \{x\} \cup (\bigcup_{1 \leq k \leq z} \mathcal{L}(A_k) - \{x_1, x_2, \ldots, x_{a_k}\}) \cup \mathcal{L}(A_d) \\ \mathcal{L}(e) & = & \mathcal{V}(e) \\ \mathcal{L}(\Omega) & = & \emptyset \end{array}$$

On définit aussi l'ensemble $\mathcal{U}(A)$ des *variables utilisables* par un automate A comme suit :

$$\begin{array}{rcl} \mathcal{U}(F) & = & \{x\} \cup (\bigcup_{1 \leq k \leq z} \mathcal{U}(A_k) - \{x_1, x_2, \ldots, x_{a_k}\}) \cup \mathcal{U}(A_d) \\ \mathcal{U}(e) & = & \emptyset \\ \mathcal{U}(\Omega) & = & \emptyset \end{array}$$

Ainsi que l'ensemble $\mathcal{B}(A)$ des *variables liées* dans un automate :

$$\begin{array}{rcl} \mathcal{B}(F) & = & (\bigcup_{1 \leq k \leq z} \mathcal{B}(A_i) \cup \{x_1, x_2, \ldots, x_{a_k}\}) \cup \mathcal{B}(A_d) \\ \mathcal{B}(e) & = & \emptyset \\ \mathcal{B}(\Omega) & = & \emptyset \end{array}$$

Les variables utilisables sont les variables libres de l'automate A qui sont susceptibles d'être examinés par un filtrage simple. Les variables de $\mathcal{B}(A)$ sont les variables susceptibles d'être liées à des sous-termes du terme analysé par l'automate A. On s'impose deux conditions de cohérence sur les variables des automates, à savoir que les variables peuvent être liées ou utilisées au plus une fois. Soit, pour tout automate A qui est un filtrage simple, on a les deux condition :

$$x \notin \mathcal{U}(A_1) \cup \ldots \cup \mathcal{U}(A_z) \cup \mathcal{U}(A_d) \qquad \forall k \in [1 \ldots z], \; \{x_1, x_2, \ldots, x_{a_k}\} \cap \mathcal{B}(A_k) = \emptyset$$

À condition de spécifier des liaisons pour les variables libres d'un automate, la variable x qui apparaît dans la syntaxe `match x with` ... est liée à un terme applicatif. Dans ce contexte, un filtrage simple est l'examen de l'éventuel constructeur de tête de ce terme. Plus précisément, étant donnés une substitution σ, un automate A et un terme partiel N, on définit le *jugement* $\sigma \vdash A \Rightarrow N$ que l'on interprète comme "l'automate A rend le résultat N dans l'environnement σ". Les jugements $\sigma \vdash A \Rightarrow N$ se prouvent à l'aide des règles d'inférence et des axiomes donnés dans la figure 6.3. Ce type de définition est couramment utilisé pour décrire la sémantique opérationnelle de ML quand l'ordre d'évaluation est fixé [Mil 90, Wri 91], ce qui est le cas pour les automates de filtrage qui procèdent de la racine des termes vers leurs feuilles ("*top-down matching automata*").

Les règles de l'évaluation filtrage simple sont compatibles avec la définition du filtrage (définition 6.4.1), ce qui justifie la syntaxe commune. Le seul point un peu délicat est la règle (échec) ; il faut juste remarquer que la liste de clauses du filtrage simple contient toujours au moins une clause en plus de la clause par défaut. C'est à dire que la valeur indéfinie Ω, qui est compatible avec n'importe quel motif, n'est jamais filtrée par la clause par défaut.

Enfin, les automates servent à analyser des termes dont un préfixe est déjà connu. La partie résultat N d'un jugement $\sigma \vdash A \Rightarrow N$, étant le résultat de l'*exécution* de l'automate A.

$$\frac{\sigma \cup [x\backslash c_k(M_1,\ldots,M_{a_k})] \cup [x_1\backslash M_1,\ldots,x_{a_k}\backslash M_{a_k}] \vdash A_k \Rightarrow N}{\sigma \cup [x\backslash c_k(M_1,\ldots,M_{a_k})] \vdash \texttt{match x with } \ldots \texttt{ | } \texttt{c}_\texttt{k}\texttt{(x}_\texttt{1},\ldots,\texttt{x}_{\texttt{a}_\texttt{k}}\texttt{)} \to A_k \ldots \Rightarrow N} \text{(constr)}$$

$$\frac{\sigma \cup [x\backslash c(M_1,\ldots,M_a)] \vdash A_d \Rightarrow N}{\sigma \cup [x\backslash c(M_1,\ldots,M_a)] \text{ , avec } c \neq c_k \vdash \texttt{match x with } \ldots \texttt{ | \_} \to A_d \Rightarrow N} \text{(défaut)}$$

$$\sigma \cup [x\backslash M], \text{ avec } \omega(M) = \Omega \vdash \texttt{match x with} \ldots \Rightarrow \Omega \text{ (échec}_1\text{)}$$

$$\sigma \vdash \Omega \Rightarrow \Omega \text{(échec}_2\text{)} \qquad\qquad \sigma \vdash e \Rightarrow \sigma(e) \text{(succès)}$$

Figure 6.3 : Les règles de l'évaluation du filtrage simple.

Définition 6.4.2 *Soit un automate A et un terme linéaire M qui est tel que toutes les variables libres dans l'automate A sont des variables du terme M ($\mathcal{L}(A) \subseteq \mathcal{V}(M)$). On définit l'exécution Auto$[M,A]$ comme une fonction sur l'ensemble des termes (partiels), en posant :*

1. *Pour tout terme N qui est une instance du préfixe M, on calcule le terme Auto$[M,A](M)$ en prouvant le jugement :*
$$[N\backslash M] \vdash A \Rightarrow Auto[M,A](N)$$
2. *Sinon, on pose Auto$[M,A](N) = \Omega$.*

6.4.2 Exécution pas à pas des automates de filtrage

Il est parfois utile d'avoir une vue très précise du fonctionnement des automates de filtrage. Une *configuration* d'automate est un triplet (M,σ,A), où M est un terme linéaire, σ est une substitution et A est un automate. Un configuration doit obéir aux contraintes de cohérence suivantes $\mathcal{L}(A) \subseteq \mathcal{D}(\sigma)$ et $\mathcal{U}(A) \subseteq \mathcal{V}(M) \subseteq \mathcal{D}(\sigma)$ —rappelons que $\mathcal{D}(\sigma)$ est le domaine de la substitution σ qui est une fonction partielle des variables dans les termes—. Les configurations qui obéissent à ces contraintes sont susceptibles de représenter l'état d'avancement de l'analyse d'un terme (partiel) $N = \sigma(M)$ par un automate. Le terme M est le *préfixe exploré* du terme N, la substitution σ lie, entre autres, les variables libres de l'automate A aux sous-termes non encore explorés du terme M et l'automate A est l'automate utilisable pour continuer l'analyse. Cette analyse progresse par une *transition en un pas*.

Définition 6.4.3 *Une configuration (M,σ,A) donne lieu à une transition en un pas quand l'automate A est un filtrage simple de la variable x, c'est à dire lorsque l'on a*

$$A = \begin{cases} \texttt{match x with} & \\ \quad \texttt{c}_\texttt{1}\texttt{(x}_\texttt{1},\ldots,\texttt{x}_{\texttt{a}_\texttt{1}}\texttt{)} & \to \; A_1 \\ \texttt{|} \qquad \vdots & \\ \texttt{|} \;\; \texttt{c}_\texttt{z}\texttt{(x}_\texttt{1},\ldots,\texttt{x}_{\texttt{a}_\texttt{z}}\texttt{)} & \to \; A_z \\ \texttt{|} \;\; \texttt{\_} & \to \; A_d \end{cases}$$

On a ensuite deux transitions possibles selon le terme (partiel) $\sigma(x)$.

1. *Si le terme $\sigma(x)$ est de la forme $c_k(M_1, \ldots, M_{a_k})$, où le constructeur c_k est le constructeur reconnu par la k-ième clause du filtrage simple A, alors on pose :*

$$(M, \sigma, A) \hookrightarrow (M[x \backslash c_k(x_1, \ldots, x_{a_k})], \sigma \cup [x_1 \backslash M_1, \ldots, x_{a_k} \setminus M_{a_k}], A_k)$$

2. *Si le terme $\sigma(x)$ est de la forme $c(M_1, \ldots, M_a)$, où le constructeur c n'est égal à aucun des constructeurs reconnu par le filtrage simple A, alors on pose :*

$$(M, \sigma, A) \hookrightarrow (M[x \backslash c(x_1, \ldots, x_a)], \sigma \cup [x_1 \backslash M_1, \ldots, x_a \setminus M_a], A_d)$$

Où les x_i sont des variables nouvelles.

Dans tous les autres cas, il n'y a pas de transition en un pas à partir de la configuration (M, σ, A) qui est une configuration terminale.

Soit une transition en un pas $(M, \sigma, A) \hookrightarrow (M', \sigma', A')$. Il est clair que le terme M' est linéaire, que les conditions de cohérence $\mathcal{L}(A') \subseteq \mathcal{D}(\sigma')$ et $\mathcal{U}(A') \subseteq \mathcal{V}(M') \subseteq \mathcal{D}(\sigma')$ sont vérifiées, que l'on a l'égalité $\sigma'(M') = \sigma(M)$ et que le terme linéaire M' est une instance du préfixe M, selon la substitution ρ qui change la variable x en $c_k(x_1, \ldots, x_{a_k})$ et ne touche pas aux autres variables de M. On note $\hookrightarrow^*$ la relation de *transition* entre configurations qui est la fermeture réflexive et transitive de la relation de transition en un pas. Les résultats de cohérence, l'égalité $\sigma'(M') = \sigma(M)$ et l'existence d'une substitution ρ, telle que $\rho(M) = M'$ restent valables par transitivité. En particulier l'égalité $\sigma'(M') = \sigma(M)$ montre que le terme M' est bien un préfixe du terme analysé $N = \sigma(M)$ tandis l'égalité $\rho(M) = M'$ montre que ce préfixe va croissant.

La relation de transition est compatible avec les règles de l'évaluation des automates telles que données par la figure 6.3.

Lemme 6.4.4 *Soient deux configurations (M, σ, A) et (M', σ', A'), avec $(M, \sigma, A) \hookrightarrow^* (M', \sigma', A')$. Alors, les exécutions des automates A et A' rendent le même résultat dans leurs environnements respectifs. C'est à dire que l'on a l'équivalence :*

$$(\sigma \vdash A \Rightarrow N) \Leftrightarrow (\sigma' \vdash A' \Rightarrow N)$$

Démonstration : Le résultat pour les transitions en un pas est évident. Il s'étend facilement aux transitions quelconques par induction. □

Corollaire 6.4.5 *Soit un automate A et un terme linéaire M qui est tel que $\mathcal{L}(A) \subseteq \mathcal{V}(M)$. Alors, la transition $(M, \sigma, A) \hookrightarrow^* (M', \sigma', A')$, entraîne l'égalité des exécutions correspondantes :*

$$Auto[M, A](\sigma(M)) = Auto[M', A'](\sigma'(M'))$$

Les automates décrits sont déterministes (à chaque instant, au plus une transition en un pas est possible), et il n'y a pas de transition de longueur infinie (la partie automate d'une configuration diminue strictement à chaque transition). On calcule donc facilement le terme Auto$[M, A](\sigma(M))$ en atteignant d'abord une unique configuration terminale à partir de la configuration initiale (M, σ, A), puis en appliquant un des axiomes (échec) ou (succès).

L'exécution des automates est une fonction monotone sur l'ensemble des termes partiels. L'expression de cette propriété à l'aide des transitions entre configurations est la plus commode.

Lemme 6.4.6 *Soient une transition $(M, \sigma, A) \hookrightarrow^* (M', \sigma', A')$ et une substitution ρ qui est telle que pour toute variable x utilisable dans l'automate A ($x \in \mathcal{U}(A)$), on a la relation $\sigma(x) \preceq \rho(x)$. Alors, il existe une transition de la forme $(M, \rho, A) \hookrightarrow^* (M', \rho', A')$*

Démonstration : La preuve est par induction sur le nombre de pas de la transition initiale. Si cette transition est nulle, alors il n'y a rien à prouver. Sinon, la transition $(M, \sigma, A) \hookrightarrow^* (M', \sigma', A')$ se décompose en un pas $(M, \sigma, A) \hookrightarrow (M'', \sigma'', A'')$ et une transition $(M'', \sigma'', A'') \hookrightarrow^* (M', \sigma', A')$. L'automate A est nécessairement un filtrage simple de la forme `match x with ... | _` $\to A_d$. On suppose —par exemple— que le terme $\sigma(x)$ est de la forme $c(M_1, \ldots, M_a)$, où le constructeur c n'est pas reconnu par le filtrage simple A. Dès lors, on a $M'' = M[x \backslash c(x_1, \ldots, x_a)]$, $\sigma'' = \sigma \cup [x_1 \backslash M_1, \ldots, x_a \setminus M_a]$ et $A'' = A_d$. Par hypothèse, on a $\sigma(x) \preceq \rho(x)$. Soit ici, $\rho(x) = c(N_1, \ldots, N_a)$, avec $M_1 \preceq N_1, \ldots M_a \preceq N_a$. On a donc la transition en un pas $(M, \rho, A) \hookrightarrow (M'', \rho'', A'')$, avec $\rho'' = \rho \cup [x_1 \backslash N_1, \ldots, x_a \setminus N_a]$. Soit y une variable utilisable par l'automate A'' Si y est l'un des x_i, alors on a $\sigma''(y) = M_i \preceq N_i = \rho''(y)$. Sinon y est utilisable par A et on a $\sigma''(y) = \sigma(y) \preceq \rho(y) = \rho''(y)$. Par induction, il existe donc une transition de la forme $(M'', \rho'', A'') \hookrightarrow^* (M', \rho', A')$. Soit finalement, on a la transition $(M, \rho, A) \hookrightarrow^* (M', \rho', A')$. □

Enfin, le préfixe exploré d'une configuration quelconque calculée à partir d'une configuration initiale donnée, est un terme reconnu par l'automate initial.

Lemme 6.4.7 *Soient une transition $(M, \sigma, A) \hookrightarrow^* (M', \sigma', A')$. Alors, il existe une substitution ρ définie sur l'ensemble des termes partiels et qui est telle que l'on a $\rho(M) = M'$, ainsi que la transition $(M, \rho, A) \hookrightarrow^* (M', \rho', A')$.*

Démonstration : On raisonne par induction sur le nombre de pas de la transition en jeu. Si cette transition est vide, alors, on pose la substitution ρ égale à la substitution identité définie sur les variables de M. Sinon, il existe une transition en un pas $(M, \sigma, A) \hookrightarrow (M'', \sigma'', A'')$ et une transition $(M'', \sigma'', A'') \hookrightarrow^* (M', \sigma', A')$. Par hypothèse d'induction, il existe une substitution ϕ qui est telle que $\phi(M) = M''$ et que l'on a la transition $(M'', \phi, A'') \hookrightarrow^* (M', \phi', A')$. L'automate A est nécessairement un filtrage simple de la forme `match x with ... | _` $\to A_d$. On suppose —par exemple— que le terme $\sigma(x)$ est de la forme $c_k(M_1, \ldots, M_{a_k})$. On a alors $M'' = M[x \backslash c_k(x_1, \ldots, x_{a_k})]$. On définit la substitution ρ en posant $\rho(x) = c(\phi(x_1), \ldots, \phi(x_a))$ et $\rho(y) = \phi(y)$ pour toutes les autres variables du domaine de la substitution ϕ. On a alors la transition en un pas $(M, \rho, A) \hookrightarrow (M'', \rho'', A'')$ avec $\rho'' = \rho \cup [x_1 \backslash \phi(x_1), \ldots, x_a \setminus \phi(x_a)]$. Par construction de ρ'', les substitutions ρ'' et ϕ ne peuvent différer que sur la variable x qui n'est pas élément de $\mathcal{U}(A'')$. On peut donc appliquer le lemme 6.4.6 et affirmer l'existence d'une transition $(M'', \rho'', A'') \hookrightarrow^* (M', \rho', A')$. □

6.4.3 Compilation de la fonction de filtrage

La compilation est définie par une fonction de compilation, définie de l'ensembles des définitions par filtrage vers l'ensemble des automates séquentiels. Plus précisément, la *fonction de compilation* $\mathcal{C}$ prend trois arguments et un appel typique s'écrit $\mathcal{C}(\vec{x}, (P), (E))$. Le vecteur de n variables $\vec{x} = (x_1\ x_2\ \ldots\ x_n)$ abstrait le vecteur de termes partiels donné en argument à la fonction de filtrage à compiler. Autrement dit et pour être tout à fait complet, on pose la valeur $\mathcal{C}((M_1\ M_2\ \ldots\ M_n), (P), (E))$ égale à l'exécution selon les règles de la figure 6.3 de l'automate $\mathcal{C}((x_1\ x_2\ \ldots\ x_n), (P), (E))$ dans l'environnement $[x_1 \backslash M_1, x_2 \backslash M_2, \ldots, x_n \backslash M_n]$:

$$[x_1 \backslash M_1, x_2 \backslash M_2, \ldots, x_n \backslash M_n] \vdash \mathcal{C}((x_1\ x_2\ \ldots\ x_n), (P), (E)) \Rightarrow \mathcal{C}((M_1\ M_2\ \ldots\ M_n), (P), (E))$$

La matrice de motifs (P) possède m lignes et n colonnes. Initialement, cette matrice est la matrice de motifs à compiler. Par la suite, elle représente les sous-motifs des motifs initiaux qui n'ont pas encore étés traités. Enfin, le troisième argument de la fonction $\mathcal{C}$ est un vecteur colonne (E), composé de m termes applicatifs, il représente les termes résultats du filtrage initial qui sont encore atteignables à un stade donné de la compilation.

