Démonstration automatique : réécriture avancée

Evelyne Contejean
LRI, CNRS UMR 8623, Bât. 490,
Université Paris-Sud, Centre d'Orsay,
91405 Orsay Cedex, France

31 octobre 2006

Table des matières

Chapitre 1

Introduction

Ce document est le support de la partie «Réécriture avancée» du cours de démonstration automatique du MPRI 2006-2007 (2-5-1 et 2-5-2). Le but général de ce cours est d'aborder les notions nécessaires pour comprendre le fonctionnement de la plupart des outils de démonstration automatique développés en particulier dans le monde académique[1], et éventuellement coder un tel outil.

Ce document vise à être auto-suffisant, et contient donc des rappels sur des notions de base qui ne seront que très rapidement abordées durant le cours.

[1] voir la page TPTP `http://www.cs.miami.edu/~tptp/`

Chapitre 2

Petits rappels sur le principe d'induction

Nous aurons l'occasion dans ce cours de faire un certain nombre de preuves par induction sur un ordre bien fondé. Nous rappelons donc ici les quelques notions de base.

Définition 2.1 (Accessibilité) *Soient $\mathscr{A}$ un ensemble et une relation R sur $\mathscr{A}$. Un élément x de $\mathscr{A}$ est accessible par R si (et seulement si) tous les éléments y tels que $y\ R\ x$ le sont. On le note $Acc_R(x)$.*

Par exemple 0 est accessible dans $\mathbb{N}$ par la relation $<$, mais ne l'est pas dans $\mathbb{Z}$.

La définition formelle de l'accessiblité en Coq :

```
Inductive Acc (A : Set) (R : A -> A -> Prop) : A -> Prop :=
     Acc_intro : forall x:A, (forall y:A, R y x -> Acc A R y) -> Acc A R x.
```

La preuve de la réciproque («et seulement si») en Coq :

```
Lemma Acc_inv :
  forall (A : Set) (R : A -> A -> Prop) (x:A),
  Acc A R x -> forall y:A, R y x -> Acc A R y.
Proof.
intros A R x Acc_x; destruct Acc_x; trivial.
Defined.
```

Intuitivement (et non constructivement), un élément est accessible pour R s'il n'a pas de chaînes infinies descendantes à partir de lui. C'est la même notion que celle de la normalisation forte en λ-calcul par exemple.

On peut prouver des propriétés sur les éléments accessibles par induction sur leur accessibilité :

Théorème 2.2 *Soient $\mathscr{A}$ un ensemble, R une relation sur $\mathscr{A}$ et P une prédicat sur $\mathscr{A}$. Pour prouver P sur tous les éléments accessibles par R, il suffit de montrer :*

$$\forall x \quad ((\forall y, \quad y\ R\ x \quad \Rightarrow \quad Py) \quad \wedge \quad \underset{R}{Acc}(x)) \quad \Rightarrow \quad Px$$

```
Lemma mon_induction :
forall (A : Set) (R : A -> A -> Prop) (P : A -> Prop),
       (forall x : A,
```

```
        (forall y : A, R y x -> P y) /\ Acc A R x -> P x) ->
       forall x : A, Acc A R x -> P x.
Proof.
intros A R P IH x Acc_x; induction Acc_x as [a Acc_a IHa].
apply IH; split.
apply IHa.
apply Acc_intro; trivial.
Defined.
```

On peut donc utiliser l'hypothèse supplémentaire ($\forall y,\ \ y\ R\ x \quad \Rightarrow \quad Py$) pour prouver $\forall x\ \ Acc_R(x)) \ \Rightarrow\ Px.$

La notion classique de bonne fondation peut se définir en termes d'accessibilité :

Définition 2.3 *Une relation R est bien fondée sur un ensemble $\mathscr{A}$ si tout élement de $\mathscr{A}$ est accessible par R,*

et le théorème ci-dessus a pour conséquence bien connue le théorème classique d'induction noethérienne :

Théorème 2.4 *Soient $\mathscr{A}$ un ensemble, R une relation sur* bien fondée *sur $\mathscr{A}$ et P une prédicat sur $\mathscr{A}$. Pour prouver P sur tous les éléments de $\mathscr{A}$, il suffit de montrer :*

$$\forall x \quad ((\forall y, \quad y\ R\ x \quad \Rightarrow \quad Py) \quad \Rightarrow \quad Px$$

Chapitre 3

Les algèbres de termes

3.1 Un exemple

Considérons le petit programme CAML suivant :

```
type entier_penao :
    Zero
  | Succ of entier_peano
;;
let rec plus_petit (x,y) =
  match x with
      Zero -> true
    | Succ(x') ->
        begin
          match y with
              Zero -> false
            | Succ(y') -> plus_petit(x',y')
        end
;;
let rec add(x,y) =
  match y with
      Zero -> x
    | Succ(y) -> Succ(add(x,y))
;;
let rec mul(x,y) =
  match y with
      Zero -> Zero
    | Succ(y) -> add(x,(mul(x,y)))
;;
```

Les objets manipulés par ce programme, comme par exemple $add(x, mul(x, y))$ sont des *termes*, bâtis à partir d'une *signature*. Les termes apparaissent naturellement en programmation fonctionnelle, mais aussi en programmation logique (PROLOG), en programmation par contraintes, dans les formalismes logiques et donc *a fortiori* dans les démonstrateurs et les outils d'aide à la preuve.

Cet exemple sera utilisé tout au long de ce chapitre.

3.2 Les termes

Les termes sont définis formellement à partir d'une signature et d'un ensemble de variables.

3.2.1 Signature

Définition 3.1 (Signature) *Une signature est un triplet* $(\mathscr{S}, \mathcal{F}, \tau)$,
- $\mathscr{S}$ *est un ensemble non vide de sortes,*
- $\mathcal{F}$ *est un ensemble de symboles de fonctions.*
- τ *est une fonction définie sur les éléments de* $\mathcal{F}$ *et à valeur dans les séquences non vides d'éléments de* $\mathscr{S}$*. Si* $\tau(f) = (s_1, \ldots, s_n, s)$*, on note*

$$f : s_1 \times s_2 \times \ldots \times s_n \to s$$

Dans ce cas, le domaine de f *est égal à* $s_1 \times \ldots \times s_n$*, son codomaine est égal à* s *et l'arité de* f *est égale à* n*. Les symboles de fonction d'arité 0 seront parfois appelés symboles de constante.*

Si l'ensemble $\mathscr{S}$ est réduit à un singleton, il suffit de connaître l'arité de chaque symbole de fonction de $\mathcal{F}$ pour définir τ.

Exemple 3.2 *La signature utilisée pour les entiers de Peano est égale à* $(\mathscr{S}_{peano}, \mathcal{F}_{peano}, \tau_{peano})$ *:*

$$\begin{aligned} \mathscr{S}_{peano} &= \{entier\_peano, bool\} \\ \mathcal{F}_{peano} &= \{Zero, Succ, plus\_petit, add, mul\} \end{aligned}$$

τ_{peano} *est définie par :*

$$\begin{aligned} &Zero : entier\_peano \\ &Succ : entier\_peano \to entier\_peano \\ &plus\_petit : entier\_peano \times entier\_peano \to bool \\ &add : entier\_peano \times entier\_peano \to entier\_peano \\ &mul : entier\_peano \times entier\_peano \to entier\_peano \end{aligned}$$

3.2.2 Une première définition des termes

On peut définir récursivement l'algèbre des termes construits sur une signature et éventuellement un ensemble de symboles de variable comme le plus petit ensemble tel que :

Définition 3.3 (Termes) *Soit* $(\mathscr{S}, \mathcal{F}, \tau)$ *une signature et* $\mathcal{X} = \uplus_{s \in \mathscr{S}} \mathcal{X}_s$ *un ensemble dont les éléments sont appelés symboles de variable, et tel que les* $\mathcal{X}_s$ *sont deux à deux disjoints.*
- *si* x *appartient à* $\mathcal{X}_s$*,* x *est un terme de sorte* s*.*
- *si* f *appartient à* $\mathcal{F}$*,* $f : s_1 \times s_2 \times \ldots \times s_n \to s$*,* $t_1, \ldots, t_n$ *sont des termes respectivement de sortes* $s_1, \ldots, s_n$*, alors* $f(t_1, \ldots, t_n)$ *est un terme de sorte* s*.*

L'ensemble des termes bâtis sur $(\mathscr{S}, \mathcal{F}, \tau)$ et $\mathcal{X}$ sera noté $\mathcal{T}(\mathcal{F}, \mathcal{X})$. Si $\mathcal{X}$ est réduit à l'ensemble vide, $\mathscr{T}(\mathcal{F}, \emptyset)$ sera notée $\mathcal{T}(\mathcal{F})$ et appelée l'algèbre des termes clos sur $\mathcal{F}$. Notons que cette algèbre est non vide si et seulement si $\mathcal{F}$ contient au moins un symbole de constante.

Exemple 3.4 *$mul(add(Zero, Succ(Succ(Zero))), Succ(Zero))$ est un terme bâti sur la signature $(\mathscr{S}_{peano}, \mathcal{F}_{peano}, \tau_{peano})$.*

Un terme peut être vu comme un arbre fini étiqueté par des éléments de $\mathcal{X}$ et $\mathcal{F}$. Pour cela, nous commencerons par introduire la notion de position.

3.2.3 Positions

Soit $\mathbb{N}$ l'ensemble des entiers naturels usuels, $\mathbb{N}_+$ l'ensemble des entiers naturels strictement positifs, et $\mathbb{N}_+^*$ l'ensemble des séquences sur $\mathbb{N}_+$. La séquence vide sera notée Λ, et la concaténation de deux séquences p et q par $p \cdot q$. $\mathbb{N}_+^*$ est muni d'un ordre partiel $\leq_{pref}$ défini par :

$$p \leq_{pref} q \text{ si et seulement si } \exists r \in \mathbb{N}_+^* \quad p \cdot r = q$$

Si $p \leq_{pref} q$, on dira que p est un préfixe de q. Un ensemble de positions $\mathscr{P}$ est clos par préfixe si

$$\forall p, q \in \mathbb{N}_+^* \quad (p \leq q) \wedge q \in \mathscr{P} \Rightarrow p \in \mathscr{P}$$

Une feuille d'un ensemble de positions clos par préfixe est une position qui n'est préfixe d'aucune autre. Un ensemble de positions $\mathscr{P}$ est un *arbre de positions* s'il est clos par préfixe et si

$$\forall p \in \mathbb{N}_+^* \forall n, i \in \mathbb{N}_+ \quad (p \cdot n \in \mathscr{P}) \wedge (i \leq n) \Rightarrow p \cdot i \in \mathscr{P}$$

3.2.4 Une deuxième définition des termes

Définition 3.5 (Termes) *Soient $(\mathscr{S}, \mathcal{F}, \tau)$ une signature, $\mathcal{X} = \uplus_{s \in \mathscr{S}} \mathcal{X}_s$ un ensemble de symboles de variable. Soient $\mathscr{P}$ un arbre de positions, et t une fonction de $\mathscr{P}$ dans $\mathcal{F} \cup \mathcal{X}$ tels que*

- *Si $t(p)$ appartient à $\mathcal{X}$ ou est un symbole de constante de $\mathcal{F}$, alors p est une feuille de $\mathscr{P}$.*
- *Si $t(p) = f$, avec $f : s_1 \times \ldots \times s_n \to s$, alors $p \cdot i$ appartient à $\mathscr{P}$, pour $i = 1, \ldots n$, $f_i = t(p \cdot i)$ a pour codomaine s_i, et*

$$\forall j \in \mathbb{N}_+ \quad p \cdot j \in \mathscr{P} \Rightarrow j \leq n$$

$\mathscr{P}$ sera noté $\mathcal{P}os(t)$, et sera appelé l'ensemble des positions de t.

Il faut noter que cette définition n'est pas exactement équivalente à celle qui précède, car ici, il est possible de définir des termes infinis comme celui qui suit :

Exemple 3.6 *Nous utilisons encore une fois la signature $(\mathscr{S}_{peano}, \mathcal{F}_{peano}, \tau_{peano})$. $\mathscr{P}$ est égal à l'ensemble (infini) de toutes les séquences composées uniquement de 1 :*

$$\mathscr{P} = \{\Lambda, 1, 1 \cdot 1, 1 \cdot 1 \cdot 1, 1 \cdot 1 \cdot 1 \cdot 1, \ldots\}$$

et la fonction t est la fonction constante qui associe à chaque position le symbole de fonction $Succ$.

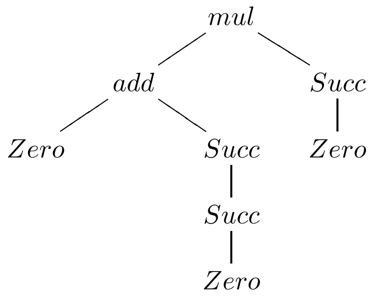
Si nous nous restreignons aux cas où l'ensemble des positions est fini, nous retrouvons les mêmes termes que ceux de la définition 3.3. Il est facile de voir que si t est un terme de sorte s au sens de la définition 3.3, alors $t(\Lambda)$ (au sens de la définition 3.5) a pour codomaine s. La profondeur d'un terme t est la longueur maximale des positions de $\mathcal{P}os(t)$ et la taille de t est égale au cardinal de $\mathcal{P}os(t)$.

Exemple 3.7 *Le terme $mul(add(Zero, Succ(Succ(Zero))), Succ(Zero))$ peut ainsi être vu comme l'arbre étiqueté défini par :*

$$\mathscr{A} = \{\Lambda, 1, 1 \cdot 1, 1 \cdot 2, 1 \cdot 2 \cdot 1, 1 \cdot 2 \cdot 1 \cdot 1, 2, 2 \cdot 1\}$$

et

$$\begin{array}{lcl} t(\Lambda) & = & mul \\ t(1) & = & add \\ t(1 \cdot 1) & = & Zero \\ t(1 \cdot 2) & = & Succ \\ t(t \cdot 2 \cdot 1) & = & Succ \\ t(t \cdot 2 \cdot 1 \cdot 1) & = & Succ \\ t(2) & = & Succ \\ t(2 \cdot 1) & = & Zero \end{array}$$



3.2.5 Sous-termes

Intuitivement un sous-terme d'un terme vu comme une expression (définition 3.3) est une sous-expression. Cependant pour repérer de quelle sous-expression on parle, on utilise la notion de position, c'est pourquoi nous utiliserons la définition 3.5 :

Définition 3.8 (Sous-termes) *Soit t un terme de $\mathcal{T}(\mathcal{F}, \mathcal{X})$, et p une position de $\mathcal{P}os(t)$. Le sous-terme de t à la position p noté $t|_p$ est le terme dont l'ensemble des positions est égal à*

$$\{q \in \mathbb{N}_+^* \mid p \cdot q \in \mathcal{P}os(t)\}$$

et défini par

$$t|_p(q) = t(p \cdot q)$$

Exemple 3.9 *Soit t le terme $mul(add(Zero, Succ(Succ(Zero))), Succ(Zero))$. Le terme $t|_{1 \cdot 2}$ est égal à $Succ(Succ(Zero))$.*

Exercice 3.10 *Vérifier que cette définition est correcte, c'est-à-dire que l'ensemble des positions de $t|_p$ est bien un arbre de positions, et que les conditions de la définition 3.5 sur l'arité et les sortes sont bien satisfaites.*

Il est facile de voir que $t|p$ est un terme de sorte égale au codomaine de $t(p)$.

Définition 3.11 *Soit t, et u deux termes de $\mathcal{T}(\mathcal{F}, \mathcal{X})$ et p une position de t telle que $t|_p$ et u sont deux termes de même sorte. Le terme $t[u]_p$ est le terme défini par*

$$\mathcal{P}os(t[u]_p) = \{q \in \mathbb{N}_+^* \mid q \in \mathcal{P}os(t) \wedge p \not\leq_{pref} q\} \cup \{p \cdot q \mid q \in \mathcal{P}os(u)\}$$

et

$$\begin{array}{lcll} t[u]_p(q) & = & t(q) & \text{si } q \in \mathcal{P}os(t) \wedge p \not\leq_{pref} q \\ t[u]_p(p \cdot q) & = & u(q) & \end{array}$$

$t[u]_p$ est égal au terme t, où l'on a mis le terme u à la position p.