La fonction $\mathcal{C}$ est définie inductivement comme suit :

1. Si l'entier n est nul, alors la compilation du filtrage est terminée. Si la matrice de motifs est complètement vide ($m = 0$), alors le filtrage échoue ;

$$\mathcal{C}((),(),()) = \Omega$$

Dans le cas contraire ($m > 0$), il y a une ligne vide ou plus dans la matrice (P) et la première composante du vecteur colonne (E) est le résultat de la compilation :

$$\mathcal{C}((), \begin{pmatrix} \\ \\ \vdots \\ \\ \end{pmatrix}, \begin{pmatrix} e^1 \\ e^2 \\ \vdots \\ e^m \end{pmatrix}) = e^1$$

2. Si la première ligne de motifs n'est composée que de variables, alors le filtrage par cette première ligne réussit toujours et rend l'expression résultat correspondante :

$$\mathcal{C}((x_1\ x_2\ \ldots\ x_n), \begin{pmatrix} y_1\ y_2 \cdots y_n \\ p_1^2\ p_2^2 \cdots p_n^2 \\ \vdots \\ p_1^m\, p_2^m \cdots p_n^m \end{pmatrix}, \begin{pmatrix} e^1 \\ e^2 \\ \vdots \\ e^m \end{pmatrix}) = e^1[y_1\backslash x_1,\, y_2\backslash x_2, \ldots, y_n\backslash x_n]$$

3. Si une colonne de motifs —la première, par exemple— n'est composée que de variables, alors la composante correspondante du vecteur de variables $\vec{x}$ n'a pas besoin d'être examinée :

$$\mathcal{C}((x_1\ x_2\ \ldots\ x_n), \begin{pmatrix} y^1\ p_2^1 \cdots p_n^1 \\ y^2\ p_2^2 \cdots p_n^2 \\ \vdots \\ y^m\, p_2^m \cdots p_n^m \end{pmatrix}, \begin{pmatrix} e^1 \\ e^2 \\ \vdots \\ e^m \end{pmatrix}) = \mathcal{C}((x_2\ \ldots\ x_n), \begin{pmatrix} p_2^1 \cdots p_n^1 \\ p_2^2 \cdots p_n^2 \\ \vdots \\ p_2^m \cdots p_n^m \end{pmatrix}, \begin{pmatrix} e^1[y^1\backslash x_1] \\ e^2[y^2\backslash x_1] \\ \vdots \\ e^m[y^m\backslash x_1] \end{pmatrix})$$

4. Si aucune des règles *1* ou *2* ci-dessus ne s'applique, on peut aussi faire progresser le filtrage en examinant une variables x_i, qui est telle que la i-ème colonne de la matrice (P) comprend au moins un motif qui n'est pas une variable. Pour le moment le choix de cette variable est arbitraire et le résultat de la compilation en dépend *a priori*. Lorsque la variable x_i est choisie on dit que la compilation progresse en suivant l'indice i. Supposons, par exemple, que la première variable x_1 soit choisie. Soient alors $\Sigma = \{\, c_k \mid 1 \leq k \leq z \,\}$ l'ensemble des constructeurs qui apparaissent en tête d'un motif de la première colonne. Pour chaque indice k de l'intervalle $[1 \ldots z]$, une nouvelle matrice de motifs (P_k) est construite, ainsi qu'un nouveau vecteur colonne (E_k). Les lignes de la nouvelle matrice (P_k) sont construites à partir des lignes de (P) qui peuvent admettre des instances $\vec{M}$ dont la première composante est de la forme $M_1 = c_k(M_1, \ldots, M_{a_k})$. Les composantes du vecteur colonne (E_k) sont des composantes du vecteur colonne (E) sélectionnées selon le même principe. Le cas échéant, on procède à un changement de variable approprié. Plus précisément, pour chaque numéro de ligne i, le tableau suivant donne la construction d'une ligne de la matrice (P_k), ainsi que de la composante

associée dans le vecteur de résultats (E_k) :

p_1^i	ligne de (P_k)	ligne de (E_k)
y	$\_\ \ \_ \cdots \_\quad p_2^i \cdots p_n^i$	$e^i[y\backslash x_1]$
$c_k(q_1^i, q_2^i, \ldots, q_{a_k}^i)$	$q_1^i\quad q_2^i \cdots q_{a_k}^i\quad p_2^i \cdots p_n^i$	e^i
$c_{k'}(q_1^i, q_2^i, \ldots, q_{a_{k'}}^i)\ (k \neq k')$	pas de ligne numéro i	

Si l'ensemble de constructeurs Σ ne forme pas une signature complète, alors il existe des vecteurs $\vec{M}$ dont la première composante est de la forme $M_1 = c(M_1, \ldots, M_a)$, où c n'est pas un constructeur de Σ. De tels vecteurs peuvent être filtrés par les lignes de motifs de la matrice (P) dont la première composante est une variable. Pour traiter ce cas, une *matrice par défaut* (P_d) est construite ligne à ligne (une matrice similaire est construite en *3* ci-dessus) :

p_1^i	ligne de (P_d)	ligne de (E_d)
y	$p_2^i \cdots p_n^i$	$e_i[y\backslash x_1]$
$c(q_1^i, q_2^i, \ldots, q_a^i)$	pas de ligne numéro i	

L'ordre des lignes est bien entendu conservé dans les matrices (P_k) et (P_d). On pose ensuite :

$$\mathcal{C}((x_1\ x_2\ \ldots\ x_n),(P),(E)) = \left\{ \begin{array}{ll} \multicolumn{2}{l}{\texttt{match x}_1\texttt{ with}} \\ \quad \texttt{c}_1\texttt{(y}_1\texttt{,...,y}_{\texttt{a}_1}\texttt{)} & \to \mathcal{C}((y_1\ \ldots\ y_{a_1}\ x_2\ \ldots\ x_n),(P_1),(E_1)) \\ \texttt{| c}_2\texttt{(y}_1\texttt{,...,y}_{\texttt{a}_2}\texttt{)} & \to \cdots \\ & \vdots \\ \texttt{| c}_z\texttt{(y}_1\texttt{,...,y}_{\texttt{a}_z}\texttt{)} & \to \mathcal{C}((y_1\ \ldots\ y_{a_z}\ x_2\ \ldots\ x_n),(P_z),(E_z)) \\ \texttt{| \_} & \to \mathcal{C}((x_2\ \ldots\ x_n),(P_d),(E_d)) \end{array} \right.$$

Où les y_j sont des variables nouvelles.

La compilation termine toujours car, à chaque appel récursif de $\mathcal{C}$, le deuxième argument décroît strictement pour l'ordre défini sur les matrices de motifs (P) par l'ordre lexicographique sur les paires d'entiers $(N_c(P), N_v(P))$, où $N_c(P)$ et $N_v(P)$ sont les sommes sur tous les motifs de (P) des fonctions n_c et n_v définies par :

$$\left\{ \begin{array}{rcl} n_c(x) & = & 0 \\ n_c(c(p_1, \ldots, p_a)) & = & 1 + n_c(p_1) + \cdots + n_c(p_a) \end{array} \right. \qquad \left\{ \begin{array}{rcl} n_v(x) & = & 1 \\ n_v(c(p_1, \ldots, p_a)) & = & 0 \end{array} \right.$$

Un point important concerne le terme indéfini Ω : celui-ci ne peut être produit par la compilation que lorsque la matrice de motifs est vide (cf *1*, ci-dessus), cas qui ne peut être introduit que par l'étape non-triviale de compilation *4*, lorsque l'ensemble des constructeurs filtré n'est par exhaustif. C'est alors la matrice par défaut (P_d) qui peut être vide si la colonne choisie ne comporte aucune variable. On a donc déjà de fortes raisons de penser que, si un automate rend le terme indéfini, alors c'est que le filtrage compilé n'est pas exhaustif et que l'automate a reconnu un terme qui n'est filtré par aucun motif.

Un point mineur concerne l'étape inductive triviale *3*. Ce cas est en fait superflu, car on peu laisser en place les colonnes de variables, jusqu'à aboutir à une matrice dont la première ligne ne comporte que des variables. Toutefois, l'étape *3* diminue notablement la tailles des matrices (P), c'est pourquoi je la conserve, à titre d'optimisation possible du processus de compilation.

À titre d'exemple, considérons le filtrage défini par la fonction `or` de la figure 6.1. L'appel initial de la fonction de compilation $\mathcal{C}$ est donné dans le haut de la figure 6.4. Il y a deux débuts de

$$\mathcal{C}((x\ y), \begin{pmatrix} \texttt{\_} & \texttt{true} \\ \texttt{true} & \texttt{\_} \\ \texttt{\_} & \texttt{\_} \end{pmatrix}, \begin{pmatrix} \texttt{true} \\ \texttt{true} \\ \texttt{false} \end{pmatrix})$$

$$\begin{array}{l} \texttt{match x with} \\ \quad \texttt{true} \rightarrow \mathcal{C}((y), \begin{pmatrix} \texttt{true} \\ \texttt{\_} \\ \texttt{\_} \end{pmatrix}, \begin{pmatrix} \texttt{true} \\ \texttt{true} \\ \texttt{false} \end{pmatrix}) \\ \texttt{| \_} \quad \rightarrow \mathcal{C}((y), \begin{pmatrix} \texttt{true} \\ \texttt{\_} \end{pmatrix}, \begin{pmatrix} \texttt{true} \\ \texttt{false} \end{pmatrix}) \\ \qquad \vdots \\ \qquad \vdots \end{array}$$

$$\texttt{or}_1(x\ y) = \begin{cases} \texttt{match x with} \\ \quad \texttt{true} \rightarrow \texttt{(match y with} \\ \qquad\qquad \texttt{true} \rightarrow \texttt{true} \\ \qquad\qquad \texttt{| \_} \quad \rightarrow \texttt{true)} \\ \texttt{| \_} \quad \rightarrow \texttt{(match y with} \\ \qquad\qquad \texttt{true} \rightarrow \texttt{true} \\ \qquad\qquad \texttt{| \_} \quad \rightarrow \texttt{false)} \end{cases}$$

$$\begin{array}{l} \texttt{match y with} \\ \quad \texttt{true} \rightarrow \mathcal{C}((x), \begin{pmatrix} \texttt{\_} \\ \texttt{true} \\ \texttt{\_} \end{pmatrix}, \begin{pmatrix} \texttt{true} \\ \texttt{true} \\ \texttt{false} \end{pmatrix}) \\ \texttt{| \_} \quad \rightarrow \mathcal{C}((x), \begin{pmatrix} \texttt{true} \\ \texttt{\_} \end{pmatrix}, \begin{pmatrix} \texttt{true} \\ \texttt{false} \end{pmatrix}) \\ \qquad \vdots \\ \qquad \vdots \end{array}$$

$$\texttt{or}_2(x\ y) = \begin{cases} \texttt{match y with} \\ \quad \texttt{true} \rightarrow \texttt{true} \\ \texttt{| \_} \quad \rightarrow \texttt{(match x with} \\ \qquad\qquad \texttt{true} \rightarrow \texttt{true} \\ \qquad\qquad \texttt{| \_} \quad \rightarrow \texttt{false)} \end{cases}$$

Figure 6.4 : Les deux compilations de la fonction `or`

$$\mathcal{C}((x\ y), \begin{pmatrix} \texttt{true} & \texttt{true} \\ \texttt{\_} & \texttt{\_} \end{pmatrix}, \begin{pmatrix} \texttt{true} \\ \texttt{false} \end{pmatrix})$$

$$\vdots$$

$$\texttt{and}_1(x\ y) = \begin{cases} \texttt{match x with} \\ \quad \texttt{true} \rightarrow \texttt{(match y with} \\ \qquad\qquad \texttt{true} \rightarrow \texttt{true} \\ \qquad\qquad \texttt{| \_} \quad \rightarrow \texttt{false)} \\ \texttt{| \_} \quad \rightarrow \texttt{false} \end{cases}$$

$$\texttt{and}_2(x\ y) = \begin{cases} \texttt{match y with} \\ \quad \texttt{true} \rightarrow \texttt{(match x with} \\ \qquad\qquad \texttt{true} \rightarrow \texttt{true} \\ \qquad\qquad \texttt{| \_} \quad \rightarrow \texttt{false)} \\ \texttt{| \_} \quad \rightarrow \texttt{false} \end{cases}$$

Figure 6.5 : Les deux compilations de la fonction `and`

compilation possibles, selon que l'on choisisse de commencer par un filtrage simple de la variable x ou de la variable y. La suite de ces compilations est facile et on obtient finalement deux automates $\texttt{or}_1$ et $\texttt{or}_2$ (donnés dans le bas de la figure 6.4). Les deux automates produits sont syntaxiquement différents, ils diffèrent aussi sémantiquement. En effet, on a $\texttt{or}_1(\Omega\ \texttt{true}) = \Omega$ —puisque le filtrage simple sur la première composante Ω échoue d'entrée— et $\texttt{or}_2(\Omega\ \texttt{true}) = \texttt{true}$. Pour toutes les autres valeurs possibles du vecteur de deux booléens $\vec{V}$, on a $\texttt{or}_1(\vec{V}) = \texttt{or}_2(\vec{V}) = \texttt{or}(\vec{V})$. Soit finalement, l'automate $\texttt{or}_2$ est le seul qui met correctement en œuvre la fonction `or`. De même, la figure 6.5 illustre les deux compilations possibles de la fonction `and` qui sont toutes deux incorrectes, car on a les égalités $\texttt{and}(\texttt{false}\ \Omega) = \texttt{and}(\Omega\ \texttt{false}) = \texttt{false}$, $\texttt{and}_1(\Omega\ \texttt{false}) = \Omega$ et $\texttt{and}_2(\texttt{false}\ \Omega) = \Omega$.

6.5 Compilation correcte

6.5.1 Directions

Une compilation est correcte quand elle produit un automate dont l'exécution est égale à la fonction de filtrage définie par les clauses compilées.