Exemple 3.12 *Soit t le terme $mul(add(Zero, Succ(Succ(Zero))), Succ(Zero))$, et u le terme $mul(Zero, Zero)$. $t[u]_{1\cdot 2}$ est égal à $mul(add(Zero, mul(Zero, Zero)), Succ(Zero))$.*

Exercice 3.13 *Soient t et u deux termes de $\mathcal{T}(\mathcal{F}, \mathcal{X})$ et p une position de t telle que $t|_p$ et u ont la même sorte. Vérifier que*
- *$t[u]_p$ est bien défini par rapport à la définition 3.5.*
- *Le terme t et le terme $t[t|_p]_p$ sont identiques.*
- *Le terme u et le terme $(t[u]_p)|_p$ sont identiques.*

3.3 Les substitutions

Définition 3.14 (Substitution) *Soit $\mathcal{T}(\mathcal{F}, \mathcal{X})$ une algèbre de termes. Une substitution σ de $\mathcal{T}(\mathcal{F}, \mathcal{X})$ est une application de $\mathcal{X}$ dans $\mathcal{T}(\mathcal{F}, \mathcal{X})$ qui conserve les sortes et qui est égale à l'identité sauf sur un nombre fini de symboles de variables. Cet ensemble fini est appelé le domaine de σ et est noté $\mathcal{D}om(\sigma)$. Si $\mathcal{D}om(\sigma) = \{x_1, \ldots, x_n\}$, σ est parfaitement définie dès lors que l'on connaît $t_1, \ldots, t_n$ les valeurs prises sur $x_1, \ldots, x_n$. On utilisera la notation*

$$\sigma = \{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$$

Si tous les termes $t_1, \ldots, t_n$ sont clos, la substitution elle-même sera dite close.

La notation usuelle dans la littérature pour $\sigma(x)$ est postfixée : $x\sigma$.

Une substitution σ s'étend de façon naturelle en une fonction *unique* $\mathscr{H}_\sigma$ de $\mathcal{T}(\mathcal{F}, \mathcal{X})$ dans $\mathcal{T}(\mathcal{F}, \mathcal{X})$ par :
- $\mathscr{H}_\sigma(x) = x$ si $x \notin \mathcal{D}om(\sigma)$,
- $\mathscr{H}_\sigma(x_i) = x_i\sigma$ si $x_i \in \mathcal{D}om(\sigma)$,
- $\mathscr{H}_\sigma(f(s_1, \ldots, s_m)) = f(\mathscr{H}_\sigma(s_1), \ldots, \mathscr{H}_\sigma(s_m))$ si $f(s_1, \ldots, s_m)$ est un terme de $\mathcal{T}(\mathcal{F}, \mathcal{X})$.

À partir de maintenant nous confondrons volontairement une substitution et son extension.

La composition des substitutions est ainsi définie comme la composition usuelle des fonctions. Il est facile de vérifier que la fonction obtenue par composition de deux substitutions est une substitution car elle est égale à l'identité sauf sur un nombre fini de variables. Attention, pour rester compatible avec la notation postfixée, $\sigma \circ \theta$ va s'écrire $\theta\sigma$.

Une substitution σ est idempotente si $\sigma\sigma = \sigma$. Toute substitution close est idempotente.

3.3.1 Subsumption

La notion de substitution est intimement liée à celle d'instance et de subsumption :

Définition 3.15 (Subsumption) *Soit $\mathcal{T}(\mathcal{F}, \mathcal{X})$ une algèbre de termes et soient s et t deux termes de cette algèbre. s est* plus général *que t, ce qui est noté $s \geq t$, s'il existe une substitution θ de $\mathscr{T}(\mathcal{F}, \mathcal{X})$ telle que $s\theta = t$. On dit que s subsume t ou encore que t est une instance de s (on peut préciser une instance par θ). Cette définition s'étend naturellement aux substitutions : soient σ et τ deux substitutions, σ est plus générale que τ s'il existe une substitution θ telle que $\sigma\theta = \tau$.*

Exercice 3.16 *Est-ce que la propriété suivante est correcte ? σ est plus générale que τ si et seulement si pour toute variable x du domaine de σ ou du domaine de τ, $x\sigma$ est plus général que $x\tau$. Si oui, donner une preuve, sinon, un contre-exemple.*

Contre-exemple : avec une signature monosortée $\mathcal{F} = \{f, g\}$, f étant un symbole binaire et g un symbole unaire $\mathcal{X} = \{x, y, z, u\}$.

$$\begin{aligned}\sigma &= \{x \mapsto f(z,u), y \mapsto g(z)\} \\ \tau &= \{x \mapsto f(g(u),u), y \mapsto g(f(u,u))\}\end{aligned}$$

$x\sigma$ est plus général que $x\tau$, $x\sigma\{z \mapsto g(u)\} = x\tau$, et $y\sigma$ est plus général que $y\tau$, $y\sigma\{z \mapsto f(u,u)\} = y\tau\}$, mais σ n'est pas plus générale que τ, car on ne peut pas trouver une même substitution θ telle que $x\sigma\theta = x\tau$ et $y\sigma\theta = y\tau$.

3.3.2 Renommages

Dans la suite, nous utiliserons parfois des substitutions particulières appelées renommages, et qui sont pour les termes l'analogue de l'α-conversion en λ-calcul.

Définition 3.17 *Soit $\mathcal{T}(\mathcal{F}, \mathcal{X})$ une algèbre de termes. Un renommage est une substitution $\sigma = \{x_1 \mapsto y_1, \ldots, x_n \mapsto y_n\}$ telle que les y_i sont des variables deux à deux distinctes.*

Si t est un terme de $\mathcal{T}(\mathcal{F}, \mathcal{X})$ et ρ un renommage dont le domaine contient toutes les variables de t, le terme $t' = t\rho$ est appelé un renommage de t par ρ, et on notera $t \doteq t'$.

Exercice 3.18 *Montrer que $\doteq$ est une relation d'équivalence.*

Exemple 3.19 *Si ρ est un renommage égal à $\{x \mapsto y, x' \mapsto y'\}$, et t est un terme égal à $add(add(x,y),z)$, on ne peut pas parler du renommage $add(add(x',y'),z)$ de t par ρ car $\mathcal{D}om(\rho)$ ne contient pas z, mais il est possible d'étendre ρ en $\rho' = \{x \mapsto y, x' \mapsto y', z \mapsto z\}$, et de parler du renommage de t en $add(add(x',y'),z)$ par ρ'.*

Si nous considérons maintenant le terme $t' = add(x,y)$, on ne peut pas parler non plus du renommage de t' en $add(y,y)$ par le renommage ρ, mais cette fois on ne peut pas trouver un renommage ρ'' dont le domaine contient x et y et tel que $add(y,y) = t'\rho''$: $add(y,y)$ est une instance stricte de $add(x,y)$, et non pas un renommage.

3.3.3 Interprétation des termes non clos

Un terme non clos t est en fait interprété par l'ensemble de ses instances closes, $[\![t]\!]$, et on a la propriété suivante :

Proposition 3.20 *Soit $\mathcal{T}(\mathcal{F}, \mathcal{X})$ une algèbre de termes et soient s et t deux termes de cette algèbre.*
- *$s \geq t$ si et seulement si $[\![t]\!] \subseteq [\![s]\!]$.*
- *$s \doteq t$ si et seulement si $[\![t]\!] = [\![s]\!]$.*

Cette proposition justifie le fait que si $s \geq t$, on dit que s est plus général que t, en effet, s représente un ensemble d'instances closes qui contient celui associé à t.

3.4 Les $\mathcal{F}$-algèbres

Jusqu'à présent, nous nous sommes concentrés sur l'aspect purement syntaxique des termes. Les algèbres de termes peuvent être interprétées en des structures bien connues telles que $\mathbb{N}$ muni de $\leq, +$ et de $\times$, en donnant une interprétation des symboles de la signature $\mathcal{F}$.

Définition 3.21 *Soit $(\mathscr{S}, \mathcal{F}, \tau)$ une signature. Une $\mathcal{F}$-algèbre est constituée*
- *d'un support non vide $\mathscr{A}_s$ associé à chaque sorte s de $\mathscr{S}$,*
- *d'une interprétation $f_{\mathscr{A}}$ pour chaque symbole de fonction f de $\mathcal{F}$ telle que si $f : s_1 \times \ldots s_n \rightarrow s$, $f_{\mathscr{A}}$ est une application de $\mathscr{A}_{s_1} \times \ldots \times \mathscr{A}_{s_n} \rightarrow \mathscr{A}_s$*

Exemple 3.22 *On peut définir la $\mathcal{F}_{peano}$-algèbre suivante :*
- *Le support de la sorte entier_peano est l'ensemble $\mathbb{N}$, le support de la sorte bool est l'ensemble des booléens $\{vrai, faux\}$.*
- *L'interprétation de la constante Zero est égale à 0, celle de Succ est la fonction $n \mapsto n+1$, celle de plus_petit est $\leq$, celle de add est $+$, et celle de mul est $\times$.*

Exemple 3.23 *Étant donnée une signature $(\mathscr{S}, \mathcal{F}, \tau)$, si $\mathcal{F}$ contient au moins un symbole de chaque sorte, il existe une $\mathcal{F}$-algèbre «naturelle», qui est $\mathcal{T}(\mathcal{F})$; l'interprétation d'un symbole de fonction $f : s_1 \times \ldots s_n \rightarrow s$, étant définie par*

$$f_{\mathcal{T}(\mathcal{F})}(t_1, \ldots, t_n) = f(t_1, \ldots, t_n)$$

$\mathcal{T}(\mathcal{F}, \mathcal{X})$ est également une $\mathcal{F}$-algèbre.

De même que pour définir une substitution, il suffit de donner les valeurs prises par les symboles de variable de son domaine, on peut définir une application de $\mathcal{T}(\mathcal{F}, \mathcal{X})$ dans une $\mathcal{F}$-algèbre $\mathscr{A}$, compatible avec la structure d'algèbre en donnant les valeurs prises sur $\mathcal{X}$.

Définition 3.24 *Soient $\mathscr{A}$ et $\mathscr{B}$ deux $\mathcal{F}$-algèbres. Un homomorphisme de $\mathscr{A}$ dans $\mathscr{B}$ est un ensemble d'applications $\{h_s \mid s \in \mathscr{S}\}$ tel que pour toute sorte s de $\mathscr{S}$, h_s est une application de $\mathscr{A}_s$ dans $\mathscr{B}_s$, et pour tout symbole de fonction $f : s_1 \times \ldots \times s_n \rightarrow s$*

$$\forall a_1, \ldots, a_n \in \mathscr{A} \qquad h_s(f_{\mathscr{A}}(a_1, \ldots, a_n)) = f_{\mathscr{B}}(h_{s_1}(a_1), \ldots, h_{s_n}(a_n))$$

Une $\mathscr{A}$-assignation est un homomorphisme de $\mathcal{T}(\mathcal{F}, \mathcal{X})$ dans la $\mathcal{F}$-algèbre $\mathscr{A}$. Elle est définie de façon unique à partir des valeurs prises sur $\mathcal{X}$.

3.4.1 Algèbres quotients

Définition 3.25 (Précongruence, congruence) *Soit $\mathscr{A}$ une $\mathcal{F}$-algèbre. Une* précongruence $\sim$ *est une relation définie sur $\mathscr{A}$ compatible avec la structure de $\mathcal{F}$-algèbre, c'est-à-dire que deux éléments en relation appartiennent au support de la même sorte, et pour tout symbole de fonction $f : s_1 \times \ldots \times s_n \rightarrow s$, pour tous les éléments $t_1, \ldots, t_n, u_1, \ldots, u_n$ tels que $t_i, u_i \in \mathscr{A}_{s_i}$, on a l'implication suivante :*

$$t_1 \sim u_1 \wedge \ldots t_n \sim u_n \Rightarrow f_{\mathscr{A}}(t_1, \ldots, t_n) \sim f_{\mathscr{A}}(u_1, \ldots, u_n)$$

Une précongruence est une congruence *si c'est également une relation d'équivalence.*

À partir d'une $\mathcal{F}$-algèbre $\mathscr{A}$ et d'une congruence $\sim$ sur cette algèbre, on peut définir une *$\mathcal{F}$-algèbre quotient $\mathscr{A}/\sim$* de la façon suivante :
- Pour chaque sorte s de $\mathscr{S}$, le support de l'algèbre quotient est égal à $\{\overline{t} \mid t \in \mathscr{A}_s\}$, où $\overline{t}$ désigne la classe d'équivalence de t modulo $\sim$.

– Pour chaque symbole de fonction $f : s_1 \times \ldots \times s_n \to s$, $f_{\mathscr{A}/\sim}$ l'interprétation de f dans l'algèbre quotient est donnée par :

$$f_{\mathscr{A}/\sim}(\overline{t_1}, \ldots, \overline{t_n}) = \overline{f_{\mathscr{A}}(t_1, \ldots, t_n)}$$

Cette définition est correcte :la classe de $f_{\mathscr{A}}(t_1, \ldots, t_n)$ ne dépend que des classes de $t_1, \ldots, t_n$, et pas des représentants choisis, car $\sim$ est une précongruence.

Exemple 3.26 *Si l'on considère une signature* $(\mathscr{S}'_{peano}, \mathcal{F}'_{peano}, \tau'_{peano})$ *un peu moins riche que celle des entiers de Peano, avec une unique sorte entier_peano, et les symboles de fonction* $Zero, Succ, add$ *et* mul, $(\mathbb{N}, 0, n \mapsto n+1, +, \times)$ *est une* $\mathcal{F}'_{peano}$*-algèbre. La relation d'équivalence «égal modulo p», notée* $\equiv_p$ *est une congruence au sens de la définition ci-dessus, et* $(\mathbb{Z}/p\mathbb{Z}, 0, n \mapsto n+1, +, \times)$ *est la* $\mathcal{F}'_{peano}$*-algèbre quotient obtenue.*

Chapitre 4

Logique équationelle

Une équation est une paire de termes de même sorte $\{s, t\}$ habituellement notée $s = t$.

Définition 4.1 *Soit $\mathcal{T}(\mathcal{F}, \mathcal{X})$ une algèbre de termes, et soit E un ensemble d'équations. Une $\mathcal{F}$-algèbre $\mathscr{A}$ est un modèle de E si pour toute $\mathscr{A}$-assignation σ, pour toute équation $s = t$ de E, on a $s\sigma = t\sigma$. On le note $\mathscr{A} \models E$.*

Exemple 4.2 *Reprenons l'exemple 3.2 des entiers de Peano. La $\mathcal{F}_{peano}$-algèbre $(\mathbb{N}, 0, n \mapsto n+1, \leq, +, \times)$ est un modèle de l'ensemble d'équations*

$$\{add(x, y) = add(y, x); \quad mul(add(x, y), z) = add(mul(x, z), mul(y, z))\}$$

4.1 La congruence $=_E$.

Soit $\mathcal{T}(\mathcal{F}, \mathcal{X})$ une algèbre de termes, et soit E un ensemble d'équations. On note $=_E$ la plus petite congruence sur $\mathcal{T}(\mathcal{F}, \mathcal{X})$ telle que, pour toute substitution σ,

$$\forall s = t \in E \quad s\sigma =_E t\sigma$$

Une telle congruence existe et est unique. $\mathcal{T}(\mathcal{F}, \mathcal{X})/ =_E$ est munie d'une structure de $\mathcal{F}$-algèbre de façon canonique, en utilisant la construction de la section 3.4.1. Il est clair que par définition

$$\mathcal{T}(\mathcal{F}, \mathcal{X})/ =_E \models E$$

Proposition 4.3 *Soit $\mathcal{T}(\mathcal{F}, \mathcal{X})$ une algèbre de termes et E un ensemble fini d'équations. Soit $\mathscr{A}$ une $\mathcal{F}$-algèbre telle que $\mathscr{A}$ est un modèle de E. Si $\mathcal{F}$ contient au moins un symbole de chaque sorte, il existe un unique homomorphisme de $\mathcal{T}(\mathcal{F})/ =_E$ dans $\mathscr{A}$.*

On note $E \models s = t$ le fait que tout modèle de E est aussi un modèle de $\{s = t\}$.