Définition 6.5.1 *Étant donnés une matrice de motifs (P) de largeur n et un vecteur colonne de termes (E), la compilation $\mathcal{C}((\vec{x}, (P), (E))$ est correcte, si et seulement, pour tout vecteur de termes partiels $\vec{M}$ de taille n, on a l'égalité :*

$$\mathcal{C}(\vec{M}, (P), (E)) = \mathit{match}[(P), (E)](\vec{M})$$

Comme le montre l'automate $\texttt{or}_1$ de la section précédente, la fonction de compilation $\mathcal{C}$ n'est pas toujours correcte. Dans cet exemple, le problème vient de l'examen intempestif de la première composante du vecteur $(\Omega\ \texttt{true})$ qui est par ailleurs une instance de la la première ligne de la matrice de motifs en jeu. Les directions sont les indices de colonne pour lesquelles une telle situation est impossible.

Définition 6.5.2 (Directions) *Soit une matrice de motifs (P) de largeur n. Soit d un indice de colonne dans (P) $(1 \leq d \leq n)$. Par définition, l'indice d est une direction de la matrice (P) et on note $d \in \mathit{Dir}[(P)]$, si et seulement si les deux conditions suivantes sont remplies :*

1. *Le vecteur $\vec{\Omega} = (\Omega\ \Omega\ \ldots\ \Omega)$ n'est pas filtré par la matrice (P).*
2. *Il n'existe pas de vecteur de valeurs partielles $\vec{V} = (V_1\ V_2\ \ldots\ V_n)$ qui est filtré par la matrice (P) et dont la composante V_d est égale à Ω.*

La restriction $\text{match}[(P)](\vec{\Omega}) = \Omega$ sert à éviter le cas pathologique où la compilation du filtrage par la matrice (P) se résoud immédiatement par un appel terminal (cas *1* et *2* de la définition de la fonction $\mathcal{C}$).

On définit aussi l'ensemble $\text{Dir}_i[(P)]$ des *directions pour la ligne numéro i de la matrice de motifs (P)*.

$$\text{Dir}_i[(P)] = \{\, d \in [1\ldots n] \mid \text{match}_i[(P)](V_1\ V_2\ \ldots\ V_n) \Rightarrow V_d \succ \Omega \,\}$$

L'ensemble des directions de une matrice de motifs (P) est aisément calculable comme intersection des ensembles des directions pour toutes les lignes de la matrice (P). Par exemple, si on reprend l'exemple de la fonction `or`, il vient :

$$(P) = \begin{pmatrix} \_ & \texttt{true} \\ \texttt{true} & \_ \\ \_ & \_ \end{pmatrix} \quad \text{et} \quad \begin{array}{|l|} \hline \text{match}_1[(P)](\vec{V}) \Leftrightarrow \texttt{true} \preceq V_2 \\ \hline \text{match}_2[(P)](\vec{V}) \Leftrightarrow \texttt{true} \preceq V_1 \wedge \texttt{true} \# V_2 \\ \hline \text{match}_3[(P)](\vec{V}) \Leftrightarrow \texttt{true} \# V_1 \wedge \texttt{true} \# V_2 \\ \hline \end{array}$$

Soit ici, $\mathrm{Dir}_1[(P)] = \{2\}$, $\mathrm{Dir}_2[(P)] = \{1, 2\}$, $\mathrm{Dir}_3[(P)] = \{1, 2\}$ et $\mathrm{Dir}[(P)] = \{2\}$. Pour la description d'une méthode générale de calcul des directions, voir la section 6.5.4.

6.5.2 Une condition suffisante de correction

Les directions donnent une condition suffisante de correction des automates produits par la fonction de compilation $\mathcal{C}$.

Lemme 6.5.3 *Soit une définition par filtrage. Une compilation de cette définition est correcte, quand, à chaque appel de type 4 de la fonction $\mathcal{C}$ de la forme $\mathcal{C}((x_1\ x_2\ \ldots\ x_n), (P), (E))$, il existe une direction d de la matrice (P) et que la compilation se poursuit en suivant cette direction, c'est à dire en produisant un filtrage simple qui porte sur la variable x_d.*

Démonstration: La preuve est par induction sur la définition de la fonction de compilation $\mathcal{C}$:

1. Si la liste de variables est vide, alors on vérifie facilement qu'il y a égalité du filtrage et de sa compilation.
$$\mathcal{C}((), (), ()) = \mathrm{match}[(), ()]() = \Omega$$
$$\mathcal{C}((), \begin{pmatrix} \\ \\ \vdots \\ \\ \end{pmatrix} \begin{pmatrix} e^1 \\ e^2 \\ \vdots \\ e^m \end{pmatrix}) = \mathrm{match}[\begin{pmatrix} \\ \\ \vdots \\ \\ \end{pmatrix} \begin{pmatrix} e^1 \\ e^2 \\ \vdots \\ e^m \end{pmatrix}]() = e^1$$
2. Si les motifs de la première ligne sont tous des variables, alors, ici encore, il y a égalité entre le filtrage et sa compilation. Posons $\vec{p}^{\,1} = (y_1\ y_2\ \ldots\ y_n)$, pour tout vecteur de termes partiels $\vec{M} = (M_1\ M_2\ \ldots\ M_n)$ on a $(y_1\ y_2\ \ldots\ y_n) \preceq (M_1\ M_2\ \ldots\ M_n)$ et donc $\mathrm{match}[(P), (E)](\vec{M}) = e^1[y_1\backslash M_1, y_2\backslash M_2, \ldots, y_n\backslash M_n]$ par définition de la fonction de filtrage.
3. Dans le cas où une colonne de la matrice de motifs —la première, par exemple— n'est composée que de variables, alors, par définition de $\mathcal{C}$, on a l'égalité :
$$\mathcal{C}(\vec{M}, (P), (E)) = \mathcal{C}((M_2\ \ldots\ M_n), (P_d), (E_d)[x_1\backslash M_1])$$
D'où il vient, par hypothèse d'induction appliquée à la matrice (P_d) :
$$\mathcal{C}(\vec{M}, (P), (E))) \quad = \quad \mathrm{match}[(P_d), (E_d)[x_1\backslash M_1]](M_2\ \ldots\ M_n) \tag{6.1}$$
Quel que soit le numéro de ligne i compris entre 1 et m, on a les deux équivalences :
$$(y^i\ p_2^i\ \ldots\ p_n^i) \preceq_\omega (M_1\ M_2\ \ldots\ M_n) \quad \Leftrightarrow \quad (p_2^i\ \ldots\ p_n^i) \preceq_\omega (M_2\ \ldots\ M_n)$$
$$(y^i\ p_2^i\ \ldots\ p_n^i) \#_\omega (M_1\ M_2\ \ldots\ M_n) \quad \Leftrightarrow \quad (p_2^i\ \ldots\ p_n^i) \#_\omega (M_2\ \ldots\ M_n)$$
C'est à dire que l'on a l'égalité :
$$\mathrm{match}[(P_d), (E_d)[x_1\backslash M_1]](M_2\ \ldots\ M_n) = \mathrm{match}[(P), (E)](\vec{M})$$
Ce qui permet de conclure, $\mathcal{C}(\vec{M}, (P), (E))) = \mathrm{match}[(P), (E)](\vec{M})$ par l'équation 6.1.

4. Dans le cas de l'étape inductive *4*, on réutilise les notations de la définition de $\mathcal{C}$, en supposant que la compilation progresse en suivant l'indice 1. C'est à dire que l'appel de la fonction $\mathcal{C}$ se résoud en :

$$\mathcal{C}(\vec{x},(P),(E)) = \left\{ \begin{array}{lcl} \texttt{match x}_1\texttt{ with} & & \\ \quad \texttt{c}_1\texttt{(y}_1\texttt{,...,y}_{\texttt{a}_1}\texttt{)} & \rightarrow & \mathcal{C}((y_1\ \ldots\ y_{a_1}\ x_2\ \ldots\ x_n),(P_1),(E_1)) \\ \texttt{| c}_2\texttt{(y}_1\texttt{,...,y}_{\texttt{a}_2}\texttt{)} & \rightarrow & \cdots \\ & \vdots & \\ \texttt{| c}_\texttt{z}\texttt{(y}_1\texttt{,...,y}_{\texttt{a}_\texttt{z}}\texttt{)} & \rightarrow & \mathcal{C}((y_1\ \ldots\ y_{a_z}\ x_2\ \ldots\ x_n),(P_z),(E_z)) \\ \texttt{| \_} & \rightarrow & \mathcal{C}((x_2\ \ldots\ x_n),(P_d),(E_d)) \end{array} \right.$$

Pour tout vecteur de termes $\vec{M}$ de taille n, on distingue trois cas selon la valeur de la première composante M_1 de $\vec{M}$.

Si le terme M_1 n'a pas de constructeur de tête ($\omega(M_1) = \Omega$), alors le filtrage simple de M_1 échoue (règle (échec)) et on a $\mathcal{C}(\vec{M},(P),(E)) = \Omega$. Or, par hypothèse, l'indice 1 est une direction de la matrice de motifs (P) et le vecteur de valeurs $\omega(\vec{M}) = (\Omega\ \omega(M_2)\ \ldots\ \omega(M_n))$ n'est pas filtré par une des lignes de la matrice $\omega(P)$. Soit $\mathrm{match}[(P),(E)](\vec{M}) = \Omega$ par définition de la fonction de filtrage.

Si le terme M_1 est de la forme $c(N_1,\ldots,N_a)$, où le constructeur c n'est pas élément de l'ensemble Σ des constructeurs de tête des motifs de la première colonne de la matrice (P), alors, par la règle (défaut) du filtrage simple, on prouve l'égalité suivante en une étape :

$$\mathcal{C}(\vec{M},(P),(E)) = C((M_2\ \ldots\ M_n),(P_d),(E_d)[x_1\backslash M_1]))$$

Soit, par hypothèse d'induction :

$$\mathcal{C}(\vec{M},(P),(E)) = \mathrm{match}[(P_d),(E_d)[x_1\backslash M_1]](M_2\ \ldots\ M_n)$$

Or, pour tout numéro de ligne i compris entre 1 et m, on a ici les deux équivalences :

$$(p_1^i\ \ldots\ p_n^i) \preceq_\omega ((c(N_1,\ldots,N_a))\ \ldots\ M_n) \quad\Leftrightarrow\quad p_1^i = y^i \text{ et } (p_2^i\ \ldots\ p_n^i) \preceq_\omega (M_2\ \ldots\ M_n)$$

$$(p_1^1\ \ldots\ p_n^i) \#_\omega ((c(N_1,\ldots,N_a))\ \ldots\ M_n) \quad\Leftrightarrow\quad \left\{ \begin{array}{l} p_1^i = c_k(q_1^i,\ldots,q_{a_k}^i),\ \text{avec } c_k \in \Sigma \\ \qquad\text{ou bien} \\ p_1^i = y^i \text{ et } (p_2^i\ \ldots\ p_n^i) \#_\omega (M_2\ \ldots\ M_n) \end{array} \right.$$

Autrement dit, on ici équivalence des filtrages par les matrices (P) et (P_d), comme on l'avait dans le cas *3*, et on peut conclure pareillement.

Enfin, si la première composante du vecteur $\vec{M}$ est de la forme $M_1 = c_k(N_1,\ldots,N_{a_k})$, où c_k est un constructeur de Σ, alors, par la règle (constr) du filtrage simple et l'hypothèse d'induction, il vient :

$$\mathcal{C}(\vec{M},(P),(E)) = \mathrm{match}[(P_k),(E_k)[x_1\backslash c_k(N_1,\ldots,N_{a_k})]](N_1\ \ldots\ N_{a_k}\ M_2\ \ldots\ M_n)$$

Par ailleurs, pour tout numéro de ligne valide, on a les deux équivalences :

$$\vec{p}^i \preceq_\omega \vec{M} \quad \Leftrightarrow \quad \begin{cases} p_1^i = y^i \text{ et } (p_2^i \ \dots \ p_n^i) \preceq_\omega (M_2 \ \dots \ M_n) \\ \qquad \text{ou bien} \\ p_1^i = c_k(q_1^i, \dots, q_{a_k}^i) \text{ et } (q_1^i \ \dots \ q_{a_k}^i \ p_2^i \ \dots \ p_n^i) \preceq_\omega (N_1 \ \dots \ N_{a_k} \ M_2 \ \dots \ M_n) \end{cases}$$

$$\vec{p}^i \#_\omega \vec{M} \quad \Leftrightarrow \quad \begin{cases} p_1^i = y^i \text{ et } (p_2^i \ \dots \ p_n^i) \#_\omega (M_2 \ \dots \ M_n) \\ \qquad \text{ou bien} \\ p_1^i = c_{k'}(q_1^i, \dots, q_{a_{k'}}^i) \text{ avec } k \neq k' \\ \qquad \text{ou bien} \\ p_1^i = c_k(q_1^i, \dots, q_{a_k}^i) \text{ et } (q_1^i \ \dots \ q_{a_k}^i \ p_2^i \ \dots \ p_n^i) \#_\omega (N_1 \ \dots \ N_{a_k} \ M_2 \ \dots \ M_n) \end{cases}$$

C'est à dire que, pour un vecteur de valeurs partielles dont la première composante est de la forme $M_1 = c_k(N_1, \dots, N_{a_k})$, les filtrages par $(P), (E)$ et $(P_k), (E_k)[x_1 \backslash M_1]$ sont égaux.

$$\text{match}[(P_k), (E_k)[x_1 \backslash c_k(N_1, \dots, N_{a_k})]](N_1 \ \dots \ N_{a_k} \ M_2 \ \dots \ M_n) = \text{match}[(P), (E)](\vec{M})$$

Soit finalement, l'égalité de correction $\mathcal{C}(\vec{M}, (P), (E)) = \text{match}[(P), (E)](\vec{M})$ est vraie à toutes les étapes d'une compilation qui procède en suivant les directions et pour tout vecteur de termes partiels $\vec{M}$. □

La condition suffisante de correction permet de prouver la correction la compilation `or`$_2$ de la fonction `or`, puisque la compilation `or`$_2$ (figure 6.4) satisfait le critère de correction à chaque appel inductif de type *4*. En particulier, c'est la variable y qui correspond à la direction 2 qui est choisie initialement.

Le critère de correction n'est pas une condition nécessaire de correction d'un automate, comme le montre le filtrage `bizarre` et sa compilation :

<pre>let bizzare = function
  true true → true
| x   _   → x</pre>
	<pre>let BIZ x y = match x with
	  true → (match y with
	             true → true
	           | _ → false)
	| _ → x</pre>

La matrice de motifs définie par la fonction `bizarre` est la même que celle de la fonction `and` et n'a pas de directions. Néanmoins, de par le nouveau membre droit de la deuxième clause, on a `bizarre` Ω `false` $= \Omega$. Dès lors, l'automate `BIZ` peut se permettre d'échouer en tentant de filtrer la composante Ω. Étant donné une définition de combinateur $D = F, (P), (E)$, ce type de phénomène ne peut se produire que quand certains des termes applicatifs du vecteur colonne (E) sont des variables. Il se peut alors qu'il existe des compilations correctes de la fonction de filtrage match$[(P), (E)]$, lorsque le prédicat de filtrage match$[(P)]$ n'est pas compilable correctement et l'algorithme de compilation de la section suivante peut rejeter ces fonctions de filtrage. Heureusement, ce type de fonctions de filtrage est somme toute marginal.