4.2 Problème du mot, problème d'unification, problème inductif

Définition 4.4 *Soit $\mathcal{T}(\mathcal{F}, \mathcal{X})$ une algèbre de termes, soit E un ensemble fini d'équations et soit $s = t$ une équation. Le problème du mot (associé à) $s = t$ consiste à décider si $E \models s = t$ (*i.e.* tout modèle de E est aussi un modèle de $\{s = t\}$).*

Exemple 4.5 *En utilisant la signature des entiers de Peano,*

$$\{add(x, Zero) = x;\ \ add(x, Succ(y)) = Succ(add(x, y))\} \models add(x, Succ(Succ(y)) = Succ(Succ(add(x, y))$$

Définition 4.6 *Soit $\mathcal{T}(\mathcal{F}, \mathcal{X})$ une algèbre de termes, soit E un ensemble fini d'équations et soit $s = t$ une équation. Le problème d'unification (modulo E) (associé à) $s = t$ consiste à trouver toutes les substitutions σ telles que $E \models s\sigma = t\sigma$.*

Exemple 4.7 *Soit $E = \{add(x, Zero) = x;\quad add(x, Succ(y)) = Succ(add(x, y))\}$. Une des solutions au problème d'unification modulo E $add(x, Succ(y)) = Succ(Succ(z))$ est $\{y \mapsto Succ(u); z \mapsto add(x, u)\}$.*

Définition 4.8 *Soit $\mathscr{E}$ un ensemble de substitutions. Une* substitution principale *de $\mathscr{E}$ est une substitution σ de $\mathscr{E}$ telle que tous les éléments de $\mathscr{E}$ sont des instances de σ.*

Théorème 4.9 *Soit $\mathcal{T}(\mathcal{F}, \mathcal{X})$ une algèbre de termes, et $s = t$ un problème d'unification modulo $\emptyset$. L'ensemble des solutions de ce problème est vide, ou bien il admet une substitution principale.*

Ce théorème sera démontré au chapitre sur l'unification.

Définition 4.10 *Soit $\mathcal{T}(\mathcal{F}, \mathcal{X})$ une algèbre de termes, soit E un ensemble d'équations, et soit $s = t$ une équation. Le problème inductif modulo E (associé à) $s = t$ consiste à décider si $\mathcal{T}(\mathcal{F})/ =_E \models s = t$.*

4.3 Raisonnement équationnel

Le raisonnement équationnel est ce que nous faisons naturellement pour résoudre le problème du mot. Par exemple, on se donne une signature monosortée contenant une constante e, un symbole unaire I, et un symbole binaire $\cdot$ (infixe). Considérons maintenant l'ensemble d'équations suivantes :

$$\begin{array}{rcl} x \cdot e & = & x \\ x \cdot I(x) & = & e \\ (x \cdot y) \cdot z & = & x \cdot (y \cdot z) \end{array}$$

On reconnaît les équations usuelles des groupes non commutatifs. Maintenant, comment démontrer que dans un groupe, l'élément neutre à droite est aussi un élément neutre à gauche ? La méthode classique consiste à essayer de montrer que pour tout x, on a $e \cdot x = e$ en utilisant les équations de E. Cette démonstration se présente habituellement sous la forme suivante :

$$\begin{array}{lcl} e \cdot I(I(x)) & = & (x \cdot I(x)) \cdot I(I(x)) \\ & = & x \cdot (I(x) \cdot I(I(x))) \\ & = & x \cdot e \\ & = & x \\ x \cdot I(I(y)) & = & (x \cdot e) \cdot I(I(y)) \\ & = & x \cdot (e \cdot I(I(y))) \\ & = & x \cdot y \\ e \cdot x & = & e \cdot I(I(x)) \\ & = & x \end{array}$$

Le raisonnement équationnel peut être formalisé comme l'application des règles de déduction suivantes, en partant des axiomes qui sont les équations de E (on parle alors de logique équationnelle) :

$$\dfrac{}{s = s} \qquad \text{Réflexivité}$$

$$\dfrac{s = t}{t = s} \qquad \text{Symétrie}$$

$$\dfrac{s = t \qquad t = u}{s = u} \qquad \text{Transitivité}$$

$$\dfrac{s = t}{u[s\sigma]_p = u[t\sigma]_p} \qquad \text{Remplacement}$$

La démonstration ci-dessus se présente alors sous la forme d'un arbre de dérivation que nous avons décomposé, tout d'abord le sous-arbre correspondant à $e \cdot I(I(x)) = x$, puis l'arbre pour $e \cdot x = x$, où les sous-arbres pour $e \cdot I(I(x)) = x$ ne sont pas détaillés :

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{(x \cdot I(x)) = e}{e = (x \cdot I(x))}}{e \cdot I(I(x)) = (x \cdot I(x)) \cdot I(I(x))} \qquad \dfrac{(x \cdot y) \cdot z = x \cdot (y \cdot z)}{(x \cdot I(x)) \cdot I(I(x)) = x \cdot (I(x) \cdot I(I(x)))}}{e \cdot I(I(x)) = x \cdot (I(x) \cdot I(I(x)))} \qquad \dfrac{x \cdot I(x) = e}{x \cdot (I(x) \cdot I(I(x))) = x \cdot e}}{e \cdot I(I(x)) = x \cdot e} \qquad x \cdot e = x}{e \cdot I(I(x)) = x}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{(x \cdot e) = x}{x = (x \cdot e)}}{x \cdot I(I(y)) = (x \cdot e) \cdot I(I(y))} \qquad \dfrac{\dfrac{(x \cdot y) \cdot z = x \cdot (y \cdot z)}{(x \cdot e) \cdot I(I(y)) = x \cdot (e \cdot I(I(y)))} \qquad \dfrac{\dfrac{\vdots}{e \cdot I(I(x)) = x}}{x \cdot (e \cdot I(I(y))) = x \cdot y}}{(x \cdot e) \cdot I(I(y)) = x \cdot y}}{x \cdot I(I(y)) = x \cdot y}}{\dfrac{e \cdot I(I(x)) = e \cdot x}{e \cdot x = e \cdot I(I(x))}} \qquad \dfrac{\vdots}{e \cdot I(I(x)) = x}}{e \cdot x = x}$$

Si on arrive à obtenir une équation $s = t$ à partir de E et en utilisant les règles de déduction de la logique équationnelle, on le note $E \vdash s = t$.

Le théorème suivant établit la correction de ce mode de raisonnement pour résoudre le problème du mot :

Théorème 4.11 (Birkoff) *Si chaque sorte contient au moins un terme clos, alors*

$$E \vdash s = t \Leftrightarrow E \models s = t \Leftrightarrow s =_E t$$

Démonstration On montre par récurrence sur la longueur de la déduction que si $E \vdash s = t$ alors $E \models s = t$:

- Si la déduction a une longueur nulle, c'est que s et t sont identiques, et dans ce cas $E \models s = t$.
- Si la déduction a une longueur supérieure à 1, alors on isole la dernière déduction :

$$E \vdash E_1 \vdash s = t$$

Par hypothèse de récurrence, $E \models E_1$, et $E_1 \vdash s = t$ par l'une des quatre règles de déduction de la logique équationnelle. L'égalité est une relation d'équivalence dans tout modèle de E, donc si l'on a appliqué une des trois premières règles, $s = t$ est valide dans tout modèle de E. Si l'on appliqué la quatrième règle, s est de la forme $u[s'\sigma]_p$, t de la forme $u[t'\sigma]_p$ et E_1 est égal à $\{s' = t'\}$. Par hypothèse de récurrence, tout modèle de E est un modèle de $s' = t'$, et par définition d'un modèle, c'est aussi un modèle de $s'\sigma$ et $t'\sigma$. Comme l'égalité est une congruence, c'est aussi un modèle de $s = t$. Dans tous les cas, on a $E \models s = t$.

Si $E \models s = t$, comme $\mathcal{T}(\mathcal{F}, \mathcal{X})/=_E$ est un modèle de E, c'est aussi un modèle de $s = t$, s et t sont donc dans la même classe d'équivalence modulo $=_E$: $s =_E t$.

Supposons que $s =_E t$. Considérons la relation $\mathscr{R}_E$ définie par $u\mathscr{R}_E v$ si $E \vdash u = v$: $\mathscr{R}_E$ est une congruence telle que

$$\forall u = v \in E\ \forall \sigma \quad u\sigma \mathscr{R}_E v\sigma$$

Comme $=_E$ est la *plus petite congruence* qui vérifie cette propriété, $s =_E t$ entraîne que $s\mathscr{R}_E t$, donc par définition que $E \vdash s = t$.

$\square$

4.4 Étape équationnelle, théorie équationnelle

$=_E$ est définie comme la plus petite congruence sur $\mathcal{T}(\mathcal{F}, \mathcal{X})$ qui contient E. Nous considérerons une relation plus pauvre, $\leftrightarrow_E$, la plus petite précongruence réflexive contenant E. $=_E$ est alors la clôture transitive de $\leftrightarrow_E$. Si $s \leftrightarrow_E t$, c'est qu'il existe une équation $u = v$ de E et une substitution σ telle que

$$s = s[u\sigma]_p \quad t = s[v\sigma]_p$$

Dans ce cas, on notera

$$s \longleftrightarrow^{p}_{u=v,\sigma} t$$

Cette notation permet une représentation plus agréable des preuves. Par exemple l'arbre de dérivation associé à $e \cdot x = x$ devient :

$$\begin{array}{ll} e \cdot x & \longleftrightarrow^{2}_{N} e \cdot (x \cdot e) \longleftrightarrow^{22}_{I} e \cdot (x \cdot (I(x) \cdot I(I(x)))) \longleftrightarrow^{2}_{A} e \cdot ((x \cdot I(x)) \cdot I(I(x))) \\ & \longleftrightarrow^{21}_{I} e \cdot (e \cdot I(I(x))) \longleftrightarrow^{\Lambda}_{A} (e \cdot e) \cdot I(I(x)) \longleftrightarrow^{1}_{N} e \cdot I(I(x)) \longleftrightarrow^{1}_{I} (x \cdot I(x)) \cdot I(I(x)) \\ & \longleftrightarrow^{\Lambda}_{A} x \cdot (I(x) \cdot I(I(x))) \longleftrightarrow^{2}_{I} x \cdot e \longleftrightarrow^{\Lambda}_{N} x \end{array}$$

La théorie équationnelle associée à E est l'ensemble des équations $s = t$ telles que $E \models s = t$.

4.5 Exercices

Exercice 4.12 *Le but de cet exercice est de montrer que qu'en général le problème du mot et le problème inductif associé à une même équation ont des comportements différents. Considérons la signature $(\mathscr{S}''_{peano}, \mathcal{F}''_{peano}, \tau''_{peano})$ qui comporte une seule sorte et un symbole de fonction constante Zero, un symbole unaire Succ et un symbole binaire add. Soit E l'ensemble d'équations*

$$\{add(x, Zero) = x; \quad add(x, Succ(y)) = Succ(add(x, y))\}$$

Montrer que la propriété $\mathscr{T}(\mathcal{F}''_{peano})/ =_E \models add(Zero, x) = x$ est vraie, mais que $E \models add(Zero, x) = x$ est faux.

Correction On montre par récurrence sur la taille de $x\sigma$ que pour toute substitution close σ, on a

$$add(Zero, x)\sigma =_E x\sigma$$

On commence par montrer que tout terme clos t de $\mathscr{T}(\mathcal{F}''_{peano})$ est égal modulo E à un terme de la forme $Succ(\ldots Succ(Zero)\ldots)$. Pour cela, on montre par récurrence sur la taille de t qu'il existe un terme s de la forme voulue, et que $|s| \leq |t|$, où $|\ |$ désigne la taille d'un terme.

- Le seul terme clos de taille 1 est $Zero$, qui est de la forme voulue.
- Soit t un terme de taille $n+1$, dont le symbole de tête est égal à $Succ$: $t = Succ(t')$. Par hypothèse de récurrence, $t' =_E Succ^m(Zero)$ avec $m+1 \leq |t'|$. Donc $t =_E Succ^{m+1}(Zero)$, et $|t| = 1 + |t'| \geq 1 + (m+1) = |Succ^{m+1}(Zero)|$.
- Soit t un terme de taille $n+1$, dont le symbole de tête est égal à add : $t = add(t', t'')$. Par hypothèse de récurrence $t'' =_E Succ^{m''}(Zero)$ avec $m''+1 \leq |t''|$. Deux cas peuvent se produire :
  - $t'' =_E Zero$, dans ce cas

    $$t = add(t', t'') =_E add(t', Zero) =_E t'$$

    Par hypothèse de récurrence, $t' =_E Succ^{m'}(Zero)$ avec $m'+1 \leq |t'|$. donc

    $$t = add(t', t'') =_E add(t', Zero) =_E t' =_E \overset{m'}{Succ}(Zero)$$

    Pour ce qui est des tailles :

    $$\begin{array}{rcl} |t| & = & 1 + |t'| + |t''| \\ & \geq & |t'| \\ & \geq & m'+1 \\ & = & |\,Succ^{m'}(Zero)| \end{array}$$

  - $t'' =_E Succ^{m''}(Zero)$ avec $m'' \neq 0$. Dans ce cas,

    $$t = add(t', t'') =_E add(t', \overset{m''}{Succ}(Zero)) =_E Succ(add(t', \overset{m''-1}{Succ}(Zero)))$$

    On peut appliquer l'hypothèse de récurrence au terme $add(t', Succ^{m''-1}(Zero))$ car

    $$\begin{array}{rcl} |\,add(t', Succ^{m''-1}(Zero))| & = & 1 + |t'| + (m''-1) + 1 \\ & = & |t'| + m'' + 1 \\ & \leq & |t'| + |t''| \\ & < & |t| \end{array}$$

    Donc $add(t', Succ^{m''-1}(Zero)) =_E Succ^p(Zero)$ avec $p + 1 \leq |\,add(t', Succ^{m''-1}(Zero))|$:

    $$t = add(t', t'') =_E add(t', \overset{m''}{Succ}(Zero)) =_E Succ(add(t', \overset{m''-1}{Succ}(Zero))) =_E Succ(\overset{p}{Succ}(Zero))$$

    Finalement

    $$\begin{array}{rcl} |\,Succ(Succ^p(Zero))| & = & p+2 \\ & \leq & |\,add(t', Succ^{m''-1}(Zero))| + 1 \\ & \leq & |t| \end{array}$$

Grâce à ce qui précède, on peut se restreindre aux substitutions closes de la forme $\sigma = \{x \mapsto Succ^n(Zero)\}$.

– La seule substitution close σ telle que $x\sigma$ est de taille 1 est $\sigma = \{x \mapsto Zero\}$. Dans ce cas,
$$add(Zero, x)\sigma \equiv add(Zero, Zero) =_E Zero \equiv x\sigma$$
– Considérons maintenant une substitution close de la forme $\sigma = \{x \mapsto Succ(t)$, t étant un terme clos. On a alors
$$add(Zero, x)\sigma \equiv add(Zero, Succ(t)) =_E Succ(add(Zero, t))$$
Soit σ' la substitution close définie par $\sigma' = \{x \mapsto t$. Il est clair que la taille de $x\sigma'$ est strictement inférieure à celle de $x\sigma$. En appliquant l'hypothèse de récurrence, on obtient
$$add(Zero, t) \equiv add(Zero, x)\sigma' =_E x\sigma' \equiv t$$
D'où l'on conclut par transitivité et mise sous contexte que
$$add(Zero, x)\sigma \equiv add(Zero, Succ(t)) =_E Succ(add(Zero, t)) =_E Succ(t) \equiv x\sigma$$

On a donc bien $\mathscr{T}(\mathcal{F}''_{peano})/ =_E \models add(Zero, x) = x$. Pour la seconde partie de l'exercice, il suffit de trouver un modèle particulier de E qui ne soit pas un modèle de $add(Zero, x) = x$. Soit donc la $\mathcal{F}''_{peano}$-algèbre de support $\mathbb{N} \times \mathbb{N}$, $Zero$ étant interprété par $(0, 1)$, $Succ$ par la fonction $S : (n, m) \mapsto (n+1, m)$ et add par la fonction $a : (n, m), (n', m') \mapsto (n + n', m)$. Cette $\mathcal{F}''_{peano}$-algèbre est bien un modèle de E car pour toute assignation h définie par
$$h(x) = (n, m) \quad h(y) = (n', m')$$
On a bien
$$\begin{array}{rcl}
h(add(x, Zero)) & = & a(h(x), (0, 1)) \\
 & = & a((n, m), (0, 1)) \\
 & = & (n + 0, m) \\
 & = & (n, m) \\
 & = & h(x) \\
 & & \\
h(add(x, Succ(y))) & = & a(h(x), S(h(y))) \\
 & = & a((n, m), (n' + 1, m')) \\
 & = & (n + n' + 1, m) \\
 & = & S(n + n', m) \\
 & = & S(a((n, m), (n', m'))) \\
 & = & S(a(h(x), h(y))) \\
 & = & h(Succ(add(x, y)))
\end{array}$$
D'autre part, soit h l'assignation définie par
$$h(x) = (0, 2)$$
Cette assignation est telle que
$$\begin{array}{rcl}
h(add(Zero, x)) & = & a((0, 1), h(x)) \\
 & = & a((0, 1), (0, 2)) \\
 & = & (0, 1) \\
 & \neq & (0, 2) = h(x)
\end{array}$$

□

Chapitre 5

Réécriture

Nous avons vu qu'il est difficile de savoir si deux termes sont égaux en utilisant le remplacement des égaux par des égaux. La réécriture permet de limiter drastiquement la recherche et dans les cas favorables, cette technique fournit un algorithme de décision pour tester l'égalité de deux termes.

5.1 Règles de réécriture

Contrairement aux équations de la logique équationnelle, les règles de réécriture sont *orientées* :

Définition 5.1 *Soit $\mathcal{T}(\mathcal{F},\mathcal{X})$ une algèbre de termes. Une règle de réécriture est un couple de termes de $\mathcal{T}(\mathcal{F},\mathcal{X})$, (l,r) qui sera noté $l \to r$. Un terme s de $\mathcal{T}(\mathcal{F},\mathcal{X})$ se récrit en t, par la règle $l \to r$, à la position p, avec la substitution σ si*

$$s|_p \equiv l\sigma \quad t \equiv s[r\sigma]_p$$

On le notera $s \to^p_{l\to r,\sigma} t$. On omettra parfois l'une où l'autre de ces indications. Un système de réécriture R est un ensemble de règles de réécriture. La relation $\to_R$ est définie par

$$s \to_R t \text{ si et seulement si } \exists l \to r \in R \;\; \exists p \in \mathcal{P}os(s) \;\; \exists \sigma \;\; s \to^p_{l\to r,\sigma} t$$

L'ensemble d'équations associés à R, $\{l = r \mid l \to r \in R\}$ est noté E_R.

Notation 5.2 *Dans la suite, nous noterons souvent une relation quelconque sur un ensemble $\mathscr{A}$ (et pas uniquement une relation de récriture sur une algèbre de termes) par $\to_R$. $\to^*_R$ désigne la clôture réflexive transitive de $\to_R$, $\to^+_R$ sa clôture transitive, et $\leftarrow_R$ la relation inverse.*

Définition 5.3 (Forme normale) *Soit $\mathscr{A}$ un ensemble, et soit $\to_R$ une relation définie sur $\mathscr{A}$. Un élément x de $\mathscr{A}$ est en forme normale pour $\to_R$ s'il n'existe pas d'élément y de $\mathscr{A}$ tel que $x \to_R y$. Un élément y est une forme normale d'un élément x s'il est en forme normale et $x \to^*_R y$.*

5.2 Confluence sur les relations abstraites

La réécriture permet de résoudre le problème du mot, c'est-à-dire de tester l'égalité de deux termes, en *calculant* leurs formes normales et en regardant si elles sont syntaxiquement égales. Cette technique n'est correcte que si un terme a une forme normale unique. Nous allons maintenant introduire plusieurs définitions liées à cette propriété.

Définition 5.4 (Church-Rosser, confluence, confluence locale) *Soit $\mathscr{A}$ un ensemble. Une relation $\rightarrow_R$ sur $\mathscr{A}$*

- *possède la propriété de* Church-Rosser *si et seulement si*

$$\forall x, y \in \mathscr{A}, \quad x \underset{R}{\overset{*}{\longleftrightarrow}} y \quad \Rightarrow \quad (\exists z \in \mathscr{A}, \quad x \underset{R}{\overset{*}{\rightarrow}} z \underset{R}{\overset{*}{\leftarrow}} y).$$

- *est* confluente *si et seulement si*

$$\forall x, y, z \in \mathscr{A}, \quad y \underset{R}{\overset{*}{\leftarrow}} x \underset{R}{\overset{*}{\rightarrow}} z \quad \Rightarrow \quad (\exists v \in \mathscr{A}, \quad y \underset{R}{\overset{*}{\rightarrow}} v \underset{R}{\overset{*}{\leftarrow}} z).$$

- *est* localement confluente *si et seulement si*

$$\forall x, y, z \in \mathscr{A}, \quad y \underset{R}{\leftarrow} x \underset{R}{\rightarrow} z \quad \Rightarrow \quad (\exists v \in \mathscr{A}, \quad y \underset{R}{\overset{*}{\rightarrow}} v \underset{R}{\overset{*}{\leftarrow}} z).$$

Les définitions ci-dessus peuvent se voir plus graphiquement :

$$\begin{tikzcd}
x \arrow[rr, leftrightarrow, "R^*"] \arrow[dr, dashed, "R^*"'] & & y \arrow[dl, dashed, "R^*"] \\
& z &
\end{tikzcd}$$

Church-Rosser

$$\begin{tikzcd}
& x \arrow[dl, "R^*"'] \arrow[dr, "R^*"] & \\
y \arrow[dr, dashed, "R^*"'] & & z \arrow[dl, dashed, "R^*"] \\
& v &
\end{tikzcd}$$

Confluence

$$\begin{tikzcd}
& x \arrow[dl, "R"'] \arrow[dr, "R"] & \\
y \arrow[dr, dashed, "R^*"'] & & z \arrow[dl, dashed, "R^*"] \\
& v &
\end{tikzcd}$$

Confluence locale

Remarque : dans le cas de la réécriture, par abus de langage, on dira d'un système de règles de réécriture R qu'il est possède la propriété de Church-Rosser (resp. est confluent, resp. est localement confluent) si c'est le cas pour la relation $\rightarrow_R$.

Proposition 5.5 *Une relation $\rightarrow_R$ possède la propriété de Church-Rosser si et seulement si elle est confluente.*

Démonstration Il est bien évident que Church-Rosser implique la confluence. La démonstration de la réciproque se fait par récurrence sur le nombre de pas de $\longleftrightarrow_R$ entre deux éléments quelconques x et y.

- Si le nombre de pas de $\longleftrightarrow_R$ entre x et y est égal à 0, x et y sont identiques et il est clair que

$$x \underset{R}{\overset{0}{\rightarrow}} x \underset{R}{\overset{0}{\leftarrow}} x \equiv y$$

- Si le nombre de pas de $\longleftrightarrow_R$ entre x et y est égal à $n+1$, soit le premier pas à partir de x est $\rightarrow_R$, soit c'est $\leftarrow_R$:
  - Cas $\rightarrow_R$:

$$x \underset{R}{\rightarrow} x' \underset{R}{\overset{n}{\longleftrightarrow}} y$$

  Par hypothèse de récurrence, il existe v tel que

$$x' \underset{R}{\overset{*}{\rightarrow}} v \underset{R}{\overset{*}{\leftarrow}} y,$$

  et donc

$$x \underset{R}{\rightarrow} x' \underset{R}{\overset{*}{\rightarrow}} v \underset{R}{\overset{*}{\leftarrow}} y,$$

– Cas $\underset{R}{\leftarrow}$:

$$x \underset{R}{\leftarrow} x' \overset{n}{\underset{R}{\longleftrightarrow}} y$$

Par hypothèse de récurrence, il existe v tel que

$$x' \overset{*}{\underset{R}{\rightarrow}} v \overset{*}{\underset{R}{\leftarrow}} y,$$

et donc

$$x \underset{R}{\leftarrow} x' \overset{*}{\underset{R}{\rightarrow}} v \overset{*}{\underset{R}{\leftarrow}} y,$$

On a un pic entre x, x' et v, et par hypothèse, comme $\rightarrow_R$ est confluente, il existe v' tel que

$$x \overset{*}{\underset{R}{\rightarrow}} v' \overset{*}{\underset{R}{\leftarrow}} v,$$

et donc

$$x \overset{*}{\underset{R}{\rightarrow}} v' \overset{*}{\underset{R}{\leftarrow}} v \overset{*}{\underset{R}{\leftarrow}} t.$$

□

Par contre, la confluence et la confluence locale sont deux notions qui ne sont pas équivalentes. Considérons l'ensemble $\mathscr{A} = \{a, b, c, d\}$ et la relation

$$\{a \rightarrow b; a \rightarrow c; b \rightarrow a; b \rightarrow d\}$$

On a alors le graphe suivant :

$$\begin{tikzcd}
a \arrow[r, bend left] \arrow[d] & b \arrow[l, bend left] \arrow[d] \\
c & d
\end{tikzcd}$$

La relation est localement confluente, mais pas confluente :

$$\begin{tikzcd}
& a \arrow[dl, "R"'] \arrow[dr, "R"] & \\
b \arrow[d, dashed, "R"'] & & c \arrow[ddll, dashed, "R"] \\
a \arrow[d, dashed, "R"'] & & \\
c & &
\end{tikzcd}
\qquad
\begin{tikzcd}
& b \arrow[dl, "R"'] \arrow[dr, "R"] & \\
a \arrow[d, dashed, "R"'] & & d \arrow[ddll, dashed, "R"] \\
b \arrow[d, dashed, "R"'] & & \\
d & &
\end{tikzcd}
\qquad
\begin{tikzcd}
& a \arrow[dl, "R^*"'] \arrow[dr, "R^*"] & \\
c & & d
\end{tikzcd}$$

Définition 5.6 *Soient $\mathscr{A}$ un ensemble et $\rightarrow_R$ une relation sur $\mathscr{A}$. $\rightarrow_R$ est convergente si elle est confluente et $\leftarrow_R$ bien fondée*[1].

Remarque : encore un fois, on dira qu'un système de réécriture R est convergent si la relation associée $\rightarrow_R$ l'est.

[1]Voir les rappels du chapitre 2.

Lemme 5.7 *Soit $\mathscr{A}$ un ensemble et $\to_R$ une relation convergente sur $\mathscr{A}$. Alors tout élément x de $\mathscr{A}$ a une R-forme normale unique qui est notée $x\downarrow_R$.*

Démonstration Nous allons montrer ce lemme par induction (Théorème 2.4) sur la relation bien fondée $\leftarrow_R$. Soit x un élément de $\mathscr{A}$. Nous allons distinguer trois cas :

- Si x n'a pas de prédécesseurs pour $\leftarrow_R$, x a une unique forme normale qui est lui-même.
- Si x a deux prédécesseurs y et y' (éventuellement identiques) pour $\leftarrow_R$, par induction, y a une forme normale unique $y\downarrow$ et y' a une forme normale unique $y'\downarrow$:

  $$x \underset{R}{\to} y \overset{*}{\underset{R}{\to}} y\downarrow \quad x \underset{R}{\to} y' \overset{*}{\underset{R}{\to}} y'\downarrow$$

  x a donc au moins une forme normale, par exemple $y\downarrow$. En utilisant l'hypothèse que $\to_R$ est confluente, on obtient qu'il existe un élément v tel que

  $$y\downarrow \overset{*}{\underset{R}{\to}} v \overset{*}{\underset{R}{\leftarrow}} y'\downarrow$$

  Or $y\downarrow$ est en forme normale, donc il y en fait 0 pas entre $y\downarrow$ et v, et $y\downarrow$ est identique à v. De la même façon, $y'\downarrow$ est identique à v. Finalement $y\downarrow$ et $y'\downarrow$ sont identiques.

□

On a vu plus haut que la confluence et la confluence locale ne sont pas en général équivalentes. C'est cependant le cas pour les systèmes bien fondés, comme l'a montré Newman en 1942 [25] :

Théorème 5.8 (Lemme de Newman) *Soit $\mathscr{A}$ un ensemble et $\to_R$ une relation sur $\mathscr{A}$. Dans le cas où $\leftarrow_R$ est bien fondée, $\to_R$ est localement confluente si et seulement si $\to_R$ est confluente.*

Démonstration Il est clair que si $\to_R$ est confluente, elle est localement confluente. Il reste à montrer que si $\to_R$ est localement confluente, alors elle est confluente. Supposons donc que $\to_R$ est localement confluente, et montrons par induction bien fondée sur $\leftarrow_R$ que

$$\forall x \in \mathscr{A} \quad P(x)$$

avec

$$P(x) \quad \equiv \quad \forall y, z \in \mathscr{A} \quad ((y \overset{*}{\underset{R}{\leftarrow}} x \overset{*}{\underset{R}{\to}} z) \Rightarrow (\exists v \in \mathscr{A} \quad y \overset{*}{\underset{R}{\to}} v \overset{*}{\underset{R}{\leftarrow}} z))$$

Soit donc (x, y, z) un triplet d'éléments de $\mathscr{A}$ tels que

$$y \overset{*}{\underset{R}{\leftarrow}} x \overset{*}{\underset{R}{\to}} z$$

- Si x n'a pas de prédécesseurs pour $\leftarrow_R$ (*i.e.* pas de successeurs pour $\to_R$), $P(x)$ est trivialement vérifiée car nécessairement y et z sont égaux à x, et donc $v \equiv x$ convient.
- Supposons maintenant que x a des prédécesseurs pour $\leftarrow_R$, et que tous ces prédécesseurs vérifient P.
  - S'il n'y a aucun pas entre x et y, alors $y \equiv x$ est en relation par $\to_R$ avec z en un certain nombre de pas, et z en relation par $\to_R$ avec z en 0 pas : on peut prendre z en guise de v.

    $$\begin{tikzcd} y \arrow[r, phantom, "\equiv"] & x \arrow[dr, "R^*"] & \\ & & z \end{tikzcd} \qquad \begin{tikzcd} y \arrow[r, phantom, "\equiv"] \arrow[dr, dashed, "R^*"'] & x \arrow[dr, "R^*"] & \\ & z \arrow[r, phantom, "\equiv"] & z \end{tikzcd}$$

    Le raisonnement est similaire s'il n'y a aucun pas entre x et z.

– Supposons maintenant qu'il y a au moins un pas entre x et y et entre x et z : $x \to_R y' \to_R^* y$ et $x \to_R z' \to_R^* z$. Comme $\to_R$ est localement confluente, il existe un élément v' tel que $y' \to_R^* v'$ et $z' \to_R^* v'$. Maintenant, par induction sur y' et avec le couple (y, v') on sait qu'il existe un élément v'' tel que $y \to_R^* v''$ et $v' \to_R^* v''$. A nouveau par induction sur z' avec le couple (v'', z), on sait qu'il existe un élément v tel que $v'' \to_R^* v$ et $z \to_R^* v$, ce qui permet de conclure.

$$\begin{tikzcd}
 & & & x \arrow[dl, "R"'] \arrow[dr, "R"] & & \\
 & & y' \arrow[ddll, "R^*"'] \arrow[dr, dashed, "R^*"] & & z' \arrow[dl, dashed, "R^*"'] \arrow[ddr, "R^*"] & \\
 & & & v' \arrow[ddll, dashed, "R^*"] & & \\
y \arrow[dr, dashed, "R^*"'] & & & & & z \arrow[ddll, dashed, "R^*"] \\
 & v'' \arrow[drr, dashed, "R^*"'] & & & & \\
 & & & v & &
\end{tikzcd}$$

□

5.3 Paires critiques

Dans le cas où l'on étudie une relation de réécriture $\to_R$ sur une algèbre de termes, les paires critiques permettent d'identifier facilement les cas à tester pour vérifier la confluence locale :

Définition 5.9 *Soit $\mathcal{T}(\mathcal{F}, \mathcal{X})$ une algèbre de termes, et R un système de réécriture. Une paire critique est une paire de termes obtenue à partir de deux règles de R $\{l \to r; g \to d\}$ qui se* superposent *; c'est-à-dire qu'il existe un renommage ρ de l, une position $p \in \mathcal{P}os(l)$ telle que $l(p) \notin \mathcal{X}$, et $l\rho|_p$ et g sont unifiables. Soit σ l'unificateur principal de $l\rho|_p$ et l. La paire critique est alors égale à*

$$r\rho\sigma = (l\rho[d]_p)\sigma$$

Remarquons qu'une règle peut se superposer sur elle-même.