6.5.3 Recherche d'une compilation correcte

Certaines fonctions de filtrage ne sont pas compilables correctement par notre méthode. C'est la cas par exemple de la fonction `and` (figure 6.5). Il existe également des filtrages non-ambigus qui ne sont pas compilables, comme le montre le célèbre exemple du à G. Berry :

```
let G = function
  true  false _     → 1
| false _     true  → 2
| _     true  false → 3
```

Toutefois, le critère de correction du lemme 6.5.3 qui s'applique au moment de la construction des automates de filtrage peut être utilisé pour éliminer des automates incorrects avant de les construire complètement. Par exemple, on se rend compte qu'il n'existe pas d'automate qui réalise correctement les filtrages `and` et `G` dès le premier appel de la fonction de compilation $\mathcal{C}$ qui découvre une matrice (P) pour laquelle on a $\mathrm{Dir}[(P)] = \emptyset$. Cette situation n'est cependant pas satisfaisante, car elle conduit à un algorithme de compilation grossièrement inefficace : si une matrice admet plusieurs directions, la recherche d'un automate correct nécessite *a priori* d'essayer toutes les compilations qui commencent par un filtrage simple en suivant chacune de ces directions. Ce processus peut être extrêmement coûteux. Pour l'éviter, je vais utiliser la théorie des fonctions séquentielles définies sur l'ensemble des termes partiels [Kah 78, Hue 79]. Une première étape est donc de définir le filtrage comme une fonction sur l'ensemble des Ω-termes :

Définition 6.5.4 (Filtrage par une définition) *Soit un combinateur F d'arité n dont la définition est $D = F, (P), (E)$. La fonction de filtrage par D, notée $match_D$, est définie sur l'ensemble des Ω-termes de la façon suivante :*

1. *Pour tout terme partiel M de la forme $F\ M_1\ M_2 \ldots\ M_n$, on pose :*

$$match_D(M) = match[(P), (E)](M_1\ M_2\ \ldots\ M_n)$$

2. *Sinon, on pose $match_D(M) = \Omega$.*

Remarquons que, dans le cas où le terme $M = F\ M_1\ M_2 \ldots\ M_n$ est un radical $\sigma(F\ p_1^i\ p_2^i \ldots\ p_n^i)$ pour la définition D, le terme $\mathrm{match}_D(M)$ est le contractum $\sigma(e^i)$. Le prédicat de filtrage par la définition D s'obtient en remplaçant tous les membres droits de la définition D par la constante `true`, on le note pred_D. Enfin, la correction d'une compilation donnée est exactement l'égalité entre l'exécution $\mathrm{Auto}[F\ x_1\ x_2 \ldots\ x_n, \mathcal{C}(\vec{x}, (P), (E))]$ et la fonction de filtrage match_D.

Les directions des matrices de motifs correspondent maintenant à la notion plus classique d'index d'une fonction monotone sur l'ensemble des termes partiels [Kah 78, Hue 79].

Définition 6.5.5 *Soit $\mathcal{F}$ une fonction monotone sur l'ensemble des termes partiels. Par définition, un Ω-terme M est* raffinable*, quand on a $\mathcal{F}(M) = \Omega$ et qu'il existe un Ω-terme $N \succeq M$, avec $\mathcal{F}(N) \succ \Omega$. Une occurrence u d'un terme raffinable M est un* index *de la fonction $\mathcal{F}$ en M, si et seulement si $M/u = \Omega$ et que, pour tout Ω-terme $N \succeq M$, on a l'implication $\mathcal{F}(N) \succ \Omega \Rightarrow N/u \succ \Omega$. La fonction $\mathcal{F}$ est* séquentielle *en M raffinable,si et seulement si il existe un index de la fonction $\mathcal{F}$ dans M. Enfin la fonction $\mathcal{F}$ est* séquentielle *quand il existe un index dans tous les termes raffinables.*

Les automates définis par imbrication de filtrages simples sont bien des fonctions séquentielles selon cette définition.

Lemme 6.5.6 *Soient un automate A et un terme linéaire M, avec $\mathcal{L}(A) \subseteq \mathcal{V}(M)$. Alors, l'exécution $Auto[M, A]$ est une fonction séquentielle sur l'ensemble des Ω-termes.*

Démonstration : Soit tout d'abord un Ω-terme N_Ω qui n'est pas une instance du préfixe M ($N_\Omega \not\preceq M_\Omega$). Si le terme N_Ω est raffinable, alors il existe un Ω-terme $P_\Omega \succeq N_\Omega$ qui est une instance du préfixe M. Soit, les Ω-termes N_Ω et M_Ω sont compatibles par définition. Dès lors, par la deuxième propriété de la proposition 6.2.5, il existe une occurrence u avec $N_\Omega/u = \Omega$ et $M_\Omega/u \succ \Omega$. Par conséquent, on a $P_\Omega/u \succeq M_\Omega/u \succ \Omega$ et l'occurrence u est un index de la fonction $\mathrm{Auto}[M, A]$ en N_Ω.

On montre ensuite, par induction sur la structure de l'automate A, qu'il existe un index en tout terme raffinable qui est une instance du terme M. Le cas de base de cette induction se divise en trois sous-cas. Si l'automate A est égal au terme indéfini Ω, alors, pour tout terme partiel N_Ω, on a $\mathrm{Auto}[M, A](N_\Omega) = \Omega$. C'est à dire qu'il n'y a pas de terme raffinable et que la fonction $\mathrm{Auto}[M, A]$ est trivialement séquentielle. Si l'automate A est un terme applicatif qui n'est pas une variable, alors, pour toute instance N_Ω du préfixe M, on a $\mathrm{Auto}[M, A](N_\Omega) \succ \Omega$ et la fonction $\mathrm{Auto}[M, A]$ est trivialement séquentielle. Enfin, si l'automate A se réduit à une variable x, soit alors u l'occurrence de la variable x dans le préfixe M. Pour toute instance N du préfixe M, on a l'égalité $\mathrm{Auto}[M, A](N) = N/u$ et on vérifie facilement que l'occurrence u est un index en N dans le cas où N est un terme partiel raffinable. Pour prouver l'étape d'induction on se donne un automate A avec :

$$A = \begin{cases} \texttt{match x with} & \\ \quad \texttt{c}_1\texttt{(x}_1\texttt{,}\ldots\texttt{,x}_{\texttt{a}_1}\texttt{)} & \to \quad A_1 \\ \texttt{|} \quad\quad \vdots & \\ \texttt{|}\ \ \texttt{c}_\texttt{z}\texttt{(x}_1\texttt{,}\ldots\texttt{,x}_{\texttt{a}_\texttt{z}}\texttt{)} & \to \quad A_z \\ \texttt{|}\ \ \texttt{\_} & \to \quad A_d \end{cases}$$

On appelle u l'occurrence de la variable x dans le préfixe M. Soit N_Ω un Ω-terme raffinable qui est une instance du préfixe M. Par définition, on a $\mathrm{Auto}[M, A](N_\Omega) = \Omega$. Si le jugement $[M\backslash N_\Omega] \vdash A \Rightarrow \Omega$ est prouvé directement par un axiome (échec), alors on a l'égalité $\omega(N_\Omega/u) = \Omega$, c'est à dire que l'on a $N_\Omega = \Omega$, $N_\Omega/u = @(N_1, N_2)$ ou $N_\Omega/u = F'$. Or, par hypothèse, il existe un terme partiel $P_\Omega \succeq N_\Omega$, avec $\mathrm{Auto}[M, A](P_\Omega) \succ \Omega$. Il existe donc nécessairement un constructeur c avec $P_\Omega/u = c(P_1, \ldots, P_a)$. Comme on a l'inégalité $P_\Omega/u \succeq N_\Omega/u$, il vient forcément $N_\Omega/u = \Omega$ et u est un index en N_Ω. Si le jugement $[M\backslash N_\Omega] \vdash A \Rightarrow \Omega$ est prouvé à l'aide d'une règle d'inférence —une règle (constr) par exemple—, alors on a l'égalité $N_\Omega/u = c_k(N_1, \ldots, N_{a_k})$ et par conséquent, tout Ω-terme $P_\Omega \succeq N_\Omega$ est une instance du préfixe M qui est telle que l'on a $P_\Omega/u = c_k(P_1, \ldots, P_{a_k})$. On pose ensuite $M_k = M[x\backslash c_k(x_1, \ldots, x_{a_k})]$ et on considère la fonction $\mathrm{Auto}[M_k, A_k]$. Il est clair (corollaire 6.4.5) que l'on a les égalités $\mathrm{Auto}[M, A](N_\Omega) = \mathrm{Auto}[M_k, A_k](N_\Omega)$ et $\mathrm{Auto}[M, A](P_\Omega) = \mathrm{Auto}[M_k, A_k](P_\Omega)$. Dès lors, les index en N_Ω des fonctions $\mathrm{Auto}[M, A]$ et $\mathrm{Auto}[M_k, A_k]$ se confondent et la fonction $\mathrm{Auto}[M, A]$ est séquentielle en N_Ω par application de l'hypothèse d'induction. □

Étant donnée une définition par filtrage D, l'idée est ensuite de montrer que des directions des matrices de motifs qui apparaissent au cours de la compilation du filtrage par D sont des index pour le prédicat de filtrage pred_D. Tout d'abord, il faut préciser les termes raffinables dans lesquels ces index sont susceptibles de se trouver.

Lemme 6.5.7 *Soit une définition par filtrage* $D = F, (P), (E)$ *et un automate* $A = \mathcal{C}(\vec{x}, (P), (E))$ *produit d'une compilation de* D*. Soit un automate* $A' = \mathcal{C}(\vec{x}\,', (P'), (E'))$ *qui est un sous-automate de l'automate* A*. Soit enfin un terme linéaire* M *avec* $\{x_1, x_2, \ldots, x_n\} \subseteq \mathcal{V}(M)$*, où* $(x_1\ x_2\ \ldots\ x_n) = \vec{x}$*. Alors, il existe une substitution* σ *et une transition de la forme* $(M, \sigma, A) \hookrightarrow^* (M', \sigma', A')$

Démonstration : La preuve est par induction sur l'automate A. Si A' n'est pas A lui-même, alors l'automate A est un filtrage simple de la forme

$$A = \texttt{match x with } \texttt{c}_1(\texttt{x}_1,\ldots,\texttt{x}_{\texttt{a}_1}) \;\rightarrow\; A_1 \;\ldots\; \texttt{|}\; \texttt{c}_z(\texttt{x}_1,\ldots,\texttt{x}_{\texttt{a}_z}) \;\rightarrow\; A_z \;\texttt{|}\; \_ \;\rightarrow\; A_d$$

On suppose ensuite que A' est un sous-automate de l'un des A_k et on pose le préfixe augmenté $M_k = M[x\backslash c_k(x_1,\ldots,x_{a_k})]$. Par hypothèse d'induction, il existe une substitution ρ et une transition $(M_k,\rho,A_k) \hookrightarrow^* (M',\rho',A')$. On définit la substitution σ en posant $\sigma(x) = c_k(\rho(x_1),\ldots,\rho(x_{a_k}))$ et $\sigma(y) = \rho(y)$ pour toutes les autres variables du domaine de ρ. On a alors la transition en un pas $(M,\sigma,A) \hookrightarrow (M_k,\sigma_k,A_k)$, avec $\sigma_k = \sigma \cup [x_1\backslash\rho(x_1),\ldots,x_{a_k}\backslash\rho(x_{a_k})]$. Les substitutions ρ et σ_k ne peuvent différer que sur la variable x qui n'est pas utilisable par l'automate A_k. On a donc la transition $(M_k,\sigma_k,A_k) \hookrightarrow^* (M',\sigma',A')$ par le lemme 6.4.6. Soit finalement, il existe une transition de la forme $(M,\sigma,A) \hookrightarrow^* (M',\sigma',A')$. Le cas où l'automate A' est un sous-automate de l'automate par défaut A_d se produit quand l'ensemble Σ des constructeurs reconnus par le filtrage simple A ne forme pas une signature complète. Il existe alors au moins un constructeur c qui n'appartient pas à Σ et on procède comme dans le cas ci-dessus en posant $M_d = M[x\backslash c(x_1,\ldots,x_a)]$. □

On explicite ensuite le lien entre le prédicat de filtrage selon une définition de combinateur et les matrices de motifs qui surgissent au cours de la compilation du filtrage selon cette définition.

Lemme 6.5.8 *Soient une définition par filtrage $D = F,(P),E$ et une compilation quelconque de cette définition. Alors, pour toute transition de la forme :*

$$(F\ x_1\ x_2\ldots\ x_n, [x_1\backslash M_1,\, x_2\backslash M_2,\ldots,x_n\backslash M_n], \mathcal{C}(\vec{x},(P),(E))) \hookrightarrow^* (N',\sigma',\mathcal{C}(\vec{x}\,',(P'),(E')))$$

On a l'égalité :

$$pred_D(F\ M_1\ M_2\ldots\ M_n) = match[(P')](\sigma'(x'_1)\ \sigma'(x'_2)\ \ldots\ \sigma'(x'_n))$$

Démonstration : La preuve est par induction sur la longueur de la transition en jeu. Si cette transition est vide, alors il n'y a rien à prouver. Sinon, on pose $A = \mathcal{C}(\vec{x},(P),(E))$, $A' = \mathcal{C}(\vec{x}\,',(P'),(E'))$, $N = F\ x_1\ x_2\ldots\ x_n$, $\sigma = [x_1\backslash M_1,\, x_2\backslash M_2,\ldots,x_n\backslash M_n]$ et on opère la décomposition de la transition $(N,\sigma,A) \hookrightarrow^* (N',\sigma',A')$ en la transition $(N,\sigma,A) \hookrightarrow^* (N'',\sigma'',A'')$ et le pas $(N'',\sigma'',A'') \hookrightarrow (N',\sigma',A')$. L'automate A'' est nécessairement un filtrage simple et on suppose que c'est la variable x''_1 qui est examinée à cette étape. On suppose aussi —par exemple– que que l'on a $(P') = (P''_d)$. Le terme $\sigma''(x''_1)$ est donc ici de la forme $\sigma''(x''_1) = c(M''_1,\ldots,M''_a)$, où le constructeur c n'est pas reconnu par le filtrage simple A''. Par hypothèse d'induction, on a l'égalité

$$\mathrm{pred}_D(F\ M_1\ M_2\ldots\ M_n) = \mathrm{match}[(P'')](c(M''_1,\ldots,M''_a)\ \sigma''(x''_2)\ \ldots\ \sigma''(x''_{n''}))$$

Par ailleurs, comme il a déjà été montré dans la preuve du lemme 6.5.3 dans le cas plus général de la fonction de filtrage, on a l'égalité :

$$\mathrm{match}[(P'')](c(M''_1,\ldots,M''_a)\ \sigma''(x''_2)\ \ldots\ \sigma''(x''_{n''})) = \mathrm{match}[(P''_d)](\sigma''(x''_2)\ \ldots\ \sigma''(x''_{n''}))$$

Soit ici, compte tenu des égalités $x'_1 = x''_2, \ldots x'_{n'} = x''_{n''}$, $\sigma' = \sigma''$ et $(P') = (P''_d)$:

$$\mathrm{pred}_D(F\ M_1\ M_2\ldots\ M_n) = \mathrm{match}[(P')](\sigma'(x'_1)\ \ldots\ \sigma'(x'_{n'}))$$

□

Enfin, étant donnée une définition par filtrage D, on montre que les index du prédicat pred_D en certains termes et les directions des matrices de motifs qui apparaissent au cours de la compilation du filtrage selon D se correspondent exactement.