Exemple 5.10 *Soit $(\mathscr{S}, \mathcal{F}, \tau)$ la signature monosortée telle que*

$$\begin{array}{rcl}
\mathscr{S} & = & \{s\} \\
\mathcal{F} & = & \{e, I, \cdot\} \\
\tau(e) & = & s \\
\tau(I) & = & s \to s \\
\tau(\cdot) & = & s \times s \to s
\end{array}$$

Considérons la règle de réécriture $(x \cdot y) \cdot z \to x \cdot (y \cdot z)$. Elle se superpose sur elle-même à la position 1 *(nous utiliserons le renommage $\rho = \{x \mapsto x'; y \mapsto y'; z \mapsto z'\}$) : $x' \cdot y'$ s'unifie à $(x \cdot y) \cdot z$ avec pour unificateur principal $\{x' \mapsto x \cdot y; y' \mapsto z\}$. La paire critique est donc :*

$$(x \cdot y) \cdot (z \cdot z') = ((x \cdot y) \cdot z) \cdot z'$$

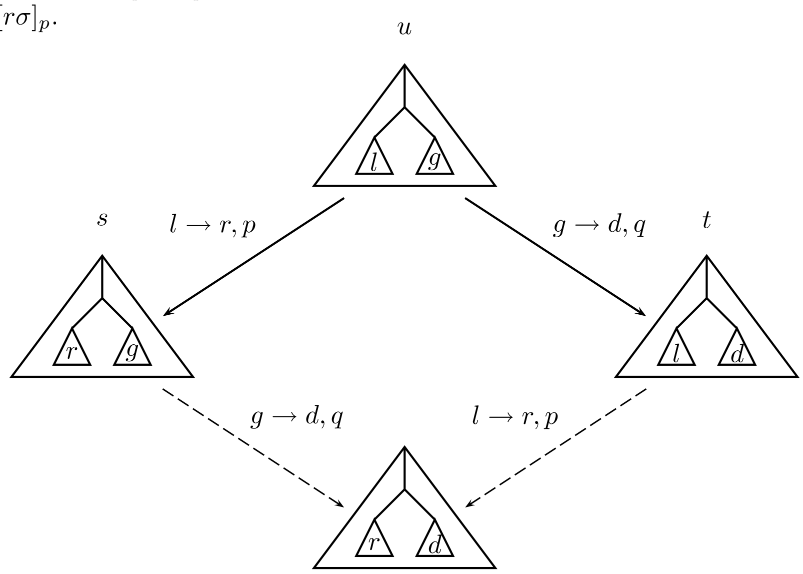
Considérons la règle $x \cdot e \to x$. Cette règle se superpose à la règle $(x \cdot y) \cdot z \to x \cdot (y \cdot z)$ à la racine : $x' \cdot e$ s'unifie à $(x \cdot y) \cdot z$ avec pour unificateur principal $\{x' \mapsto x \cdot y; z \mapsto e\}$. La paire critique est donc :

$$(x \cdot y) = x \cdot (y \cdot e)$$

Théorème 5.11 *Un système de réécriture est localement confluent si et seulement si toutes ses paires critiques sont localement confluentes.*

Démonstration Soit (u, s, t) un triplet de termes tels que $u \to^{p}_{l \to r, \sigma} s$ et $u \to^{q}_{g \to d, \tau} t$. Sans perte de généralité, nous supposerons que la règle $l \to r$ a déjà été renommée, c'est-à-dire que l'ensemble des variables de l et r est disjoint de celui des variables de g et d. Nous allons distinguer trois cas, les autres étant symétriques :

- Si les redexes sont disjoints, c'est-à-dire que p et q ne sont pas préfixes l'un de l'autre, les deux réécritures commutent, en effet, $s|_q = (u[r\sigma]_p)|_q = u|_q = g\sigma$, donc s peut se réécrire à la position q par la règle $g \to d$ en $u[r\sigma]_p[d\sigma]_q$, et de façon analogue, t peut se réécrire à la position p par la règle $l \to r$ en $u[d\sigma]_q[r\sigma]_p$.



- Si q est de la forme $pq'q''$ avec $l(q') = x \in \mathcal{X}$, comme dans le cas précédent, l'idée est d'appliquer la règle $g \to d$ à s et $l \to r$ à t. Le problème est que $t|_p = u[d\sigma]_{pq'q''}|_p = u|_p[d\sigma]_{q'q''} = l\sigma[d\sigma]_{q'q''}$ n'est plus une instance de l si x a plus d'une occurrence dans l. Il faut commencer par réécrire tous les autres sous-termes $x\sigma \equiv x\sigma[g\sigma]_{q''}$ de $l\sigma$ en $x\sigma[d\sigma]_{q''}$. Soit donc $\{q'_1, \ldots, q'_n\} \subseteq \mathcal{P}os(l)$ l'ensemble des positions de l distinctes de q' telles que $l(q'_i) = x$. Il est clair que

$$\begin{array}{rcl} t & \equiv & u[l\sigma]_p[d\sigma]_{pq'q''} \\ & \equiv & u[l\sigma]_p[d\sigma]_{pq'q''}[x\sigma]_{pq'_1} \ldots [x\sigma]_{pq'_n} \\ & \equiv & u[l\sigma]_p[d\sigma]_{pq'q''}[g\sigma]_{pq'_1q''} \ldots [g\sigma]_{pq'_nq''} \\ & \to^*_{g \to d} & u[l\sigma]_p[d\sigma]_{pq'q''}[d\sigma]_{pq'_1q''} \ldots [d\sigma]_{pq'_nq''} \end{array}$$

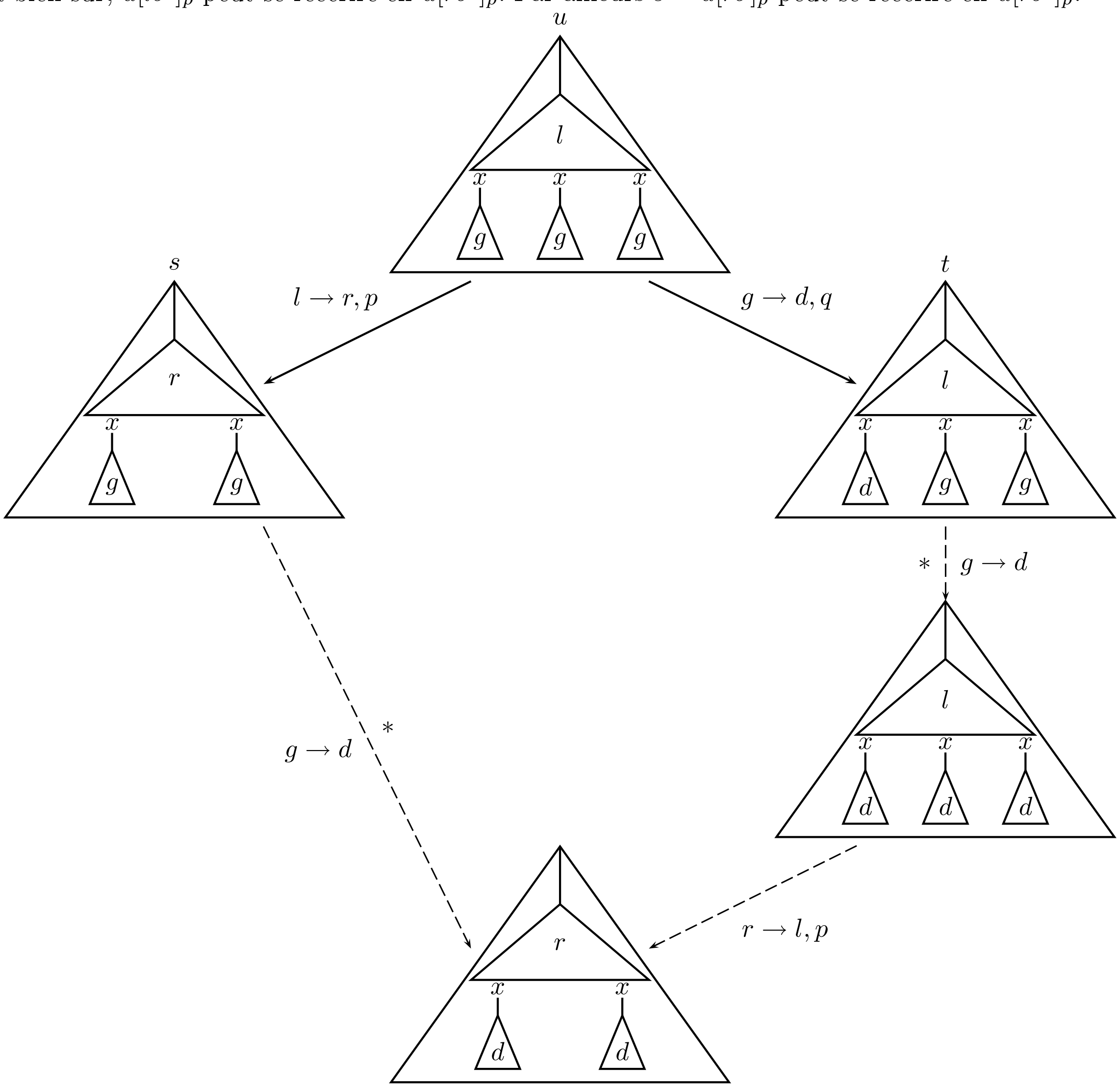
On note σ' la substitution dont le domaine est identique à celui de σ et telle que

$$\begin{array}{rcl} x\sigma' & = & x\sigma[d\sigma]_{q''} \\ y\sigma' & = & y\sigma \qquad \text{si } y \neq x. \end{array}$$

On obtient alors

$$\begin{array}{rcl} t & \rightarrow^*_{g\rightarrow d} & u[l\sigma]_p[d\sigma]_{pq'q''}[d\sigma]_{pq'_1q''}\ldots[d\sigma]_{pq'_nq''} \\ & \equiv & u[l\sigma]_p[x\sigma']_{pq'}[x\sigma']_{pq'_1}\ldots[x\sigma']_{pq'_n} \\ & \equiv & u[l\sigma']_p \end{array}$$

et bien sûr, $u[l\sigma']_p$ peut se réécrire en $u[r\sigma']_p$. Par ailleurs $s \equiv u[r\sigma]_p$ peut se réécrire en $u[r\sigma']_p$.



– Si q est de la forme pq' où q' est une position non variable de l ($l(q') \notin \mathcal{X}$), nous allons utiliser le fait que les paires critiques sont localement confluentes. Dans ce cas, $u|_p = l\sigma$ et $u|_q = d\sigma$. Or

$$\begin{array}{rcl} d\sigma & = & u|_q \\ & = & u|_{pq'} \\ & = & (u|_p)|_{q'} \\ & = & l\sigma|_{q'} \end{array}$$

Comme q' est une position de l, $l\sigma|_{q'} = l|_{q'}\sigma$. Donc

$$d\sigma = l|_{q'}\sigma$$

d et $l|_{q'}$ sont unifiables, l'ensemble de leurs unificateurs est non vide et admet un unificateur principal θ (voir le théorème 4.9). Par définition d'un unificateur principal, σ est de la forme $\theta\tau$. Par hypothèse, comme $r\theta = l\theta[d\theta]_{q'}$ est une paire critique, il existe un terme v tel que

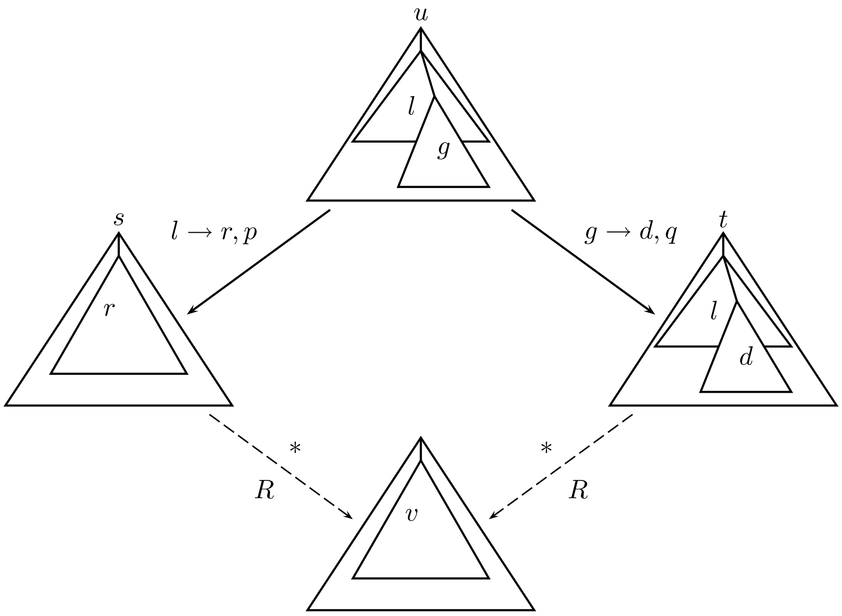
$$\begin{array}{rcl} r\theta & \to_R^* & v \\ l\theta[d\theta]_{q'} & \to_R^* & v \end{array}$$

En instanciant les deux séquences de réécriture ci-dessus par τ, on obtient

$$\begin{array}{rcl} r\sigma & \equiv & r\theta\tau \\ & \to_R^* & v\tau \\ (l[d]_{q'})\sigma & \equiv & (l[d]_{q'})\theta\tau \\ & \equiv & (l\theta[d\theta]_{q'})\tau \\ & \to_R^* & v\tau \end{array}$$

Finalement, on conclut que

$$\begin{array}{rcl} s & \equiv & u[r\sigma]_p \\ & \to_R^* & u[v\tau]_p \\ t & \equiv & u[(l[d]_{q'})\sigma]_p \\ & \to_R^* & u[v\tau]_p \end{array}$$



□

Corollaire 5.12 *Soit R un système de réécriture tel que la relation $\leftarrow_R$ est bien fondée. R est convergent si et seulement si pour toute paire critique $s = t$ de R, il existe $s\downarrow$ une forme normale de s et $t\downarrow$ une forme normale de t, telles que $s\downarrow \equiv t\downarrow$. C'est encore équivalent à pour toute paire critique $s = t$ de R, pour toute forme normale $s\downarrow$ de s et toute forme normale $t\downarrow$ de t, on a $s\downarrow \equiv t\downarrow$.*

5.4 Algèbre de formes normales

Dans le cas d'un système de réécriture R convergent (et fini), le problème du mot relatif à l'ensemble d'équations associé E_R est décidable :

Théorème 5.13 *Soit R un système de réécriture convergent. Deux termes s et t sont égaux modulo E_R si et seulement si leurs (uniques) formes normales respectives pour $\to_R$ sont identiques (dans le cas où R est un ensemble fini, on peut effectivement calculer ces formes normales).*

Démonstration Soient s et t deux termes. Il est clair que si leurs formes normales pour $\to_R$ sont identiques, s et t sont égaux modulo E_R. Supposons maintenant que s et t sont égaux modulo E_R : il existe une séquence d'étapes de $\to_R \cup \leftarrow_R$ entre s et t, et comme $\to_R$ est confluente, il existe un terme v tel que

$$\begin{tikzcd}
s \arrow[dr, "R^*"] & & t \arrow[dl, "R^*"'] \\
& v &
\end{tikzcd}$$

Soit par ailleurs une forme normale $s\downarrow_R$ de s :

$$\begin{tikzcd}
& s \arrow[dr, "R^*"] \arrow[ddl, "R^*"'] & & t \arrow[dl, "R^*"'] \\
& & v & \\
s\downarrow_R & & &
\end{tikzcd}$$

R est confluent, donc il existe une preuve par réécriture entre $s\downarrow_R$ et v, et comme $s\downarrow_R$ est en forme normale, v se récrit en $s\downarrow_R$:

$$\begin{tikzcd}
& s \arrow[dr, "R^*"] \arrow[ddl, "R^*"'] & & t \arrow[dl, "R^*"'] \\
& & v \arrow[dll, "R^*"'] & \\
s\downarrow_R & & &
\end{tikzcd}$$

On peut compléter le diagramme de façon symétrique pour n'importe quelle forme normale $t\downarrow_R$ de t :

$$\begin{tikzcd}
& s \arrow[dr, "R^*"] \arrow[ddl, "R^*"'] & & t \arrow[dl, "R^*"'] \arrow[ddr, "R^*"] & \\
& & v \arrow[dll, "R^*"'] \arrow[drr, "R^*"] & & \\
s\downarrow_R & & & & t\downarrow_R
\end{tikzcd}$$

R est confluent, donc il existe une preuve par réécriture entre $s\downarrow_R$ et $t\downarrow_R$, et comme ce sont deux termes en forme normale, ils sont en fait identiques :

$$\begin{tikzcd}
s \arrow[dr, "R^*"] \arrow[ddr, "R^*"'] & & t \arrow[dl, "R^*"'] \arrow[ddl, "R^*"] \\
& v \arrow[d, "R^*"'] & \\
& s\downarrow_R \equiv t\downarrow_R &
\end{tikzcd}$$

□

Le théorème ci-dessus a pour conséquence :

Théorème 5.14 *Soit $\mathcal{T}(\mathcal{F},\mathcal{X})$ une algèbre de termes, et R un système de réécriture convergent sur $\mathcal{T}(\mathcal{F},\mathcal{X})$. La $\mathcal{F}$-algèbre $\mathscr{A}$ définie par :*

- *Le support $\mathscr{A}_s$ est égal à l'ensemble des termes clos de sorte s qui sont en forme normale.*
- *Si $f : s_1 \times \ldots \times s_n \to s$ est un symbole de fonction de $\mathcal{F}$, alors l'interprétation de f dans $\mathscr{A}$ est égale à :*

$$f_{\mathscr{A}}(u_1\!\downarrow, \ldots, u_n\!\downarrow) = (f(u_1, \ldots, u_n))\!\downarrow$$

est isomorphe à $\mathcal{T}(\mathcal{F})/E_R$.