Lemme 6.5.9 *Soit une définition de combinateur $D = F, (P), (E)$. Et soit une étape non-terminale de la compilation du filtrage selon D, caractérisée par un triplet $\vec{x}\,', (P'), (E')$. Alors, il existe un terme $M' = F\ M'_1\ M'_2 \ldots\ M'_n$ qui est raffinable et tel que les index du prédicat de filtrage par D en M'_Ω sont exactement les occurrences dans M' des variables x_i, qui sont telles que l'indice de colonne i est une direction de la matrice de motifs (P').*

Démonstration: On pose $M = F\ x_1\ x_2 \ldots\ x_n$. Par les lemme 6.5.7 et 6.4.7, il existe une transition de la forme $(M, \sigma, A) \hookrightarrow^* (M', \sigma', A')$, avec $\sigma(M) = M'$. Par l'égalité $\sigma'(M') = M'$ et pour toute variable x' du terme M', on a l'égalité $\sigma'(x') = x'$. Par le lemme 6.5.8, on a ici $\mathrm{pred}_D(M') = \mathrm{match}[(P')](\vec{x}\,')$, ou encore $\mathrm{pred}_D(M'_\Omega) = \mathrm{match}[(P')](\vec{\Omega})$. Or, par hypothèse, le vecteur $\vec{\Omega}$ n'est pas filtré par la matrice (P') (sinon la compilation serait terminée). Autrement dit, on a $\mathrm{pred}_D(M'_\Omega) = \Omega$. On note $\vec{p}\,' = (p'_1\ p'_2\ \ldots\ p'_{n'})$, la première ligne de motifs de la matrice (P'). Par définition du filtrage avec ou sans priorité, le vecteur $\vec{p}\,'$ est filtré par la matrice (P'). En posant $N = M'[x'_1 \backslash p'_1, \ldots, x'_{n'} \backslash p'_{n'}]$ et par le lemme 6.4.6, il vient $\mathrm{pred}_D(N) = \mathrm{match}[(P')](\vec{p}\,') = \mathtt{true}$, soit encore $\mathrm{pred}_D(N_\Omega) = \mathtt{true}$, avec $N_\Omega \succeq M'_\Omega$. Autrement dit, le terme M'_Ω est raffinable pour la fonction pred_D. Soit ensuite une occurrence u dans le terme M'_Ω, avec $M'_\Omega/u = \Omega$. On distingue deux cas :

1. Si u est l'occurrence d'une variable x'_i dans le terme M', alors, tout terme partiel $R \succeq M'_\Omega$ peut s'écrire sous la forme $R = \rho(M')$ par linéarité de M'. Ensuite, par l'égalité $\mathrm{pred}_D(R) = \mathrm{match}[(P')](\rho(x'_1)\ \ldots\ \rho(x'_{n'}))$, il vient facilement que l'occurrence u est un index pour le prédicat de filtrage pred_D en M'_Ω, si et seulement si l'indice i est une direction de la matrice (P').
2. Si le sous-terme M'/u est une variable qui n'est pas l'un des x'_i, alors il existe le terme partiel N_Ω du paragraphe précédent qui est filtré par la définition D et tel que $N_\Omega/u = \Omega$. Par conséquent l'occurrence u n'est pas un index pour la fonction pred_D en M'_Ω. □

Corollaire 6.5.10 *Soit une définition de combinateur D. Alors, il existe une compilation correcte du prédicat de filtrage pred_D, si et seulement si ce prédicat est une fonction séquentielle.*

Démonstration : Il s'agit d'un corollaire des deux lemmes précédents. Le sens "si" est une conséquence directe du lemme 6.5.9 ; tandis que le sens "et seulement si" vient du fait que, si il existe une compilation correcte de pred_D, alors ce prédicat est égal à une exécution d'automate qui est une fonction séquentielle (lemme 6.5.6). □

La séquentialité du prédicat de filtrage associé à une définition de combinateur est donc décidable. En outre, nous disposons maintenant d'un algorithme correct de compilation du filtrage qui procède sans retour en arrière. Cet algorithme est complet pour la classe des définitions par filtrage qui sont telles que le prédicat pred_D est séquentiel.

Proposition 6.5.11 *Soit une définition de combinateur D. Chaque fois que le prédicat de filtrage pred_D est séquentiel, l'algorithme de compilation suivant produit un automate qui met en œuvre correctement la fonction de filtrage match_D.*

Compiler le filtrage comme décrit dans la section 6.4.3. À chaque étape inductive $\mathcal{C}(\vec{x}, (P), (E))$ qui peut se résoudre par le cas 4 de la définition de la fonction de compilation $\mathcal{C}$, deux cas sont possibles :

- *Si la matrice (P) n'a pas de direction, alors échouer.*
- *Si la matrice (P) a des directions, alors en choisir une et poursuivre la compilation en suivant cette direction.*

En fait, le compilateur GAML ne refuse pas de traiter le cas des prédicats de filtrages non-séquentiels mais les compile de façon incorrecte en poursuivant la compilation en suivant un indice de la matrice (P) qui n'est pas une direction. Le choix de cet indice n'est pas arbitraire et est décrit à la fin de la section suivante. En outre, même au cours d'une compilation incorrecte, peuvent apparaître des matrices de motifs qui ont des directions qui donneront lieu à des filtrages simples "localement corrects".

6.5.4 Calcul des directions

Le calcul des directions dans le cas général se ramène au problème de l'existence d'au moins un vecteur de valeurs filtré par la dernière ligne de certaines matrices de motifs. Étant donnés une matrice de motifs (P), un numéro de ligne i et un indice de colonne d dans (P). On définit la matrice $(Q_{(d,i)})$ comme la matrice qui se construit à partir de (P), en effaçant la d-ième colonne et toutes les lignes après la i-ème ligne :

$$(Q_{(d,i)}) = \begin{pmatrix} p_1^1 \cdots p_{d-1}^1 & p_{d+1}^1 \cdots p_n^1 \\ p_1^2 \cdots p_{d-1}^2 & p_{d+1}^2 \cdots p_n^2 \\ & \vdots \\ p_1^i \cdots p_{d-1}^i & p_{d+1}^i \cdots p_n^i \end{pmatrix}$$

Lemme 6.5.12 *Soit une matrice de motifs (P) à n colonnes et m lignes. Soit un indice de colonne d et un numéro de ligne i de la matrice (P). Soit la matrice $Q_{(d,i)}$ telle que décrite ci-dessus. Alors, on a l'équivalence.*

$$d \notin Dir_i[(P)] \Leftrightarrow (p_d^i = x \ \ et \ \exists \vec{V},\ \ match_i[(Q_{(d,i)})](\vec{V}))$$

Démonstration : Tout vecteur de valeurs partielles $\vec{V}$ qui est filtré par la i-ème ligne de la matrice (P) est une instance du vecteur de motifs $\vec{p}^i$. Par conséquent, si le motif p_d^i n'est pas une variable (i.e., $p_d^i = c(q_1, \ldots, q_a)$), alors on a $c(q_1, \ldots, q_a) \preceq V_d$ et donc, $V_d \succ \Omega$. Si le motif p_d^i est une variable x, alors, quelles que soient les $n-1$ valeurs partielles $V_1, \ldots, V_{d-1}, V_{d+1}, \ldots, V_n$, on montre facilement l'équivalence suivante en revenant à la définition du filtrage :

$$\mathrm{match}_i[(P)](V_1 \ \ldots \ V_{d-1} \ \Omega \ V_{d+1} \ \ldots \ V_n) \Leftrightarrow \mathrm{match}_i[(Q_{(d,i)})](V_1 \ \ldots \ V_{d-1} \ V_{d+1} \ \ldots \ V_n)$$

$\square$

On a donc ramené le problème de savoir si un indice n'est pas une direction au problème connu sous le nom de détection *des cas inutiles* du filtrage ("*unused match cases*" en anglais).

Étant donnée une matrice de motifs (P) à n colonnes et m lignes ($m \geq 1$), l'algorithme suivant calcule la valeur de vérité de la formule $\mathcal{F}(P) = \exists \vec{V} \ \mathrm{match}_m[(P)](\vec{V})$. Cet algorithme suit de très près l'algorithme de compilation C.

1. Si les lignes de la matrice (P) sont vides ($n = 0$) ou si la première ligne n'est faite que de variables, alors tous les vecteurs de valeurs de taille n sont filtrés par la première ligne de la matrice (P). C'est à dire que le prédicat de filtrage $\mathrm{match}_1[(P)]$ est vérifié par tous les vecteurs de taille n. Le prédicats de filtrage selon des lignes de motifs différentes étant mutuellement exclusifs, la valeur de $\mathcal{F}(P)$ dépend du nombre de lignes m de la matrice (P). Soit, $\mathcal{F}(P) = \texttt{true}$ si $m = 1$, $\mathcal{F}(P) = \texttt{false}$ sinon. Si la matrice (P) est tout à fait vide ($m = 0$), alors il vient aussi $\mathcal{F}(P) = \texttt{false}$ en revenant aux définitions.

2. Dans tous les autres cas, on choisit une colonne de la matrice de motifs (P). Supposons, par exemple, que la première colonne soit choisie. Soit alors $\Sigma = \{ c_k \mid 1 \leq k \leq z \}$ l'ensemble des constructeurs de tête des motifs de la première colonne. À chaque constructeur c_k de Σ, on associe la matrice de motifs (P_k) de l'étape *4* de la compilation du filtrage (section 6.4.3). Si l'ensemble Σ ne forme pas une signature complète, ou si l'ensemble Σ est vide, alors on considère aussi la matrice par défaut (P_d). On distingue ensuite deux cas :
   (a) Si $p_1^m = x$, alors soit un vecteur $\vec{V}$ qui est filtré par la dernière ligne de (P). Si la première composante V_1 de $\vec{V}$ est de la forme $V = c_k(U_1, \ldots, U_{a_k})$, avec $c_k \in \Sigma$, alors on montre facilement l'égalité $\mathcal{F}(P) = \mathcal{F}(P_k)$ en revenant aux définitions comme dans la preuve du lemme 6.5.3. De même, si V_1 est de la forme $c(U_1, \ldots, U_a)$, avec $c \notin \Sigma$, on a l'égalité $\mathcal{F}(P) = \mathcal{F}(P_d)$. Enfin si $V_1 = \Omega$, alors, par monotonie du prédicat de filtrage, tout vecteur $\vec{V}' = (c'(\Omega_1, \ldots, \Omega_{a'})\ V_2\ \ldots\ V_n)$, où c' est un constructeur quelconque de la signature appropriée, est filtré par la dernière ligne de la matrice (P) et on est ramené à l'un des cas précédents. Soit finalement, la formule $\mathcal{F}(P)$ est vraie, si et seulement si l'une au moins des formules $\mathcal{F}(P_1)$, $\mathcal{F}(P_2)$, $\ldots \mathcal{F}(P_z)$ ou $\mathcal{F}(P_d)$ l'est.
   (b) Sinon, le motif p_1^m est de la forme $c_k(q_1, \ldots, q_{a_k})$ ou c_k est un constructeur de Σ et on a l'égalité $\mathcal{F}(P) = \mathcal{F}(P_k)$.

Si on considère l'efficacité de cet algorithme, on voit facilement que le nombre d'appels inductifs de la fonction $\mathcal{F}$ est borné par le nombre d'appels de la fonction $\mathcal{C}$ quand la compilation est effectuée en faisant les mêmes choix aux étapes critiques. Cette borne pouvant être atteinte quand $\mathcal{F}(P)$ est fausse et quand la dernière ligne de la matrice (P) n'est composée que de variables. Comme le montre un exemple de compilation de filtrage donné dans [Sek 92, Mar 92] cette borne supérieure peut être exponentielle en la somme des tailles des motifs de la matrice (P). Bien que je ne sache pas si cette borne supérieure est effectivement atteinte, l'expérience montre qu'une réalisation naïve de la fonction $\mathcal{F}(P)$ conduit à des calculs énormes. Et, en effet, il est montré dans [Sek 92] que le problème de savoir si un indice de colonne n'est pas une direction est NP-complet. Le compilateur GAML utilise des heuristiques qui se sont révélées satisfaisantes :

1. On peut réduire la taille de la matrice (P) elle même. Soient deux lignes de motifs $\vec{p}^i$ et $\vec{p}^j$ et qui ne sont pas la dernière ligne de la matrice (P). Si on a $\vec{p}^i \preceq \vec{p}^j$, alors tout vecteur de valeurs $\vec{V}$ qui est incompatible avec le vecteur $\vec{p}^i$ est aussi incompatible avec $\vec{p}^j$, par définition même de l'incompatibilité. Autrement dit, $\mathcal{F}(P)$ est égal à $\mathcal{F}(P')$ où (P') est la matrice (P) avec la j-ème ligne effacée. La matrice (P) peut donc être simplifiée en ne retenant que les lignes de motifs qui sont minimales pour l'ordre $\preceq$ sur les vecteurs de motifs. Cette simplification est particulièrement intéressante quand la matrice (P) contient beaucoup de variables. Dans la pratique, il suffit de l'appliquer à la matrice $Q_{(d,i)}$ initiale.
2. Quand il existe une matrice par défaut (P_d) dans le cas inductif *2-(a)* du calcul de $\mathcal{F}(P)$, alors cette matrice est testée la première. Ce choix repose sur l'observation que les êtres humains ont tendance à introduire des motifs "attrape-tout" et des variables dans les derniers motifs des définitions par filtrage, justement pour filtrer les valeurs dont les constructeurs n'apparaissent pas dans les motifs plus prioritaires.

Quant à l'efficacité du calcul des ensembles de directions $\mathrm{Dir}[(P)]$, il n'est généralement pas utile de calculer tous les ensembles $\mathrm{Dir}_i[(P)]$ complètement. Tout d'abord, si il existe une colonne de la matrice (P) qui ne contient pas de variable, alors, par le lemme 6.5.12, l'indice de cette colonne est une direction. Une telle direction est une *direction évidente*. L'algorithme de compilation de la proposition 6.5.11 qui n'a besoin que d'une seule direction pour pouvoir progresser peut se contenter d'une direction évidente. S'il n'existe pas de direction évidente dans la matrice (P), alors on calcule l'ensemble $\mathrm{Dir}[(P)]$. Si, pour un numéro de ligne i donné, l'indice d n'est pas élément de l'ensemble $\mathrm{Dir}_i[(P)]$, alors il est inutile de chercher à savoir si l'indice d appartient à l'ensemble $\mathrm{Dir}_j[(P)]$ pour tout autre numéro de ligne j, car on sait déjà que d n'est pas une direction de la matrice (P). Bien entendu, les conditions $d \in \mathrm{Dir}_i[(P)]$ sont vérifiées selon l'ordre croissant des numéros de lignes, afin d'éviter les calculs de préférence quand les matrices $(Q_{(d,i)})$ sont les plus grosses. C'est à dire que l'on calcule la suite :

$$\mathcal{D}_1 = \mathrm{Dir}_1[(P)] \qquad \mathcal{D}_{i+1} = \{\, d \in \mathcal{D}_i \mid d \in \mathrm{Dir}_{i+1}[(P)] \,\}$$

Ce calcul s'arrête quand on a $i = m$ (auquel cas, on a $\mathcal{D}_m = \mathrm{Dir}[(P)]$) ou dès que l'ensemble $\mathcal{D}_i$ est vide (auquel cas, la matrice (P) n'a pas de direction). Dans ce dernier cas, il existe un numéro de ligne max qui est tel que $\mathcal{D}_{max} \neq \emptyset$ et $\mathcal{D}_{max+1} = \emptyset$ Les indices de colonnes éléments de $\mathcal{D}_{max}$ sont des *directions partielles* qui sont utilisées en place des directions par le compilateur GAML pour compiler (incorrectement) les filtrages non-séquentiels.

6.6 Évaluation correcte et optimale des programmes GAML

On se place dans le cadre général des systèmes orthogonaux et on reprend les concepts et notations de [Hue 79, Part II] (voir aussi [Klo 89]). Étant donné un système de réécriture orthogonal Σ, on note red l'ensemble des membres gauches des règles de Σ et red_Ω l'ensemble des schémas de radicaux $red_\Omega = \{\, \alpha_\Omega \mid \alpha \in red \,\}$.

Si un terme partiel M distinct de Ω est un préfixe de nom de radical (i.e., $\exists \alpha_\Omega \in red_\Omega,\ M \preceq \alpha_\Omega$), sans être un schéma de radical (i.e., $M \notin red_\Omega$), alors on note $M \in red_\prec$ et on dit que M est un *préfixe strict* (de nom de radical). Dans le cas d'un système de combinateurs avec filtrage, un préfixe strict M est de la forme $M = \Omega\ M_1\ M_2 \ldots\ M_n$, où l'entier n est plus petit que la plus grande des arités des combinateurs du programme $\mathcal{P}$; ou bien il est de la forme $M = F\ M_1\ M_2 \ldots\ M_a$, où a est l'arité du combinateur F.

Un terme partiel M est *proprement compatible* (avec un nom de radical) et on note $M{\uparrow_+}$, quand il est distinct de Ω et compatible avec au moins un membre gauche. Soit encore, pour tout terme partiel M, on pose :

$$M{\uparrow_+} \Leftrightarrow \Omega \prec M \text{ et } \exists \alpha_\Omega \in red_\Omega,\ M \uparrow \alpha_\Omega$$

Par exemple, tous les préfixes stricts sont proprement compatibles.