5.5 Systèmes de réécriture canoniques

Soit E un ensemble d'équations sur une algèbre de termes $\mathcal{T}(\mathcal{F},\mathcal{X})$, et soit R un système de réécriture tel que E_R et E définissent la même théorie équationnelle. Si R est convergent, on peut trouver un système de réécriture canonique associé à E. Pour cela, nous introduisons la notion de système interréduit.

Définition 5.15 *Soit $\mathcal{T}(\mathcal{F},\mathcal{X})$ une algèbre de termes, et R un système de réécriture sur $\mathcal{T}(\mathcal{F},\mathcal{X})$. R est dit interréduit si pour toute règle $l \to r$ de R*

- *l est en forme normale pour $R \setminus \{l \to r\}$,*
- *r est en forme normale pour R.*

Un système de réécriture est canonique si il est convergent et interréduit.

Théorème 5.16 *Soit $\mathcal{T}(\mathcal{F},\mathcal{X})$ une algèbre de termes et soient R et R' deux systèmes de réécriture tels que*

- *les deux relations $=_{E_R}$ et $=_{E_{R'}}$ sont égales.*
- *R et R' sont canoniques.*
- *$\longleftarrow_{R \cup R'}$ est bien fondée.*

Alors R et R' sont identiques à renommage près.

Démonstration Soit $l \to r$ une règle de R. Nous allons montrer que R' contient un renommage de cette règle. l et r sont égaux modulo E_R, donc modulo $E_{R'}$. Comme R' est convergent, il existe une preuve

$$\begin{tikzcd}
l \arrow[dr, "R'^*"] & & r \arrow[dl, "R'^*"'] \\
& v &
\end{tikzcd}$$

r et v sont égaux modulo $E_{R'}$, donc modulo E_R, comme R est convergent, il existe une preuve en V entre r et v :

$$\begin{tikzcd}
l \arrow[dr, "R'^*"] & & r \arrow[dl, "R'^*"'] \arrow[dr, "R^*"] & \\
& v \arrow[rr, "R^*"'] & & v'
\end{tikzcd}$$

Comme R est interréduit, r est en forme normale pour R, et donc r et v' sont identiques :

$$\begin{tikzcd}
l \arrow[dr, "R'^*"] & & r \arrow[dl, shift right, "R'^*"'] \\
& v \arrow[ur, shift right, "R^*"'] &
\end{tikzcd}$$

Or $\leftarrow_{R \cup R'}$ est bien fondée, donc r et v sont identiques :

$$\begin{tikzcd}
l \arrow[r, "R'^*"] & r
\end{tikzcd}$$

Comme $\leftarrow_R$ est bien fondée, l et r ne sont pas identiques, donc il existe au moins une étape de réécriture $\rightarrow_{R'}$ entre l et r. Soit $l' \rightarrow r'$ la règle utilisée lors de la première étape :

$$\begin{tikzcd}
l \arrow[rr, "R"] \arrow[dr, "l' \rightarrow r'"'] & & r \\
& {} \arrow[ur, "R'^*"'] &
\end{tikzcd}$$

On peut appliquer à $l' \rightarrow r'$ un raisonnement symétrique à celui fait sur $l \rightarrow r$, et on obtient :

$$\begin{tikzcd}
l' \arrow[rr, "R'"] \arrow[dr, "g \rightarrow d"'] & & r' \\
& {} \arrow[ur, "R^*"'] &
\end{tikzcd}$$

l contient une instance de l' qui lui-même contient une instance de g, l est donc réductible par la règle $g \rightarrow d$ de R. Comme R est interréduit, la seule possibilité est que $g \rightarrow d$ soit la règle $l \rightarrow r$ elle-même. l contient une instance de l' qui contient une instance de l : l' est un renommage de l par ρ :

$$\begin{tikzcd}
l\rho \arrow[rr, "R'"] \arrow[dr, "l \rightarrow r"'] & & r' \\
& r\rho \arrow[ur, "R^*"'] &
\end{tikzcd}$$

r est en forme normale pour R, $r\rho$ également, donc r' et $r\rho$ sont identiques, et la règle $l' \rightarrow r'$ de R' est un renommage de $l \rightarrow r$ par ρ.
□

5.6 Réécriture modulo

Le principe de la réécriture est d'utiliser des égalités pour faire du remplacement d'égaux par des égaux *de façon orientée*. Toutes les égalités ne peuvent cependant pas être orientées, le cas le plus simple et le plus classique étant la commutativité :

$$x + y = y + x,$$

où le membre droit et le membre gauche sont des renommages l'un de l'autre. Dès lors qu'un système de réécriture R contient un axiome de la forme ci-dessus, la relation $\leftarrow_R$ n'est pas bien fondée. Plusieurs solutions ont été proposées pour résoudre ce problème, toutes étant basées sur une partition du système de réécriture de départ en deux parties, l'une «orientable» et classiquement notée R, l'autre non-orientable et notée S. Le but ultime est encore de décider le problème du mot modulo $E_{R\cup S}$.

5.6.1 Réécriture dans les classes

Le principe de la réécriture dans les classes est d'appliquer la relation $\rightarrow_R$ directement dans les classes modulo $=_S$, ce qui conduit à la définition suivante introduite par Ballantine et Lankord [23] :

Définition 5.17 *Soient R un système de de réécriture et S un ensemble d'équations. Un terme s se récrit en t avec R modulo S si :*

$$\exists s',\ \exists t',\ s =_S s' \quad \wedge \quad t =_S t' \quad \wedge \quad s' \rightarrow_R t'$$

On le note $s \rightarrow_{R/S} t$.

Pour utiliser effectivement cette réécriture, il faut pouvoir énumérer la classe d'un terme modulo S, c'est-à-dire qu'elle soit finie. En outre, celà peut conduire à une combinatoire rédhibitoire. En pratique, on utilise une relation plus faible introduite par Peterson et Stickel [27].

5.6.2 Réécriture étendue

Définition 5.18 *Soit $\mathcal{T}(\mathcal{F}, \mathcal{X})$ une algèbre de termes, R un système de réécriture et S une congruence sur $\mathcal{T}(\mathcal{F}, \mathcal{X})$. La relation de réécriture S-étendue par R, notée $\rightarrow_{S\backslash R}$ est définie par :*

$$s \rightarrow_{S\backslash R} t \text{ si et seulement si } \exists l \rightarrow r \in R \quad \exists p \in \mathcal{P}os(s) \quad \exists \sigma \quad s|_p =_S l\sigma \quad \wedge \quad t \equiv s[r\sigma]_p$$

Certaines notions définies précédemment peuvent être reprises sans changement (formes normales), d'autres sont redefinies :

Définition 5.19 (Church-Rosser, confluence, confluence locale, cohérence, cohérence locale) *Soient $\mathscr{A}$ un ensemble, $\rightarrow_R$ et $\rightarrow_S$ deux relations sur $\mathscr{A}$, et $\rightarrow_{R^S}$ une relation sur $\mathscr{A}$ comprise entre $\rightarrow_R$ et $\rightarrow_{R/S}$.*

- *$\rightarrow_R$ est $\rightarrow_{R^S}$-Church-Rosser modulo $\rightarrow_S$ si et seulement si*

$$\forall x, y \in \mathscr{A}, \quad x \quad (\underset{R}{\longleftrightarrow} \cup \underset{S}{\longleftrightarrow})^* \quad y \quad \Rightarrow \quad (\exists z, z' \in \mathscr{A}, \quad x \ \underset{R^S}{\overset{*}{\rightarrow}} \ z \ \underset{S}{\overset{*}{\longleftrightarrow}} \ z' \ \underset{R^S}{\overset{*}{\leftarrow}} \ y).$$

– $\rightarrow_{R^S}$ *est confluente modulo* $\rightarrow_S$ *si et seulement si*

$$\forall x, y, z \in \mathscr{A}, \quad y \overset{*}{\underset{R^S}{\leftarrow}} x \overset{*}{\underset{R^S}{\rightarrow}} z \quad \Rightarrow \quad (\exists v, v' \in \mathscr{A}, \quad y \overset{*}{\underset{R^S}{\rightarrow}} v \overset{*}{\underset{S}{\longleftrightarrow}} v' \overset{*}{\underset{R^S}{\leftarrow}} z).$$

– $\rightarrow_{R^S}$ *est localement confluente avec* $\rightarrow_R$ *modulo* $\rightarrow_S$ *si et seulement si*

$$\forall x, y, z \in \mathscr{A}, \quad y \underset{R^S}{\leftarrow} x \underset{R}{\rightarrow} z \quad \Rightarrow \quad (\exists v, v' \in \mathscr{A}, \quad y \overset{*}{\underset{R^S}{\rightarrow}} v \overset{*}{\underset{S}{\longleftrightarrow}} v' \overset{*}{\underset{R^S}{\leftarrow}} z).$$

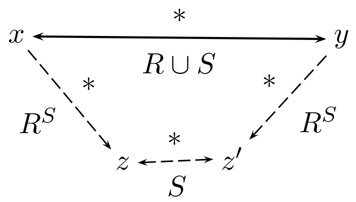
– $\rightarrow_{R^S}$ *est cohérente modulo* $\rightarrow_S$ *si et seulement si*

$$\forall x, y, z \in \mathscr{A}, \quad y \overset{*}{\underset{R^S}{\leftarrow}} x \overset{*}{\underset{S}{\longleftrightarrow}} z \quad \Rightarrow \quad (\exists v, v' \in \mathscr{A}, \quad y \overset{*}{\underset{R^S}{\rightarrow}} v \overset{*}{\underset{S}{\longleftrightarrow}} v' \overset{*}{\underset{R^S}{\leftarrow}} z).$$

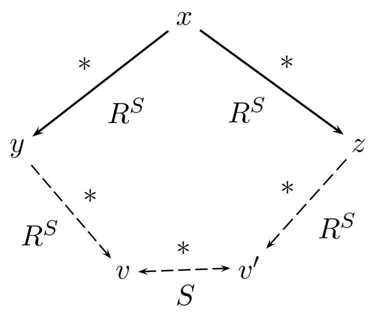
– $\rightarrow_{R^S}$ *est localement cohérente modulo* $\rightarrow_S$ *si et seulement si*

$$\forall x, y, z \in \mathscr{A}, \quad y \underset{R^S}{\leftarrow} x \underset{S}{\rightarrow} z \quad \Rightarrow \quad (\exists v, v' \in \mathscr{A}, \quad y \overset{*}{\underset{R^S}{\rightarrow}} v \overset{*}{\underset{S}{\longleftrightarrow}} v' \overset{*}{\underset{R^S}{\leftarrow}} z).$$

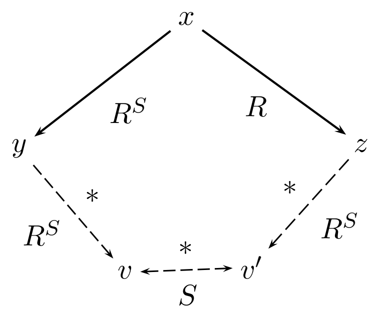
Vision graphique des définitions :



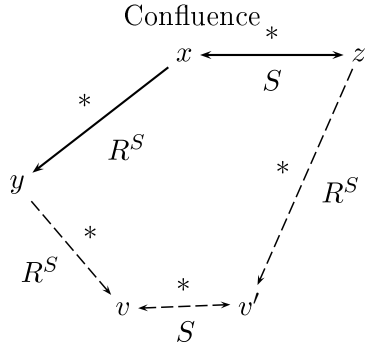
Church-Rosser



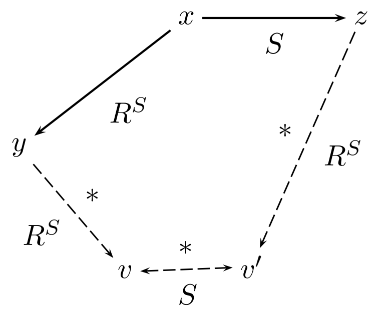
Confluence



Confluence locale



Cohérence



Cohérence locale

Le lemme de Newman 5.8 se généralise en le théorème suivant :

Théorème 5.20 (Jouannaud & Kirchner 1984) *Soient $\mathscr{A}$ un ensemble, $\rightarrow_R$, $\rightarrow_S$ et $\rightarrow_{R^S}$ trois relations sur $\mathscr{A}$, telles que $\rightarrow_{R^S}$ est comprise entre $\rightarrow_R$, et $\rightarrow_{R/S}$ et $\leftarrow_{R/S}$ est bien fondée. Les propriétés suivantes sont équivalentes :*

1. *$\rightarrow_R$ est $\rightarrow_{R^S}$-Church-Rosser modulo $\rightarrow_S$,*
2. *$\rightarrow_{R^S}$ est confluente et cohérente modulo $\rightarrow_S$,*
3. *$\rightarrow_{R^S}$ est localement confluente et localement cohérente avec $\rightarrow_R$ modulo $\rightarrow_S$,*
4. *pour tous éléments x et y de $\mathscr{A}$, $x \ \longleftrightarrow^*_{R\cup S}\ y$ si et seulement si pour toutes les formes normales x' de x et toutes les formes normales y' de y pour $\rightarrow_{R^S}$, $x' \ \longleftrightarrow^*_S\ y'$.*

Démonstration Il est évident que 1. $\Rightarrow$ 2. $\Rightarrow$ 3..
Montrons maintenant que 3. $\Rightarrow$ 4.
Il est encore clair que pour tous les éléments x et y de $\mathscr{A}$ tels que leurs formes normales pour $\rightarrow_{R^S}$ sont S-égales, $x \ \longleftrightarrow^*_{R\cup S}\ y$. Montrons maintenant que

$$\begin{gathered}\forall n,\ \forall\{x_1, x_2, \ldots, x_{n-1}, x_n\}\ \ x_1 \underset{R\cup S}{\longleftrightarrow} x_2 \underset{R\cup S}{\longleftrightarrow} \ldots \underset{R\cup S}{\longleftrightarrow} x_{n-1} \underset{R\cup S}{\longleftrightarrow} x_n \\ \Rightarrow \\ \forall x'_1, x'_n \quad x'_1 \text{ est une forme normale de } x_1 \wedge x'_n \text{ est une forme normale de } x_n \quad \Rightarrow \quad x'_1 \overset{*}{\underset{S}{\longleftrightarrow}} x'_n\end{gathered}$$

par induction bien fondée avec l'extension multiensemble de $\leftarrow_{R/S}$ sur $\{x_1, x_2, \ldots, x_{n-1}, x_n\}$.