Un terme *imprévu* est un terme M qui ne peut plus devenir un radical. Ici, les seuls termes imprévus utilisés sont les termes de la forme $M = F\ M_1\ M_2 \ldots\ M_n$ où n est l'arité du combinateur F, qui sont tels que, pour tous les noms de radicaux de la forme $\alpha = F\ p_1\ p_2 \ldots\ p_n$, on a l'incompatibilité $\vec{p} \,\#_\omega\, \vec{M}$. Par définition de la relation $\#_\omega$, ces termes imprévus admettent un *préfixe imprévu*, c'est à dire qu'il existe un terme de la forme $P = F\ q_1\ q_2 \ldots\ q_n$, où $\vec{q}$ est un vecteur de motifs, avec $P \preceq M$ et P est lui même un terme imprévu. Si besoin est, l'existence de préfixes imprévus minimaux découle du lemme 6.3.5, qui s'applique puisque l'ensemble des préfixes imprévus est clairement un sous-ensemble monotone de valeurs partielles.

On note "match" le prédicat de filtrage défini par le système Σ. Le système Σ étant orthogonal, le prédicat match est un prédicat de filtrage non-ambigu :

$$\mathrm{match}(M) \Leftrightarrow \exists \alpha_\Omega \in red_\Omega,\ \alpha_\Omega \preceq M$$

Pour tout terme partiel M raffinable, on note $\mathrm{Dir}(M)$ l'ensemble des index du prédicat de filtrage match dans le terme M qui sont les *directions* dans le terme M.

On suppose l'existence d'une fonction Q qui associe un sous-ensemble non-vide de $\mathrm{Dir}(M)$ de *directions fortes* à tout préfixe strict M. La propriété fondamentale des directions fortes, telle que reprise du travail de Huet et Lévy, s'énonce comme suit : pour tout préfixe strict M, toute occurrence u dans $Q(M)$, tout préfixe strict N, tel que le terme partiel $M[u \leftarrow N]$ est encore un préfixe strict, on a les deux conditions de transitivité :

$$Q_1\colon\ \exists v \in \mathcal{O}(N),\ u.v \in Q(M[u \leftarrow N])$$
et
$$Q_2\colon\ \forall v \in \mathcal{O}(N),\ u.v \in Q(M[u \leftarrow N]) \Rightarrow v \in Q(N)$$

Selon Huet et Lévy, le système Σ est alors *fortement séquentiel* et il est possible de calculer certains radicaux standards.

Lemme 6.6.1 (Huet-Lévy) *Soient un terme M et un sous-terme $N = M/u$. On suppose qu'il existe une suite $(M_i)_{1\leq i\leq n}$ de sous-termes du terme M, qui est telle que $M_1 = M$, $M_n = N$ et que pour tout entier i de l'intervalle $[1 \ldots n[$, il existe un préfixe strict P_i, et une occurrence v_i dans $Q(P_i)$, avec :*

$$P_i \preceq M_i \qquad\qquad M_{i+1} = M_i/v_i \qquad\qquad \forall w \in \mathcal{O}(P_i),\ M/w{\uparrow_+} \Rightarrow w \prec v_i$$

Alors, on a les deux résultats suivants :

1. *Si le sous-terme N est un radical, alors ce radical est standard à partir du terme M.*
2. *Si le sous-terme N est imprévu, soit alors le préfixe imprévu associé $P_n = F\ q_1\ q_2 \ldots\ q_m$. Dès lors, le terme partiel $P = P_1[v_1 \leftarrow P_2[v_2 \leftarrow \ldots P_{n-1}[v_{n-1} \leftarrow P_n]]]$ est un préfixe préservé par toutes des dérivations issues de M.*

Intuitivement, chaque sous-terme M_i, doit, pour devenir un radical, voir se transformer son sous-terme situé à l'occurrence v_i. Par la condition sur les sous-termes proprement compatibles du préfixe strict P_i de M_i, cette évolution est également nécessaire, quand, en définitive, c'est un sous-terme strict de M_i dont l'occurrence aboutit à l'intérieur de P_i qui devient un radical. Enfin, si le dernier des sous-termes M_i ne peut pas devenir un radical (c'est à dire, s'il est imprévu), alors c'est toute la chaîne des P_i qui est bloquée.

La présentation de Huet et Lévy donne directement le résultat *1* puisqu'un tel radical est susceptible d'être trouvé par leur procédure de recherche d'un radical standard [Hue 79, lemme 5.10]. Le résultat *2* n'est pas mentionné explicitement, mais il se démontre facilement en utilisant les Ω-réductions introduites par ces auteurs (voir par exemple la remarque qui suit la définition 4.9 et le lemme 4.14 dans [Hue 79]).

Lemme 6.6.2 *Si toutes les définitions de combinateurs D d'un programme $\mathcal{P}$, sont telles que le prédicat $pred_D$ est séquentiel, alors le système équivalent $\Sigma_{\mathcal{P}}$ est orthogonal et fortement séquentiel.*

Démonstration : D'après les lemmes 6.2.9 et 6.3.10, il suffit de vérifier que pour toute définition $D = F, (P), (E)$ et tout numéro de ligne i de la matrice de motifs (P), les valeurs étendues principales relativement à la i-ème ligne de la matrice (P) sont deux à deux incompatibles. Ce qui est le cas, puisque ([Cur 86, proposition 2.4.2]) les point minimaux d'un prédicat séquentiel sont deux à deux incompatibles. L'esprit d'une preuve directe —donnée dans [Lav 88, proposition 4.3.1]— est de montrer que si $\vec{U}$ et $\vec{V}$ sont deux valeurs étendues principales distinctes et compatibles, alors le prédicat de filtrage n'est pas séquentiel en le terme $F\ (U_1 \sqcap V_1)\ (U_2 \sqcap V_2) \ldots\ (U_n \sqcap V_n)$, où $\sqcap$ est l'opération borne inférieure pour l'ordre préfixe $\preceq$ sur l'ensemble des valeurs partielles.

```
type terme = Var of var | Constr of constr & terme list
| App of term & terme | Comb of comb
;;

type motif =  Simple of constr & var list
;;

type automate =
  Omega
| Succes of terme
| Filtre of var & ((motif & automate) list & automate)
;;
```

Figure 6.6 : Le type des automates en CAML

On montre ensuite que le système $\Sigma_{\mathcal{P}}$ est fortement séquentiel. Les membres gauches des règles du système $\Sigma_{\mathcal{P}}$ sont tous de la forme $\alpha = F\ q_1\ q_2 \ldots\ q_n$, où n est l'arité du combinateur F et $\vec{q}$ est un motif étendu principal relativement à la i-ème ligne de la matrice de motifs (P) de la définition $D = F, (P), (E)$ du programme $\mathcal{P}$. Soit maintenant un préfixe strict M. Dans le cas ou ce préfixe est de la forme générale $M = \Omega\ M_1\ M_2 \ldots\ M_{n'}$, soit $1_{n'} = 1.1 \ldots 1$, l'occurrence composée de l'entier 1 répété n' fois. Soit ensuite un terme $N \succeq M$ tel que match$(M) =$ `true`, alors le terme N est nécessairement de la forme $N = F\ M_1\ M_2 \ldots\ M_n$, où F est un combinateur dont l'arité n est supérieure ou égale a n'. On a alors $N/1_{n'} \succ \Omega$ et l'occurrence $1_{n'}$ est une direction dans le terme M. On pose donc $Q(M) = 1_{n'}$. Dans le cas où le préfixe M est de la forme $M = F\ M_1\ M_2 \ldots\ M_n$, alors pour tout terme partiel $N \succeq M$ qui est nécessairement de la forme $N = F\ M_1\ M_2 \ldots\ M_n$ on a l'égalité match$(M) = \text{pred}_D(M)$ par définition des valeurs étendues principales. Par conséquent, les index du prédicat de filtrage pred_D en M sont exactement les directions dans le terme M et on pose $Q(M)$ égal à l'ensemble de ces index. Il reste à vérifier que la fonction Q satisfait les conditions Q_1 et Q_2. Soient tout d'abord un préfixe strict M de la forme $M = F\ M_1\ M_2 \ldots\ M_a$ et une occurrence u prise dans $Q(M)$. On voit facilement qu'il n'existe pas de préfixe strict $N = @(N_1, N_2)$ ou $N = F'$, qui est tel que le terme partiel $M[u \leftarrow N]$ est un préfixe strict, puisque, pour tout préfixe strict $P \succeq M$, le symbole de tête du sous-terme P/u est nécessairement un constructeur, si il est strictement mieux défini que Ω. L'ensemble $Q(M)$ vérifie donc ici trivialement les conditions de transitivité. Soit enfin un préfixe strict de la forme $M = \Omega\ M_1\ M_2 \ldots\ M_{n'}$ et un préfixe strict N qui est tel que $P = M[1_{n'} \leftarrow N]$ est un préfixe strict. Dans le cas $N = \Omega\ M_1\ M_2 \ldots\ M_{n''}$ on a bien $1_{n'}.Q(N) = Q(P) = 1_{n'+n''}$. Le cas $N = F\ M_1\ M_2 \ldots\ M_n$ qui entraîne $n' = 0$ et donc $M = \Omega$ est impossible, car le terme indéfini n'est pas un préfixe strict. □

On va maintenant préciser un algorithme qui, étant donné un terme M, calcule l'occurrence d'un radical standard à partir du terme M. Comme on l'a déjà fait dans le cas de la standardisation cet algorithme est décrit par une fonction CAML.

On suppose connus les types `var` des noms de variables, `constr` des noms de constructeurs et `comb` des noms de combinateurs. Les déclarations CAML de la figure 6.6 définissent le type des termes applicatifs, le type des motifs simples et le type des automates, elles renvoient aux définitions des grammaires correspondantes (sections 6.1 et 6.4.1). Le constructeur `Filtre` de la signature du type des automates rend compte des filtrages simples, son argument est une paire dont la première composante est la variable à examiner et la seconde composante est elle même une paire composée de la liste (non-vide) des clauses ordinaires et d'une clause par défaut. Le motif "_" de la clause par défaut est sous-entendu.

Les substitutions sont des fonctions des termes dans les termes. On fait abstraction de leur mise en œuvre en se contentant d'une abréviation de type.

```
type substitution == terme → terme
```

En outre, on conserve toutes les notations déjà introduites pour les termes et les substitutions. Par exemple, un motif noté $c_k(x_1, x_2, \ldots, x_{a_k})$ est en fait l'objet CAML qui s'écrit normalement comme `Simple (c_k, [x_1 ; x_2 ;...; x_{a_k}])`.

```
let rec trouve_clause c (cls,A_d) = match cls with
  []                        →  nouvelles_variables (constr_arite c) A_d
| (c_k(x_1,x_2,...,x_{a_k}),A_k)::cls →
  if c=c_k then ((x_1,x_2,...,x_{a_k}),A_k) else trouve_clause c (cls,A_d)
;;

type reponse = Stop | Oui of terme & substitution & automate
;;

let pas (M,σ,A) = match A with
  Filtre (x,cls)  →  (match σ x with
      c(M_1,M_2,...,M_a)  →
        let (x_1,x_2,...,x_a),A' = trouve_clause c cls in
        Oui (M[x\c(x_1,x_2,...,x_a)],σ ∪ [x_1\M_1, x_2\M_2,...,x_a\M_a],A')
    | _   →  Stop)
| _  →  Stop
;;

let rec termine (M,σ,A) = match pas (M,σ,A) with
  Oui (M',σ',A') →  termine (M',σ',A')
| Stop           →  (M,σ,A)
;;
```

Figure 6.7 : Transitions des automates

La figure 6.7 explicite le calcul des transitions. La fonction `trouve_clause` est une fonction auxiliaire qui recherche le sous-automate à associer à une terme dont le symbole de tête est un constructeur `c`. La fonction `trouve_clause` utilise les appels prédéfinis `constr_arite c` qui associe son arité à tout constructeur c et `nouvelle_variables a` qui associe une liste de a variables nouvelles à tout entier naturel a. L'appel de fonction `pas (M,σ,A)` réalise les transitions en un pas proprement dites. Si la configuration donnée en argument à la fonction `pas` est une configuration terminale, alors la fonction `pas` renvoie la réponse `Stop`, sinon elle renvoie un triplet encapsulé `Oui (M',σ',A')` qui est la nouvelle configuration, avec $(M, \sigma, A) \hookrightarrow (M', \sigma', A')$. Enfin la fonction `termine` calcule l'unique configuration terminale que l'on peut atteindre à partir d'une configuration initiale donnée en argument.

Le lemme suivant explique le sens des membres droits d'automates réduits au terme indéfini Ω : lorsque l'exécution d'un automate termine sur un tel état, c'est qu'un terme imprévu a été reconnu.

Lemme 6.6.3 *Soit un triplet $\vec{x}, (P), (E)$ qui est tel que l'automate $A = \mathcal{C}(\vec{x}, (P), (E))$ a un sens. Et soit une transition de la forme $(M, \sigma, A) \hookrightarrow^* (M', \sigma', \Omega)$. Alors, quelle que soit la ligne de motifs $(p_1^i\ p_2^i\ \dots\ p_n^i)$ prise dans la matrice (P), on a l'incompatibilité*

$$(p_1^i\ p_2^i\ \dots\ p_n^i)\ \#_\omega\ (\sigma(x_1)\ \sigma(x_2)\ \dots\ \sigma(x_n))$$

Démonstration : On procède par induction sur la transition $(M, \sigma, A) \hookrightarrow^* (M', \sigma', \Omega)$. Si cette transition est vide, alors on a $A = \Omega$. La seule manière possible d'obtenir le terme indéfini Ω par compilation est le cas *1* de la définition de la fonction $\mathcal{C}$ de la section 6.4.3, La matrice (P) est donc nécessairement vide, de sorte que la condition demandée est trivialement vérifiée. Sinon, l'automate A est un filtrage simple d'une des variables x_i (on pose $x_i = x_1$ sans perte de généralité), avec

$$A = \texttt{match x}_1\ \texttt{with c}_1\texttt{(y}_1\texttt{,}\dots\texttt{,y}_{\texttt{a}_1}\texttt{)}\ \to\ A_1\ \dots\ \texttt{| c}_\texttt{z}\texttt{(y}_1\texttt{,}\dots\texttt{,y}_{\texttt{a}_\texttt{z}}\texttt{)}\ \to\ A_z\ \texttt{|}\ \_\ \to\ A_d$$

On procède ensuite à la décomposition $(M, \sigma, A) \hookrightarrow (M'', \sigma'', A'')$ et $(M'', \sigma'', A'') \hookrightarrow^* (M', \sigma', \Omega)$. On suppose —par exemple— que le terme $\sigma(x_1)$ est de la forme $c_k(N_1, \dots, N_{a_k})$, où le constructeur c_k est reconnu par la k-ième clause ordinaire de A. On a donc $A'' = A_k = \mathcal{C}(\vec{x}'', (P_k), (E_k))$. Soit ensuite $\vec{p}^i = (p_1^i\ p_2^i\ \dots\ p_n^i)$ la i-ème ligne de motifs de la matrice (P). Si le motif p_1^i possède un constructeur de tête $c_{k'} \neq c_k$, alors on a directement $p_1^i\ \#_\omega\ \sigma(x_1)$. Si le motif p_1^i est de la forme $c_k(q_1, \dots, q_{a_k})$, alors il existe une ligne de motifs de la forme $\vec{p}'' = (q_1\ \dots\ q_{a_k}\ p_2^i\ \dots\ p_n^i)$ dans la matrice (P_k) et, par induction on a la relation $\vec{p}''\ \#_\omega\ \sigma''(\vec{x''})$. Or, quels que soient les termes N_1,… N_{a_k} et M_2, … M_n et par définition de l'incompatibilité sémantique $\#_\omega$, on a l'équivalence :

$$\begin{array}{ccc} (c_k(q_1, \dots, q_{a_k})\ p_2^i\ \dots\ p_n^i) & \#_\omega & (c_k(N_1, \dots, N_{a_k})\ M_2\ \dots\ M_n) \\ & \Updownarrow & \\ (q_1\ \dots\ q_{a_k}\ p_2^i\ \dots\ p_n^i) & \#_\omega & (N_1\ \dots\ N_{a_k}\ M_2\ \dots\ M_n) \end{array}$$

D'où il vient finalement $\vec{p}^i\ \#_\omega\ \sigma(\vec{x})$, en posant $M_j = \sigma(x_j) = \sigma''(x''_{j+a_k-1})$, pour tout entier j, tel que $2 \leq j \leq n$. Enfin, si le motif p_1^i est une variable, alors il existe une ligne de motifs de la forme $(\_\ \dots\ \_\ p_2^i\ \dots\ p_n^i)$ dans la matrice (P_k) et on utilise le même raisonnement que dans le cas précédent. □

Dans le cas où l'automate $A = \mathcal{C}(\vec{x}, (P), (E))$ est la compilation du filtrage par une définition $D = F, (P), (E)$ et que l'on a la transition $(F\ x_1\ x_2 \dots\ x_n, [x_1 \backslash M_1, x_2 \backslash M_2, \dots, x_n \backslash M_n], A) \hookrightarrow^* (N', \sigma', \Omega)$, alors le terme $M = F\ M_1\ M_2 \dots\ M_n$ est tel que, tout numéro de ligne i de la matrice (P), on a l'incompatibilité $\vec{p}^i\ \#_\omega\ \vec{M}$. Par conséquent, pour tout vecteur de motif $\vec{q}$ de $\mathrm{MEP}_i[(P)]$, on a $\vec{q}\ \#_\omega\ \vec{M}$ et le terme M est imprévu pour le système équivalent $\Sigma_{\mathcal{P}}\ (\vec{q} \in \mathrm{MEP}_i[(P)] \Rightarrow \vec{p}^i \preceq_\omega \vec{q})$.