1. Si $n = 1$, x_1 et x_n sont identiques, il faut montrer que toutes les formes normales de x_1 sont S-égales.
   (a) Si x_1 est en forme normale pour $\rightarrow_{R^S}$, la seule forme normale de x_1 est x_1 lui-même, la propriété voulue est bien démontrée.
   (b) Si x_1 n'est pas en forme normale pour $\rightarrow_{R^S}$, toute forme normale de x_1 est obtenue à partir de x_1 en au moins une étape. Soient donc deux réductions de longueur au moins 1 à partir de x_1 et conduisant à deux formes normales x'_1 et x''_1 arbitraires, où l'on identifie les termes obtenus après un pas de réduction :

$$\begin{tikzcd} & & x_1 \arrow[dl, "R^S"'] \arrow[dr, "R^S"] & & \\ & y \arrow[dl, "*"', "R^S"] & & z \arrow[dr, "*", "R^S"'] & \\ x'_1 & & & & x''_1 \end{tikzcd}$$

   $\rightarrow_{R^S}$ est comprise entre $\rightarrow_R$ et $\rightarrow_{R/S}$, donc $x_1 \ \rightarrow_{R/S}\ z$. Le pas $x_1 \ \rightarrow_{R/S}\ z$ peut se décomposer par définition en

$$x_1 \leftrightarrow_S x_2 \leftrightarrow_S \ldots \leftrightarrow_S x_{m-1} \leftrightarrow_S x_m \rightarrow_R z_1 \leftrightarrow^*_S z$$

   On va montrer par induction sur m la propriété suivante :

$$P(m) \ \equiv \ \begin{array}{l}\forall x, z\ x_1 \overset{*}{\underset{S}{\longleftrightarrow}} x \wedge x \overset{m}{\underset{S}{\longleftrightarrow}} \underset{R}{\rightarrow} \overset{*}{\underset{S}{\longleftrightarrow}} z \quad \Rightarrow \\ \forall x', z' \quad x' \text{ est une forme normale de } x \wedge z' \text{ est une forme normale de } z \quad \Rightarrow \\ x' \overset{*}{\underset{S}{\longleftrightarrow}} z'\end{array}$$

i. Si $m = 0$, on est dans la situation suivante :

$$
\begin{tikzcd}
x \arrow[dr, "R"] & \\
& {} \arrow[r, leftrightarrow, "*", "S"'] & z
\end{tikzcd}
$$

Soient x' une forme normale de x et z' une forme normale de z, remarquons que comme x se réduit par $\to_R$, il n'est pas en forme normale pour $\to_{R^S}$, et qu'il y a donc au moins un pas entre x et x' :

$$
\begin{tikzcd}
& & x \arrow[dll, "R^S"'] \arrow[dr, "R"] & & \\
x'' \arrow[d, "R^S"', "*"] & & & {} \arrow[r, leftrightarrow, "*", "S"'] & z \arrow[d, "R^S", "*"'] \\
x' & & & & z'
\end{tikzcd}
$$

Grâce à la convergence locale, on peut clore en partie le diagramme :

$$
\begin{tikzcd}
& & x \arrow[dll, "R^S"'] \arrow[drr, "R"] & & & \\
x'' \arrow[d, "R^S"', "*"] \arrow[dr, dashed, "*", "R^S"'] & & & & {} \arrow[r, leftrightarrow, "*", "S"'] \arrow[dl, dashed, "*"', "R^S"] & z \arrow[d, "R^S", "*"'] \\
x' & v \arrow[rr, dashed, leftrightarrow, "*", "S"'] & & v' & & z'
\end{tikzcd}
$$

La partie basse du digramme entre x'' et z

$$
\begin{tikzcd}
x'' \arrow[dr, dashed, "*", "R^S"'] & & & {} \arrow[r, leftrightarrow, "*", "S"'] \arrow[dl, dashed, "*"', "R^S"] & z \\
& v \arrow[r, dashed, leftrightarrow, "*", "S"'] & v' & &
\end{tikzcd}
$$

peut être expansée en une séquence ne comprenant que des pas $\to_R$ et $\to_S$ entre des éléments strictement plus petits que x pour $\leftarrow^{+}_{R/S}$, on peut donc appliquer l'hypothèse d'induction la plus externe et en conclure que toutes les formes normales de x' et z, en particulier x' et

z' sont S-égales :

$$
\begin{tikzcd}
 & & & x \arrow[dll, "R^S"'] \arrow[drr, "R"] & & & & \\
 & x'' \arrow[ddl, "R^S"', "*"] \arrow[dr, dashed, "*", "R^S"'] & & & & {} \arrow[r, leftrightarrow, "*", "S"'] \arrow[dl, dashed, "*"', "R^S"] & z \arrow[ddr, "*"', "R^S"] & \\
 & & v \arrow[rr, dashed, leftrightarrow, "*", "S"'] & & v' & & & \\
x' \arrow[rrrrrrr, dashed, leftrightarrow, "S", "*"'] & & & & & & & z'
\end{tikzcd}
$$

ii. Si $m \geq 1$, on a le digramme suivant :

$$
\begin{tikzcd}
x \arrow[r, leftrightarrow, "S"'] & x_2 \arrow[r, leftrightarrow, "m-1", "S"'] & x_m \arrow[dr, "R"] & & \\
 & & & {} \arrow[r, leftrightarrow, "*", "S"'] & z
\end{tikzcd}
$$

x_2 n'est pas en forme normale, sinon, en appliquant l'hypothèse d'induction $P(m-1)$ avec x_2 et z, on peut conclure que x_2 est S-égal à une forme normale de z, et donc que $\rightarrow_{R/S}$ est cyclique, ce qui contredit la bonne fondation de $\leftarrow_{R/S}$.
De même, x n'est pas en forme normale, car sinon, en appliquant la cohérence locale entre x, x_2 et un réduit de x_2 par $\rightarrow_{R^S}$, on aurait encore un cycle pour $\rightarrow_{R/S}$, ce qui est impossible.
Soient x' une forme normale de x et z' une forme normale de z. D'après les remarques ci-dessus, il y a au moins un pas entre x et x' :

$$
\begin{tikzcd}
 & & x \arrow[dl, "R^S"'] \arrow[r, leftrightarrow, "S"'] & x_2 \arrow[r, leftrightarrow, "m-1", "S"'] & x_m \arrow[dr, "R"] & & & \\
 & x'' \arrow[dl, "R^S"', "*"] & & & & {} \arrow[r, leftrightarrow, "*", "S"'] & z \arrow[dr, "*"', "R^S"] & \\
x' & & & & & & & z'
\end{tikzcd}
$$

Appliquons la cohérence locale entre x'', x et x_2. Comme indiqué ci-dessus, x_2 n'est pas en forme normale, donc on a le diagramme suivant :

$$
\begin{tikzcd}
 & & x \arrow[dl, "R^S"'] \arrow[rr, leftrightarrow, "S"'] & & x_2 \arrow[d, dashed, "R^S"'] \arrow[r, leftrightarrow, "m-1", "S"'] & x_m \arrow[dr, "R"] & & & \\
 & x'' \arrow[ddl, "R^S"', "*"] \arrow[dr, dashed, "*", "R^S"'] & & & x_2'' \arrow[dl, dashed, "*"', "R^S"] & & {} \arrow[r, leftrightarrow, "*", "S"'] & z \arrow[ddr, "*"', "R^S"] & \\
 & & v \arrow[r, dashed, leftrightarrow, "*", "S"'] & v' & & & & & \\
x' & & & & & & & & z'
\end{tikzcd}
$$

Soit v'' une forme normale de v', en appliquant l'hypothèse d'induction $P(m-1)$ entre x_2 et z, on peut refermer partiellement le diagramme :

$$\begin{tikzcd}
 & & x \arrow[rr, leftrightarrow, "S"'] \arrow[dl, "R^S"'] & & x_2 \arrow[r, leftrightarrow, "m-1", "S"'] \arrow[d, dashed, "R^S"'] & x_m \arrow[dr, "R"] & & & \\
 & x'' \arrow[ddl, "R^S"', "*"] \arrow[dr, dashed, "*", "R^S"'] & & & x_2'' \arrow[dl, dashed, "*"', "R^S"] & & {} \arrow[r, leftrightarrow, "*", "S"'] & z \arrow[ddr, "*"', "R^S"] & \\
 & & v \arrow[r, dashed, leftrightarrow, "*", "S"'] & v' \arrow[dr, dashed, "*", "R^S"'] & & & & & \\
x' & & & & v'' \arrow[rrrr, dashed, leftrightarrow, "*", "S"'] & & & & z'
\end{tikzcd}$$

Maintenant, on peut appliquer l'hypothèse d'induction sur la séquence obtenue en expansant la séquence $x'' \rightarrow^*_{R^S} v \longleftrightarrow^*_S v''$ en des étapes $\longleftrightarrow_R \cup \longleftrightarrow_S$. En effet tous les éléments qui y apparaîssent sont plus petits que x_1 pour la relation bien fondée $\leftarrow^+_{R/S}$. On peut donc conclure grâce au diagramme :

$$\begin{tikzcd}
 & & x \arrow[rr, leftrightarrow, "S"'] \arrow[dl, "R^S"'] & & x_2 \arrow[r, leftrightarrow, "m-1", "S"'] \arrow[d, dashed, "R^S"'] & x_m \arrow[dr, "R"] & & & \\
 & x'' \arrow[ddl, "R^S"', "*"] \arrow[dr, dashed, "*", "R^S"'] & & & x_2'' \arrow[dl, dashed, "*"', "R^S"] & & {} \arrow[r, leftrightarrow, "*", "S"'] & z \arrow[ddr, "*"', "R^S"] & \\
 & & v \arrow[r, dashed, leftrightarrow, "*", "S"'] & v' \arrow[dr, dashed, "*", "R^S"'] & & & & & \\
x' \arrow[rrrr, dashed, leftrightarrow, "*", "S"'] & & & & v'' \arrow[rrrr, dashed, leftrightarrow, "*", "S"'] & & & & z'
\end{tikzcd}$$

2. Si $n \geq 2$, on va traiter 3 cas, suivant la nature du premier pas entre x_1 et x_2 :

   (a) Si $x_1 \longleftrightarrow_S x_2$, il faut encore distinguer 2 cas :

      i. Si x_1 est en forme normale, comme $\leftarrow R/S$ est bien fondée, et à cause de la cohérence locale, x_2 est également en forme normale. Soit x'_n une forme normale quelconque de x_n. On peut appliquer l'hypothèse d'induction sur $\{x_2, x_3, \ldots, x_n\}$, et obtenir le diagramme :

$$\begin{tikzcd}
 & x_1 \arrow[r, leftrightarrow, "S"'] \arrow[dl, "0", "R^S"'] & x_2 \arrow[r, leftrightarrow, "n-1", "R \cup S"'] \arrow[d, "0"', "R^S"] & x_n \arrow[dr, "R^S", "*"'] & \\
x_1 \arrow[rr, leftrightarrow, "S"'] & & x_2 \arrow[rr, dashed, leftrightarrow, "*", "S"'] & & x'_n
\end{tikzcd}$$

      ii. Si x_1 n'est pas en forme normale, soient x'_1 et x'_n des formes normales quelconques respectivement de x_1 et x_n. En mettant en évidence le premier pas de réduction à partir de x_1, on obtient :

$$
\begin{tikzcd}
& x_1 \arrow[dl, "R^S"'] \arrow[r, leftrightarrow, "S"'] & x_2 \arrow[r, leftrightarrow, "n-1", "R \cup S"'] & x_n \arrow[dd, "*"', "R^S"] \\
x_1'' \arrow[d, "R^S"', "*"] & & & \\
x_1' & & & x_n'
\end{tikzcd}
$$

Par cohérence locale, on referme partiellement le diagramme entre x_1'', x_1 et x_2. Par le même raisonnement que ci-dessus, il y a au moins un pas de $\rightarrow_{R^S}$ à partir de x_2, et on le met en évidence :

$$
\begin{tikzcd}
& x_1 \arrow[dl, "R^S"'] \arrow[rr, leftrightarrow, "S"'] & & x_2 \arrow[r, leftrightarrow, "n-1", "R \cup S"'] \arrow[d, dashed, "R^S"'] & x_n \arrow[ddd, "*"', "R^S"] \\
x_1'' \arrow[dd, "R^S"', "*"] \arrow[dr, dashed, "*", "R^S"'] & & & x_2'' \arrow[dl, dashed, "*"', "R^S"] & \\
& v \arrow[r, leftrightarrow, dashed, "*", "S"'] & v' & & \\
x_1' & & & & x_n'
\end{tikzcd}
$$

Soit v'' une forme normale quelconque de v', on peut appliquer l'hypothèse d'induction avec le multiensemble $\{x_2, \ldots, x_n\}$ qui est strictement plus petit que $\{x_1, x_2, \ldots, x_n\}$, v'' et x_n' sont donc S-égaux :

$$
\begin{tikzcd}
& x_1 \arrow[dl, "R^S"'] \arrow[rr, leftrightarrow, "S"'] & & x_2 \arrow[r, leftrightarrow, "n-1", "R \cup S"'] \arrow[d, dashed, "R^S"'] & x_n \arrow[ddd, "*"', "R^S"] \\
x_1'' \arrow[dd, "R^S"', "*"] \arrow[dr, dashed, "*", "R^S"'] & & & x_2'' \arrow[dl, dashed, "*"', "R^S"] & \\
& v \arrow[r, leftrightarrow, dashed, "*", "S"'] & v' \arrow[dr, dashed, "*", "R^S"'] & & \\
x_1' & & & v'' \arrow[r, leftrightarrow, dashed, "S", "*"'] & x_n'
\end{tikzcd}
$$

Maintenant, on peut appliquer l'hypothèse d'induction sur la séquence obtenue en expansant la séquence $x_1'' \rightarrow^*_{R^S} v \longleftrightarrow^*_S v''$ en des étapes $\longleftrightarrow_R \cup \longleftrightarrow_S$. En effet tous les éléments qui y apparaîssent sont plus petits que x_1 pour la relation bien fondée $\leftarrow^+_{R/S}$. On peut donc conclure grâce au diagramme :

$$\begin{tikzcd}
 & x_1 \arrow[rr, leftrightarrow, "S"'] \arrow[d, "R^S"'] & & x_2 \arrow[r, leftrightarrow, "n-1", "R \cup S"'] \arrow[d, dashed, "R^S"'] & x_n \arrow[ddd, "R^S", "*"'] \\
 & x_1'' \arrow[ddl, "R^S"', "*"] \arrow[dr, dashed, "*", "R^S"'] & & x_2'' \arrow[d, dashed, "*"', "R^S"] & \\
 & & v \arrow[r, dashed, leftrightarrow, "*", "S"'] & v' \arrow[d, dashed, "*"] & \\
x_1' \arrow[rrr, dashed, leftrightarrow, "S", "*"'] & & & v'' \arrow[r, dashed, leftrightarrow, "S", "*"'] & x_n'
\end{tikzcd}$$

(b) Si $x_1 \to_R x_2$, par hypothèse d'induction sur $\{x_1\}$, on sait que toutes les formes normales de x_1 sont S-égales, et donc en particulier, toutes les formes normales de x_1 et toutes les formes normales de x_2 sont S-égales. Par hypothèse d'induction sur $\{x_2, \ldots, x_n\}$, toutes les formes normales de x_2 et toutes les formes normales de x_n sont S-égales. On conclut par la transitivité de la S-égalité.

(c) Si $x_1 \leftarrow_R x_2$, par hypothèse d'induction sur $\{x_2, \ldots, x_n\}$, toutes les formes normales de x_2, en particulier toutes les formes normales de x_1, et toutes les formes normales de x_n sont S-égales.

Il est finalement évident que 4. $\Rightarrow$ 1. en prenant les formes normales pour clore le diagramme.

□

5.6.3 Réécriture AC-étendue

Un cas important en pratique est le cas où l'on a des symboles associatifs-commutatifs qui définissent une relation $\longleftrightarrow_S$ qui est la théorie équationnelle définie par

$$\begin{array}{cc} x + y = y + x & C \\ (x+y)+z = x+(y+z) & AC \end{array}$$

Dans ce cas particulier, la cohérence locale avec la commutativité est toujours acquise. Si l'étape (C) a lieu strictement au dessus de $\to_{AC\backslash R}$, $\to_{AC\backslash R}$ a lieu en dessous d'une position variable de (C), et donc on a une commutation évidente, comme dans le cas du lemme des paires critiques. Si l'étape (C) a lieu en dessous, elle est prise en charge par le filtrage modulo AC avant la réécriture par $\to_R$.

La cohérence locale avec l'associativité est assurée dès lors que la réécriture $\to_{AC\backslash R}$ a lieu dans les variables de (A), ou que (A) a lieu en dessous de $\to_{AC\backslash R}$. Le seul cas problématique est quand le membre gauche de l de la règle de R se superpose à A à la position 1 $l \equiv l_1 + l_2$. Dans ce cas,

$$r + z \leftarrow_{AC\backslash R} (l_1 + l_2) + z \longleftrightarrow_{AC} l_1 + (l_2 + z)$$

La solution est d'ajouter à R la règle $l_1 + (l_2 + z) \to r + z$, appelée extension de la règle $l_1 + l_2 \to r$. Il n'est pas necessaire d'étendre les extensions, car l'extension d'une extension est une instance modulo (AC) de l'extension de niveau 1.

Chapitre 6

Terminaison

Nous avons vu au chapitre précédent comment s'assurer de la convergence d'un système de réécriture R en vérifiant la confluence locale des paires critiques dans le cas où la relation inverse $\leftarrow_R$ est bien fondée. Les problèmes qui restent à traiter sont donc

– d'une part le calcul des paires critiques, c'est-à-dire l'unification,
– et d'autre part, la bonne fondation de $\leftarrow_R$, souvent appelée *la terminaison de R*.