Cette situation fait suite à une erreur de programmation, le programmeur n'a pas prévu tous les cas possibles d'appel d'un combinateur F donné. La figure 6.8 illustre une telle définition incomplète et sa compilation `DEUX` selon l'ordre gauche droite. Manifestement, l'exécution de cet automate peut aboutir à un état terminal indéfini quand le terme analysé n'a pas été prévu —prendre $(x\ y)$ = (`false false`), par exemple—. Il est intéressant de remarquer que ce n'est pas le cas pour tous les termes imprévus. Par exemple, le vecteur de valeurs partielles (Ω `false`) est incompatible avec la ligne de motifs (`true true`) et pourtant la configuration initiale (`deux_vrais x y`, $[x \backslash \Omega, y \backslash$`false`$]$, `match x with`…) est déjà une configuration terminale, dont la composante automate n'est pas Ω. De fait, un automate correct reconnaît obligatoirement les radicaux mais pas les termes imprévus. De sorte que l'automate A ne distingue pas forcément un terme imprévu (`deux_vrais Ω false`, ici) d'un terme qui peut devenir un radical (`deux_vrais Ω Ω`, ici).

`let deux_vrais = function`
`  true true → true`
	`let DEUX x y = match x with`
	`  true → (match y with`
	`          true → true`
	`         | _   → Ω)`
	`| _    → Ω`

Figure 6.8 :

```
let essai F (M1 M2 ... Mn) =
  let (x1,x2,...,xa) = vars F
  and A = code F in
  match termine (F x1 x2... xa,[x1\M1, x2\M2,...,xa\Ma],A) with
    (_,_,Omega)        → raise pattern
  | (_,_,Succes _)     → ϵ
  | (N,_,Filtre (x,_)) → occurrence x N
```

Figure 6.9 : La fonction essai

La fonction suivante `essai` (figure 6.9) exploite la configuration terminale des automates de filtrage. Dans le cas où un terme imprévu est identifié, une exception `pattern` est déclenchée, par la construction CAML `raise pattern`. (Les exceptions traduisent une condition anormale et interrompent immédiatement le calcul en cours). La fonction `essai` utilise les fonction prédéfinies `vars F` et `code F` qui renvoient respectivement la liste des variables introduites au début de la compilation du filtrage défini par le combinateur F et l'automate produit par cette compilation. La fonction prédéfinie `occurrence x N` renvoie l'occurrence de la variable x dans le terme N qui est supposé linéaire et contenir la variable x. La correction de la fonction `essai`, telle qu'établie par le lemme suivant, est une conséquence de la correction des automates.

Lemme 6.6.4 *Pour tout symbole de combinateur F d'arité n et tout vecteur de termes $\vec{M} = (M_1\ M_2\ \ldots\ M_n)$ on a les résultats suivant :*

1. *L'appel* `essai F` $\vec{\mathtt{M}}$ *échoue par une exception* `pattern` *et le terme $F\ M_1\ M_2 \ldots\ M_a$ est imprévu.*
2. *L'appel* `essai F` $\vec{\mathtt{M}}$ *renvoie l'occurrence vide ϵ et le terme $F\ M_1\ M_2 \ldots\ M_a$ est un radical.*
3. *L'appel* `essai F` $\vec{\mathtt{M}}$ *renvoie une occurrence non-vide u, le sous-terme $(F\ M_1\ M_2 \ldots\ M_n)/u$ n'a pas de constructeur de tête et il existe un préfixe strict $N_\Omega = F\ U_1\ U_2 \ldots\ U_n$, avec $N_\Omega \preceq F\ M_1\ M_2 \ldots\ M_n$ et $u \in Q(N_\Omega)$.*

Démonstration : La proposition *1* résulte directement du lemme 6.6.3, la proposition *2* de la correction de l'automate `code F` et la proposition *3* est une conséquence directe de la correction de l'automate `code F`, du lemme 6.5.9 et de l'égalité $Q(F\ U_1\ U_2 \ldots\ U_n) = \mathrm{Dir}(F\ U_1\ U_2 \ldots\ U_n)$ □

Enfin, la fonction `radical` (figure 6.10) s'applique à tout terme clos. Elle utilise l'appel prédéfini `comb_arite F` qui renvoie l'arité du combinateur F, ainsi que la notation $\mathtt{1_m}$ pour désigner l'occurrence composée de l'entier 1 répété m fois. Sa correction est donnée par le lemme suivant :

Lemme 6.6.5 *Pour tout terme clos M on considère l'appel de fonction* `radical M`*, on a les deux possibilités suivantes :*

```
let rec radical = function
  c(M₁,M₂,...,Mₐ) → raise tete
| F M₁ M₂... Mₙ →
    let a = arite F in
    if n < a then raise tete
    else
      match essai F  (M₁ M₂ ... Mₐ) with
        ε → 1ₙ₋ₐ
      | u → 1ₙ₋ₐ.radical ((F M₁ M₂... Mₐ)/u)
```

Figure 6.10 : Calcul d'un radical standard

1. *L'appel* `radical M` *termine et renvoie une occurrence u qui est l'occurrence d'un radical standard à partir du terme M.*
2. *L'appel* `radical M` *échoue, une exception est déclenchée et le terme M est en forme normale de tête.*

Démonstration: Tout d'abord, il est clair que l'appel `radical M` termine ou échoue toujours, puisque l'unique appel récursif prend un sous-terme strict du terme M en argument. L'exception `tete` est déclenchée en deux occasions, quand le terme M possède un constructeur de tête, ou bien quand le terme M est un combinateur appliqué à un nombre insuffisant d'arguments —auquel cas le type du terme M est fonctionnel—. Or, le sous-terme $N = (F\ M_1\ M_2 \dots\ M_a)/u$ de l'appel récursif de la fonction `radical` ne possède pas de constructeur de tête (lemme 6.6.4-*3*) et son type est celui d'un motif qui n'est jamais un type fonctionnel. Par conséquent, l'exception `tete` ne peut être levée que par l'appel initial de la fonction `radical` et le terme M est alors trivialement en forme normale de tête. Pour traiter les deux autres cas, on utilise le lemme 6.6.1. Soit donc un appel de la forme `radical M` qui donne lieu à un appel de la fonction `essai`. Un tel terme est nécessairement de la forme $M = F\ M_1\ M_2 \dots\ M_n$, où l'entier n est supérieur ou égal à l'arité a du combinateur F. Alors, il existe une suite $(N_i)_{1 \le i \le n-a}$ qui amorce la construction d'une suite de sous-termes du terme M telle que définie dans les hypothèses du lemme 6.6.1. On pose $N_1 = M$, $N_{n-a} = F\ M_1\ M_2 \dots\ M_a$ et, pour tout entier i de l'intervalle $[1 \dots n-a[$, $N_{i+1} = N_i/1$, $P_i = @(\Omega, \Omega)$ et $v_i = 1$. On vérifie facilement les trois conditions du lemme 6.6.1, puisque l'on a ici :

$$N_i = @(N_{i+1}, M_{n-i+1}) \qquad Q(@(\Omega,\Omega)) = \{1\} \qquad \{\, w \in \mathcal{O}(@(\Omega,\Omega)) \mid w \succ \epsilon \text{ et } M/w\!\uparrow_+ \} = \emptyset$$

Il y a ensuite trois possibilités pour l'appel `essai F M₁ M₂... Mₙ` qui sont reprises du lemme de correction de la fonction `essai` :

1. Si l'exception `pattern` est levée, alors le sous-terme N_{n-a} est imprévu et le terme M est en forme normale de tête.
2. Si l'occurrence vide ϵ est renvoyée, alors le terme N_{n-a} est un radical qui est un radical standard à partir du terme M.
3. Sinon, une occurrence non vide u est renvoyée et il existe un préfixe strict $N_\Omega \prec N_{n-a}$, avec $u \in Q(N_\Omega)$. Il reste à prolonger d'un terme la suite des N_i, en posant $N_{n-a+1} = N_{n-a}/u$, $P_{n-a} = N_\Omega$ et $v_{n-a} = u$. Ici encore, il n'y pas de sous-terme strict du préfixe strict $P_{n-a} = F\ U_1\ U_2 \dots\ U_a$ qui est proprement compatible, en raison de la règle de l'arité des combinateurs et du fait que le symbole de tête d'un terme proprement compatible n'est jamais

```
let rec force (σ,A) = match pas (σ,A) with
  Oui (σ',A') →  force (σ',A')
| Stop        →  match A with
    Filtre (x,cls)  →
     let M = reduit (σ(x)) in
     let σ' =  remplace σ x M in
     force (σ',Filtre (x,cls))
  | _  →  (σ,A)

and appelle F (M1 M2 ... Mn) =
  let (x1,x2,...,xn) = vars F
  and A = code F in
  match termine ([x1\M1, x2\M2,...,xn\Mn],A) with
    (_,Omega)    →  raise pattern
  | (σ,Succes t) →  σ(t)

and reduit = function
  c(M1,M2,...,Ma) →  c(M1,M2,...,Ma)
| F M1 M2 ... Mn  →
    let a = comb_arite F in
    if n < a then F M1 M2 ... Mn
    else
      let Q = appelle F (M1 M2 ... Ma) in
      reduit (Q Ma+1 ... Mn)
```

Figure 6.11 : Appel par nom

un constructeur. Le reste de la suite des N_i est construit par l'appel inductif `radical` $\mathtt{N}_{\mathtt{n-a+1}}$. Le dernier terme étant un terme imprévu —auquel cas le terme M est en forme normale de tête—, ou bien un radical —qui est standard dans le terme M—. □

On peut donc maintenant mettre en œuvre naïvement l'appel par nom de la façon suivante : calculer une occurrence de radical u standard dans le terme M, puis réduire le radical de la forme $\sigma(\alpha) = F\ N_1\ N_2 \ldots\ N_n$ à l'occurrence u et recommencer tant que le terme courant n'est pas en forme normale de tête. Pour calculer le contractum $\sigma(\beta)$ à l'occurrence u, on peut utiliser une nouvelle fois l'automate `code F`. L'appel `radical (M[u` $\leftarrow \sigma(\beta)$`])` entraînera ensuite les mêmes appels récursifs de de la fonction `radical`, jusqu'à redécouvrir l'occurrence u. On peut considérablement améliorer ce schéma naïf en utilisant directement le contractum calculé dès la première analyse du radical M/u par l'automate `code F` et en reprenant ensuite la recherche du radical standard suivant à partir de l'occurrence u. Ces remarques justifient les nouvelles fonctions `force`, `appelle` et `reduit` qui correspondent aux fonctions `termine`, `essai` et `radical` respectivement, tout en procédant au remplacement du radical par le contractum dès que possible (cf. la figure 6.11). Pour ce faire, la fonction `force` utilise l'appel de fonction prédéfini `remplace` σ `x M` qui produit une substitution σ' qui se confond avec la substitution σ, sauf sur la variable x, pour laquelle on a $\sigma'(x) = M$. La fonction `reduit` procède plus directement en fabriquant le nouveau terme `@(...@(Q,Ma+1)...,Mn)`, où Q est le contractum qui vient d'être calculé par la fonction `appelle`. Enfin, la première composante des configurations d'automates (le préfixe exploré) est devenue inutile et la fonction `force` prend en argument une configuration réduite à une paire `(`σ`,A)`.

En fait, la fonction `reduit` ne réalise pas tout à fait la stratégie de l'appel par nom telle que définie en 4.4.18. La faute en est à l'impossibilité de déterminer si un terme est en forme normale de tête dans le cas général. Par exemple, si on ajoute la définition de combinateur $\perp \;\to\; \perp$ aux définitions des primitives booléennes (figure 6.1), alors l'appel `reduit (@(@(and,true),`$\perp$`))` bouclera, car la fonction `force` appellera indéfiniment la fonction `reduit` sur le sous-terme $\perp$. Pourtant, le terme applicatif `@(@(and,true),`$\perp$`)` est déjà une forme normale de tête. Heureusement, ce type de forme normale de tête est sans intérêt, car un résultat significatif produit par le combinateur `and` est `true` ou `false`. C'est pourquoi il est sensé de restreindre son attention à la classe réduite des formes normales de tête qui sont ici les seules significatives.

Définition 6.6.6 (Forme normale de tête faible) *Une forme normale de tête faible est un terme applicatif M qui est soit de la forme $M = F\ M_1\ M_2 \dots\ M_n$, où l'entier n est strictement plus petit que l'arité du combinateur F ; soit de la forme $M = c(M_1, M_2, \dots, M_a)$, où c est un constructeur.*

Les formes normales de tête faibles sont significatives, non seulement parce que le programmeur s'attend à voir tout objet réduit d'un type de base posséder un constructeur de tête, mais aussi dans le sens que si l'appel `radical M` échoue en levant l'exception `pattern`, alors le terme M est une forme normale de tête sans être une forme normale de tête faible. En effet, Le préfixe préservé du cas *2* du lemme 6.6.1 n'est jamais une forme normale de tête faible. Or, le déclenchement de l'exception `pattern` sanctionne la faute du programmeur qui a oublié des cas en définissant ses fonctions.

Les formes normales de tête faibles sont aisément détectées par la fonction `reduit` qui retourne immédiatement quand un terme en forme normale de tête faible lui est donné en argument. Plus généralement, quand la fonction `reduit` termine normalement, alors elle renvoie toujours un terme en forme normale de tête faible. Réciproquement, si la fonction `reduit` échoue ou ne termine pas, alors le terme courant de la réduction n'est pas une forme normale de tête faible. Le cas de l'échec a été traité au paragraphe précédent. Considérons ensuite un appel `reduit M` qui ne termine pas, il y a deux possibilités :

- Si l'invocation de la fonction `appelle` boucle, alors le terme courant de la dérivation en cours est un combinateur appliqué à son bon nombre d'arguments ou plus et n'est pas une forme normale de tête faible.
- Sinon, l'appel initial de la fonction `reduit` se prolonge indéfiniment par un appel récursif terminal. Dans ce cas, le terme courant au moment de chaque appel terminal n'est jamais une forme normale de tête faible, puisqu'alors la fonction `reduit` retournerait immédiatement.

L'algorithme de réduction `reduit` est correct pour la mise en forme normale de tête faible.

Lemme 6.6.7 *Soit un terme M qui possède une forme normale de tête faible. Alors, l'appel de fonction* `reduit M` *calcule une forme normale de tête faible du terme M en un temps fini.*

Démonstration : Si le terme M n'est pas de type fonctionnel et qu'il possède une forme normale de tête faible N, alors cette dernière est forcément de la forme $N = c(N_1, \dots, N_a)$. Dès lors, toutes les formes normales de tête de M admettent le constructeur c comme symbole de tête et sont des formes normales de tête faibles. C'est donc le cas, en particulier, de la forme normale de tête du terme M qui est calculée —puisque seuls des radicaux standards sont contractés— et identifiée —d'après la discussion du paragraphe précédent— par l'appel initial de la fonction `reduit` sur le

```
let rec normalise M = match reduit M with
  c(M1,M2,...,Ma) →  c(normalise M1,normalise M2,...,normalise Ma)
| N               →  N
```

Figure 6.12 : Calcul de la forme normale faible

terme M. Si le terme M est de type fonctionnel et possède une forme normale de tête faible N, alors cette dernière est nécessairement de la forme $N = F\ N_1\ N_2 \ldots\ N_n$. Comme dans le cas des systèmes gauches, la stratégie définie par la fonction `reduit` est un appel par nom vers le préfixe $F\ x_1\ x_2 \ldots\ x_n$, en tant que telle, elle calcule un terme en forme normale de tête en un nombre fini d'étapes, terme qui est identifié immédiatement. □

Au delà des formes normales de tête faibles, le calcul peut être prolongé jusqu'aux formes normales faibles.

Définition 6.6.8 (Forme normale faible) *Un terme M est en forme normale faible, quand on se trouve dans l'un des trois cas suivants :*

- *Le terme M est réduit à une variable.*
- *Le terme M est une forme normale de tête faible de type fonctionnel (c'est à dire que l'on a $M = F\ M_1\ M_2 \ldots\ M_n$, où l'entier n est strictement plus petit que l'arité du combinateur F).*
- *Le terme M est de la forme $M = c(M_1, M_2, \ldots, M_a)$, où c est un constructeur et, pour tout entier i compris entre 1 et a, le sous-terme M_i est une forme normale faible.*

Il est clair que les formes normales faibles se confondent avec les formes normales tout court dans le cas des termes qui ont un type de résultat (i.e., un type qui ne contient aucune flèche). La figure 6.12 illustre l'utilisation du réducteur en forme normale de tête faible `reduit` pour le calcul de la forme normale faible des termes clos. Il est clair que la fonction `normalise` est correcte pour la mise en forme normale faible et que, en particulier, l'appel `normalise M` calcule la forme normale des termes M dont le type est un type de résultat, chaque fois que celle ci-existe.

On peut être plus spécifique. On appelle *préfixe gauche* tout terme linéaire qui est tel que seules des variables suivent une variable quand on fait la liste des sous-termes de ce préfixe en suivant l'ordre gauche croissant (voir aussi la définition 5.6.1 des systèmes gauches). On suppose ensuite, comme c'est d'ailleurs le cas dans GAML, que les appels récursifs de la la fonction `normalise` se font de la gauche vers la droite. Dès lors, si un terme M est normalisable relativement à un préfixe γ qui est à la fois un préfixe gauche et une forme normale faible, alors l'appel `normalise M` définit une stratégie d'appel par nom relativement à γ et atteindra une forme normale relativement au préfixe γ en un nombre fini d'étapes. Ce mode de calcul est tout à fait naturel, il consiste, par exemple, à évaluer les listes de la gauche vers la droite en réduisant leurs éléments en suivant l'appel par nom.

Dans la mise en œuvre réelle de GAML, le terme en cours de réduction est représenté par une structure de données globale, les substitutions étant quant à elles mises en œuvre à l'aide d'une pile de pointeurs vers cette structure globale. Cette structure de données est bien entendu un graphe. Ce graphe est acyclique à condition d'utiliser une procédure de λ-lifting qui n'introduit pas de définition récursives locales, ce que l'on a supposé au début de ce chapitre (voir [Aug 87, Pey 87] pour des précisions sur ce point) et d'éviter les combinateurs d'arité zéro récursifs. Compte tenu de ces restrictions (mineures, par rapport à la contrainte de séquentialité), j'estime que la fonction `normalise` est une modélisation satisfaisante de l'algorithme de réduction utilisé dans l'environnement d'exécution de GAML et démontre que cet algorithme réalise la stratégie de réduction paresseuse. En effet, à chaque étape, le réducteur `reduit` contracte un radical standard et donc, par le théorème 5.2.5 et la définition des familles standards, la mise en œuvre de ce réducteur dans les graphes acycliques réalise la stratégie paresseuse.

Par conséquent, le réducteur de graphe de l'environnement d'exécution GAML bénéficie de toutes les propriétés d'optimalité définies à la section 5.5. En particulier, ce réducteur est correct et optimal pour le calcul des formes normales et des formes normales de tête faibles. Il est également correct et optimal pour le calcul de tous les résultats intermédiaires qui sont des préfixes gauches en forme normale faible.

Bibliographie

[Aba 90] M. Abadi, L. Cardelli, P.-L. Curien, J.-J. Lévy, "*Explicit substitutions*". Conférence Principle Of Programming Languages, 1990.

[Ari 90] Z. Ariola, Arvind, "*Compilation of Id$^-$: a subset of Id*". CSG Memo 315, MIT, novembre 1990.

[Aug 87] L. Augustsson, "*Compiling Lazy Functional Languages Part II*". Ph.D. thesis, Chalmers University of Technology, Suède, 1987.

[Bae 87] J.C.M. Baeten, J.A. Bergstra, J.W. Klop, "*Term Rewriting Systems with Priorities*". Conference Rewriting Techniques and Applications, 1987 (LCNS 256).

[Ber 79] G. Berry, J.-J. Lévy, "*Minimal and Optimal Computations of Recursive Programs*". Journal of the Association for Computing Machinery, Vol. 26, No 1, 1979.

[Bar 87] H.P. Barendregt, M.C.D.J. van Eekelen, J.R.W. Glauert, J.R. Kennaway , M.J. Plasmeijer, M.R. Sleep, "*Term Graph Rewriting*". Conférence Parallel Archtecture and Languages Europe, 1987 (LCNS 259).

[Bru 72] N. de Bruijn, "*Lambda-Calculus notation with nameless dummies, a tool for automatic formula manipulation with application to the Church-Rosser theorem*". Indag. Math. 34(5), pp 381–392, 1972.

[Cou 86] B. Courcelle, "*A Representation of Graphs by Algebraic Expressions and its Use for Graph Rewriting Systems*". Third Workshop on Graph Grammars and their Applications to Computer Science, décembre 1986 (LCNS 291).

[Cur 86] P.-L. Curien, "*Categorical Combinators, Sequential Algorithm and Functional Programming*". Research Notes in Theoretical Computer Science, Pitman Publishing Ltd., 1986.

[Cré 91] P. Crégut, "*Machines à environement pour la réduction symbolique et l'évaluation partielle*". Thèse de l'Université de Paris VII, juin 1991.

[Dam 82] L. Damas, R. Milner, "*Principal type-schemes for functional programs*". Conférence Principles Of Programming Languages, 1982.

[Ehr 86] H. Ehrig, "*Tutorial Introduction to the Algebraic Approach og Graph Grammars*". Third Workshop on Graph Grammars and their Applications to Computer Science, décembre 1986, LCNS 291.

[Fie 90] J. Field, "*On Laziness and Optimality in Lambda Interpreters : Tools for Specification and Analysis*". Conférence Principles Of Programming Languages, 1990.

[Gal 89] J.H. Gallier, "*What's so special about Kruskal's theorem and the ordinal Γ_0 ? A survey of some results in proof theory*". Rapport technique de l'Université de Pennsylvanie, décembre 1989.

[Gen 90] M. Gengler, "*Réductions parallèles dans le Lambda-calcul : Non existence d'une stratégie optimale fondée sur le partage*". Thèse de doctorat de l'Université de Paris VI, 1990.

[Har 92] T. Hardin, J.-J. Lévy, "*Confluence Properties of Weak and Strong Calculi of Explicit Substitutions*". Rapport de recherche INRIA Rocquencourt 1617, février 1992.

[Hin 86] J.R. Hindley, J.P. Seldin, "*Introduction to Combinators and λ-calculus*". London Mathematical Society, Students Texts 1, Cambridge University Press, 1986.

[Hin 77] J.R. Hindley, "*Combinatory reductions and lambda reductions compared*". Zeitschrift für Matematische Logik Grundlagen der Mathematik 23, pages 169-180.

[How 90] B.T. Howard, J.C. Mitchell, "*Operational and Axiomatic Semantics of PCF*". Conférence Lisp and Functional Programming, 1990.

[Hug 84] R.J.M. Hugues, "*The design and implementation of prprogramming languages*". PhD Thesis, PRG-40, Programming Research Group, Oxford, Grande Bretagne.

[Hue 80] G. Huet, "*Confluent Reductions : Abstract Properties and Applications to Term Rewriting Systems*". Journal of the Association for Computing Machinery vol. 27, No 4 (1980), 797–821.

[Hue 79] G. Huet, J.-J. Lévy, "*Call by Need Computations in Non-Ambiguous Linear Term Rewriting Systems*". Rapport de recherche INRIA 359, 1979.

[Hue 91] G. Huet, J.-J. Lévy, "*Computations orthogonal rewriting systems*". In J.-L. Lassez and G. D. Plotkin, editors, *Computational Logic ; Essays in Honor of Alan Robinson*, pages 394–443. MIT Press, 1991.

[Joh 85] T. Johnsson, "*Lamda-lifting : tranforming programms to recursive equations*". Conference Functional Programming and Computer Architecture, 1985.

[Joh 87] T. Johnsson, "*Compiling Lazy Functional Languages*". Ph.D. thesis, Chalmers University of Technology, Suède, 1987.

[Kat 90] V.K. Kathail, "*Optimal Interpreters for Lambda-calculus Based Functional Languages*". Ph.D. thesis, MIT.

[Kah 78] G. Kahn, G. Plotkin, "*Domaines concrets*". Rapport IRIA Laboria 336, 1978.

[Ken 84] J.R. Kennaway, "*An Outline of Some Results of Staples on Optimal Reduction Orders in Replacements Systems*". Internal Report CSA/19/1984.

[Kha 90] Z. Khasidashvili, "*Normalizing Strategies for Orthogonal Term Rewriting Systems*". Proceedings of the I. Vekua Institute of Applied Mathematics, Tbilisi, Géorgie, URSS.

[Klo 80] J.W. Klop, "*Combinatory Reduction Systems*". Thèse de l'Université d'Utrecht, Pays-Bas (1980).

[Klo 89] J.W. Klop, A. Middeldorp, "*Sequentiality in Orthogonal Term Rewriting Systems*". Report Cs-R8932, Centre for Mathematics and Computer Science, Amsterdam, Pays-Bas.

[Kri 85] J.-L. Krivine. "*Un interpréteur du λ-calcul*". Note non-publiée.

[Lam 90] J. Lamping, "*An Algorithm for Optimal Lambda Calculus Reduction*". Conférence Principles Of Programming Languages, 1990.

[Lav 88] A. Laville, "*Evaluation Paresseuse Des Filtrages Avec Priorité, Application au Langage ML*". Thèse de doctorat, Université Paris VII.

[Ler 90] X. Leroy, "*The Zinc experiment : an economical implementation of the ML Language*". Rapport technique INRIA Rocquencourt 117, février 1990.

[Ler 92] X. Leroy, "*Typage polymorhe d'un langage algorithmique*". Thèse de doctorat, Université Paris VII, Juin 1992.

[Lév 78] J.-J. Lévy, "*Réductions correctes et optimales dans le λ-calcul*". Thèse de Doctorat d'Etat, Université Paris VII, 1978.

[Lév 80] J.-J. Lévy, "*Computation an Deduction*". Course note 7 at CMU.

[Mar 91] L. Maranget, "*Gaml : a Parallel implementation of Lazy ML*". Conférence Functionnal Programming and Computer Architecture, 1991.

[Mar 92] L. Maranget, "*Compiling Lazy Pattern Matching*". Conférence Lisp and Functional Programming, 1992.

[Mil 90] R. Milner, M. Tofte, R. Harper, "*The Definition of Standard ML*". The MIT Press.

[O'D 77] M. O'Donnell, "*Computing in Systems Described by Equations*". LNCS No 58, Eds G. Goos and J. Hartmanis, Springer, 1977.

[Pey 87] S.L. Peyton Jones, "*The Implementation of Functional Programming Languages*". Prentice-Hall, 1987.

[Plo 77] G.D. Plotkin, "*LCF Considered as a Programming Language*". Theoretical Computer Science, 5 :225-255, 1977.

[Pue 90] L. Puel, A. Suárez, "*Compiling Pattern Matching by Term Decomposition*". Conférence Lisp and Functional Programming, 1990.

[Sek 92] R.C. Sekar, R. Ramesh and I.V. Ramakrishnan, "*Adaptive Pattern Matching*". Conférence ICALP'92.

[Rob 65] J.A. Robinson, "*A Machine-Oriented Logic Based on the Resolution Principle*". Journal of the Association for Computing Machinery, 12, 1, 1965.

[Sta 80] J. Staples, "*Computations on graph-like expressions*", "*Optimal evaluation of graph-like expressions*", "*Speeding up subtree replacement systems*". Theoretical Computer Science 10 and 11 (1980).

[Tur 79] D. Turner, "*A new implementation technique for applicative languages*". Software : Practice and Experience 9 (1979), pages 31–49.

[Vui 74] J. Vuillemin, "*Syntaxe, Sémantique et axiomatique d'un language de programmation simple*". Thèse de doctorat d'état, Université Paris VI, 1974.

[Wei 90] P. Weis, "*The CAML Reference Manual, version 2.6.1*". Rapport technique INRIA Rocquencourt 121, 1990.

[Wri 91] A.K. Wright, M. Felleisen, "*A Syntactic Approach to Type Soundness*". Technical report of the Rice University COMP TR91-160.

Table des matières

Index

Annexe : le compilateur GAML

(cette annexe est exactement l'article [Mar 91])

Résumé

Les systèmes orthogonaux de réécriture de termes sont étudiés en tant que modèle des langages de programmation. Les conditions d'orthogonalité assurent le déterminisme des calculs dans le sens que que tout programme a au plus un résultat. Un terme qui n'est pas réductible est une forme normale. Le résultat final d'un programme donné est donc ici une forme normale unique. On considère aussi des résultats partiels, qui sont des approximations du résultat final, tels que les formes normales de tête. Les stratégies de réduction désignent, à chaque étape d'un calcul, le sous-terme, ou radical, à réduire pour passer à l'étape suivante. Les radicaux nécessaires sont les radicaux qui sont réduits quelle que soit la stratégie adoptée.

La question des stratégies se pose sur deux plans. Les stratégies *correctes* aboutissent au résultat d'un programme donné, dès que ce résultat existe. Les stratégies d'appel par nécessité et d'appel par nom sont définies, l'une comme réduisant des radicaux nécessaires, et l'autre comme un appel par nécessité particulier qui procède de la racine des termes vers leurs feuilles. La correction de l'appel par nécessité est relative au calcul de la forme normale d'un programme, tandis que l'appel par nom se montre plus précis en produisant des approximations de ce résultat, à coup sûr et dans un ordre bien déterminé.

Les stratégies *optimales* sont les stratégies correctes qui calculent leur résultat en un nombre minimal d'étapes élémentaires. L'exigence d'optimalité impose de considérer la réduction simultanée de plusieurs radicaux et introduit le procédé de réécriture parallèle. Il suffit en fait de considérer la réécriture parallèle qui est réalisable à l'aide de graphes, les radicaux réduits en parallèle correspondant à un sous-terme partagé. Les systèmes de réécriture de graphes sont ici représentés par des systèmes de réécriture de termes *étiquetés*. Les transpositions à la réécriture de graphes des stratégies d'appel par nécessité et d'appel par nom définissent l'appel complet par nécessité et l'appel paresseux respectivement. Cette première stratégie étendue est optimale pour le calcul des formes normales, la seconde étant également optimale pour le calcul des approximations de formes normales.

À titre d'exemples, sont examinés les systèmes à combinateurs, le lambda-calcul faible et le langage ML. Dans ce dernier cas, l'appel par filtrage avec priorité est complètement traité.

Mots-clés

Correction. Étiquettes. Filtrage. Lambda-calcul faible. ML. Optimalité. Réécriture de graphes. Réécriture de termes. Stratégies. Systèmes orthogonaux. Sûreté.