Le problème de savoir si un système de réécriture quelconque termine est indécidable. La première démonstration est due à Huet et Lankford en 1978, qui codent un problème indécidable avec un nombre non borné de règles de réécriture [18]. Dershowitz a ensuite montré comment utiliser seulement un nombre fixé (petit) de règles [10]. Enfin Dauchet proposé un codage pour une machine de Turing arbitraire avec une seule règle, linéaire gauche et sans superposition sur elle-même [9].

Une façon simple de montrer que la terminaison d'un système de réécriture est indécidable est de le réduire au problème de correspondance de Post. Une instance du problème de correspondance de Post est donnée par un alphabet à n lettres $\{a_1, \ldots, a_n\}$, et un ensemble de couples de mots sur cet alphabet $\{(u_1, v_1), \ldots, (u_p, v_p)\}$. Le problème a une solution si il existe une suite finie d'entiers entre 1 et p, $(j_k)_k$ telle que

$$u_{j_1} u_{j_2} \ldots = v_{j_1} v_{j_2} \ldots$$

Cette instance est codée grâce à un système de réécriture défini sur la signature $\{f, \epsilon, a_1, \ldots, a_n\}$. Le symbole f est ternaire, le symbole ϵ est unaire, et les symboles a_i sont unaires.

Pour chaque couple (u_j, v_j), on définit une règle

$$f(u_j(x), v_j(y), z) \rightarrow f(x, y, z)$$

L'ensemble de ces p règles constituent un système de réécriture R_1.

En outre pour chaque symbole a_i, on définit

$$f(\epsilon, \epsilon, a_i(y)) \rightarrow f(a_i(y), a_i(y), a_i(y))$$

Cet ensemble de n règles est appelé R_2.

Le système de réécriture $R_1 \cup R_2$ ne termine pas si et seulement si le problème de correspondance de Post a une solution.

– Si le problème de Post a une solution, il existe un mot w qui peut s'écrire $u_{j_1} u_{j_2} \ldots u_{j_q}$ et $v_{j_1} v_{j_2} \ldots v_{j_q}$, donc

$$f(\epsilon, \epsilon, w) \underset{R_2}{\rightarrow} f(w, w, w) \overset{*}{\underset{R_1}{\rightarrow}} f(\epsilon, \epsilon, w)$$

- Si $R_1 \cup R_2$ ne termine pas, il y a au moins un terme t départ d'une réduction infinie. On choisit parmi tous les termes «non normalisants» un terme t minimal, c'est-à-dire que tous ses sous-termes ne peuvent donner lieu qu'à des réductions finies. Ni R_1 ni R_2 n'augmentent la profondeur des termes qu'ils réduisent, et comme l'arité des symboles est fixée, l'ensemble des termes présents dans la réduction est fini. Il y a un nombre infini d'étapes de R_2, car R_1 seul termine (par décroissance sur les tailles), et comme t a été choisi minimal, il y a un nombre infini d'étapes de R_2 en tête.
  On a donc un schéma de la forme
  $$t \equiv f(\epsilon, \epsilon, w_1) \underset{R_2}{\rightarrow} f(w_1, w_1, w_1) \underset{R_1 \cup R_2}{\overset{*}{\rightarrow}} f(\epsilon, \epsilon, w_2) \underset{R_2}{\rightarrow} f(w_2, w_2, w_2)$$
  sans étapes R_2 en tête entre $f(w_1, w_1, w_1)$ et $f(\epsilon, \epsilon, w_2)$.
  La suite w_i est stationnaire par minimalité de t, on peut donc supposer (éventuellement en tronquant la séquence au début) qu'elle est constante et égale à w :
  $$t \equiv f(\epsilon, \epsilon, w) \underset{R_2}{\rightarrow} f(w, w, w) \underset{R_1 \cup R_2}{\overset{*}{\rightarrow}} f(\epsilon, \epsilon, w) \underset{R_2}{\rightarrow} f(w, w, w)$$
  Comme t est minimal, w est fortement normalisant, et il n'y donc pas de réécriture en dessous de la position 3 entre $f(w, w, w)$ et $f(\epsilon, \epsilon, w)$.
  Ni R_1 ni R_2 ne change les symboles de tête des termes qu'ils réduisent, donc il n'y a aucune étape en dessous des positions 1 et 2 $f(w, w, w)$ et $f(\epsilon, \epsilon, w)$.
  La seule possibilité pour réduire $f(w, w, w)$ en $f(\epsilon, \epsilon, w)$ est d'utiliser une suite d'étapes de R_1 en tête, ce qui signifie que w peut s'écrire de deux façons différentes comme
  $$u_{j_1} u_{j_2} \dots u_{j_q} = w = v_{j_1} v_{j_2} \dots v_{j_q}$$

Cependant, il existe des techniques qui s'appliquent à certains systèmes et permettent pour ces cas particuliers de prouver la terminaison. Le cœur d'une preuve de terminaison est en général composé d'un ou plusieurs ordres bien fondés, c'est pourquoi nous commencerons par traiter le problème de la terminaison et des ordres, ces derniers étant utilisés dans le chapitre sur l'unification.

Les stratégies usuelles des outils automatiques pour prouver la terminaison sont basées sur la succession de différentes phases

- la transformation du système de réécriture R en un autre R' dont la terminaison entraîne celle du premier ,
- l'examen de la structure du système (*e.g.* hiérarchique, union disjointe)
- la recherche d'un ordre bien fondé contenant la relation de réécriture.

En fait, la plupart de ces outils utilisent une notion de terminaison plus forte que la terminaison usuelle, la C_ϵ-terminaison[1]. La C_ϵ-terminaison a été introduite par Gramlich [15] et étudiée par Ohlebush [26] et Urbain [30] ; elle présente l'avantage d'être modulaire contrairement à la terminaison usuelle qui ne l'est pas, comme le montre le contre-exemple classique de Toyama :

Contre-exemple 6.1 *Considérons les deux systèmes de réécriture qui portent sur des symboles de fonction disjoints :*
$$R_1 = \{f(a, b, x) \rightarrow f(x, x, x)\}$$
$$R_2 = \{g(x, y) \rightarrow x, g(x, y) \rightarrow y\}$$
R_2 termine trivialement. On peut montrer que R_1 termine par recurence sur la taille des termes.

[1]c'est-à-dire «Collapse Extended Termination».

1. *si t est de taille 1, c'est une variable ou une constante, il ne contient aucune instance de la règle de R_1, donc toutes les derivations issues de t par R_1 (*i.e.* la séquence vide) sont finies.*
2. *si t est de la forme $s(t_1, \ldots, t_n)$, avec $n \geq 1$, et $s \neq f$, toute derivation dans R_1 à partir de t se fait uniquement dans les sous-termes t_i. On peut projeter la dérivation sur les différents t_i afin d'obtenir n dérivations, une pour chacun des t_i. L'étape*

$$s(u_1, \ldots, u_{j-1}, u_j, u_{j+1} \ldots, u_n) \rightarrow_{j \cdot p} s(u_1, \ldots, u_{j-1}, u'_j, u_{j+1} \ldots, u_n)$$

se projette en $u_j \rightarrow_p u'_j$ pour $i = j$, et en la séquence vide pour $i \neq j$. Par hypothèse de récurrence sur les t_i, les dérivations projetées sont de de longueur finie, donc la dérivation de départ également.

3. *si t est de la forme $f(a, b, t_3)$, toute dérivation dans R_1 à partir de t se fait uniquement dans t_3 (et est donc de longueur finie) ou est de la forme :*

$$f(a, b, t_3) \rightarrow^*_{\neq \Lambda} f(a, b, t'_3) \rightarrow_\Lambda f(t'_3, t'_3, t'_3) \rightarrow^* \ldots$$

À partir de $f(t'_3, t'_3, t'_3)$, il ne peut plus y avoir de réécriture en tête, car t'_3 ne peut pas se réécrire à la fois en a et b ; en effet $\rightarrow_{R_1}$ ne change pas le symbole de tête. La séquence $f(t'_3, t'_3, t'_3) \rightarrow^* \ldots$ est en fait $f(t'_3, t'_3, t'_3) \rightarrow^*_{\neq \Lambda} \ldots$, et on conclut comme dans le cas précédent par hypothèse de récurrence sur t_3

Si on combine R_1 et R_2, l'ensemble ne termine plus, comme le montre la boucle

$$f(a, b, g(a, b)) \rightarrow_{R_1} f(g(a, b), g(a, b), g(a, b)) \rightarrow_{R_2} f(a, g(a, b), g(a, b)) \rightarrow_{R_2} f(a, b, g(a, b))$$

Dans ce cours, nous n'aborderons pas la première étape. La deuxième étape sera étudiée brièvement essentiellement par le biais de la C_ϵ terminaison. La dernière étape, qui consiste à prouver la terminaison d'un système de façon monolithique sera traitée en utilisant des ordres bien fondés.

6.1 Terminaison monolithique

6.1.1 Méthode de Manna et Ness

Il est clair qu'il suffit de prouver l'inclusion d'une relation de réécriture dans un ordre bien fondé pour prouver que la réécriture termine.

La méthode dite de Manna et Ness (mais due à Lankford) consiste à plonger la relation de réécriture dans un pré-ordre dit *de réécriture*. Il suffit alors de tester uniquement les règles du système, et non pas toutes les paires de termes dont le premier se réécrit en le deuxième.

Définition 6.2 *Un pré-ordre $\preceq$ est une relation réflexive et transitive. La relation symétrique $\succeq$ associée à $\preceq$ est définie par*

$$x \succeq y \text{ si et seulement si } y \preceq x$$

La relation d'équivalence $\simeq$ associée à $\preceq$ est égale à $\preceq \cap \succeq$ et l'ordre strict $\prec$ associé à $\preceq$ est égal à $\preceq \setminus \simeq$.

Définition 6.3 *Soit $\mathcal{T}(\mathcal{F}, \mathcal{X})$ une algèbre de termes et $\mathscr{R}$ une relation sur cette algèbre.*

– *$\mathscr{R}$ est monotone si c'est une précongruence.*
– *$\mathscr{R}$ est stable (par substitution) si pour tous termes s et t et pour toute substitution σ, on a*

$$s\mathscr{R}t \quad \Rightarrow \quad s\sigma\mathscr{R}t\sigma$$

Définition 6.4 *Soit $\mathcal{T}(\mathcal{F},\mathcal{X})$ une algèbre de termes, et $\preceq$ un pré-ordre sur cette algèbre. $\preceq$ est un pré-ordre de réécriture si*
– *$\prec$ et $\simeq$ sont monotones,*
– *$\prec$ et $\simeq$ sont stables,*
– *$\prec$ est bien fondé.*

Théorème 6.5 *Soit $R = \{l_i \to r_i\}_{i=1,\ldots,n}$, et $\succeq$ un pré-ordre de réécriture. Si $r_i \prec l_i$ pour $i = 1,\ldots,n$, alors R termine.*

Démonstration Si $s \leftarrow_R t$, il existe une règle $l_i \to r_i$ de R qui a permis de faire cette réécriture : $t = t[l_i\sigma]_p$ et $s = t[r_i\sigma]_p$. Par hypothèse, $r_i \prec l_i$. Comme $\preceq$ est un pré-ordre de réécriture,
– $\prec$ est stable, donc $r_i\sigma \prec l_i\sigma$
– et $\prec$ est monotone, donc $s = t[r_i\sigma]_p \prec t = t[l_i\sigma]_p$.
En conclusion, $\leftarrow_R$ est incluse dans $\prec$, $\prec$ étant bien fondé, $\leftarrow_R$ l'est donc aussi.
□

6.1.2 Paires de dépendance

Cette méthode due à Arts et Giesel [2] est plus «fine» que la méthode de Mana et Ness en ce sens qu'elle permet d'utiliser des ordres plus faibles, et pour lesquels les règles doivent seulement décroître au sens large. Elle est basée sur l'étude des séquences de réécriture infinies, et la mise en évidence de séquences particulières contenant un nombre infini de «paires de dépendance».

Définition 6.6 (Paires de dépendance) *Soit R un système de réécriture défini sur l'algèbre de termes $\mathcal{T}(\mathcal{F},\mathcal{X})$. Les symboles de $\mathcal{F}$ sont partitionnés en deux ensembles :*
– *$\mathcal{D}$ l'ensemble des* symboles définis *est égal à*

$$\mathcal{D} = \{f \in \mathcal{F} \ \mid \ \exists\, l \to r \in R \ \ l(\Lambda) = f\}$$

– *$\mathcal{C}$ l'ensemble des* constructeurs *est égal à $\mathcal{F} \setminus \mathcal{D}$.*
Une paire de dépendance *(u, v) de R est formée à partir d'une règle $l \to r$ de R :*
– *$u = l$,*
– *v est un sous terme de r tel que $v(\Lambda) \in \mathcal{D}$.*
Une chaîne de dépendance *est une séquence de paires de dépendance $(u_i, v_i)_i$ telle qu'il existe une substitution σ et*

$$\forall i,\ v_i\sigma \xrightarrow[R]{(\neq\Lambda)^*} u_{i+1}\sigma$$

Exemple 6.7 *Soit $\mathcal{F} = \{0, S, +\}$ et R le système de réécriture défini sur $\mathcal{T}(\mathcal{F},\mathcal{X})$ par*

$$\begin{array}{lcl} x+0 & \to & x \\ x+S(y) & \to & S(x+y) \end{array}$$

Les symboles définis sont égaux à $\{+\}$ et les constructeurs sont égaux à $\{0, S\}$, les paires de dépendance se réduisent à $\{(x+S(y),\ x+y)\}$.

Théorème 6.8 (Arts et Giesel) *Soit R un système de réécriture tel que toutes ses règles $l \longrightarrow r$ vérifient $\mathcal{V}ar(r) \subseteq \mathcal{V}ar(l)$. $\longleftarrow_R$ n'est pas bien fondée si et seulement si il existe une chaîne de dépendance infinie.*

Démonstration S'il existe une chaîne de dépendance infinie $(u_i, v_i)_i$, montrons que $\longleftarrow_R$ n'est pas bien fondée : soit σ la substitution correspondante. À partir de la chaîne de dépendance, on reconstruit une séquence de réécriture dans R en accumulant les contextes perdus :
- au début le contexte $C_0[\_]$ est égal à $[\_]$,
- la paire (u_i, v_i) provenant de $l \rightarrow r$ avec $v_1 = r|_p$ donne lieu à l'étape de réécriture $C_{i-1}[l\sigma] \rightarrow C_{i-1}[r\sigma]$, le nouveau contexte $C_i[\_]$ est égal à $C_{i-1}[r[\_]_p]$, et les étapes $v_i\sigma \rightarrow^*_R u_{i+1}\sigma$ deviennent $C_i[v_i\sigma] \rightarrow^*_R C_i[u_{i+1}\sigma]$.

On a donc contruit une séquence de réécriture infinie, $\longleftarrow_R$ n'est pas bien fondée.

Réciproquement, supposons que $\longleftarrow_R$ n'est pas bien fondée. Soit t un terme non accessible minimal pour l'ordre sous-terme (*i.e.* tous les sous-termes de t sont accessibles). En particulier le symbole de tête de t est un symbole défini. Considérons le début d'une séquence de réduction infinie à partir de t, après un certain nombre de pas de $\rightarrow_R$ strictement en dessous de Λ, elle comporte une étape en tête :

$$t \xrightarrow[R]{(\neq\Lambda)^*} t' \xrightarrow[l\rightarrow r]{\Lambda} t''$$

$t'' = r\sigma$ n'est pas accessible, donc t'' comporte un sous-terme non accessible minimal t''' qui est de la forme $v\sigma$ où v est un sous-terme non variable de r. En effet si t''' est entièrement dans la partie σ de $t'' = r\sigma$, alors t''' apparaît déjà dans un sous-terme strict de $t' = l\sigma$, ce qui implique qu'un sous-terme strict de t est non-accessible, puiqu'aucune réécriture entre t et t' n'a lieu en tête.

Comme $v\sigma$ est non accessible et minimal pour l'ordre sous-terme, le symbole de tête de v est défini, (l, v) est donc une paire de dépendance de R ; le début de la séquence infinie de pas de réécriture est transformé en un début de chaîne de dépendance par

$$t \xrightarrow[R]{(\neq\Lambda)^*} t' \equiv l\sigma \underset{DP(R)}{\rightarrow} v\sigma$$

Comme $v\sigma$ est non accessible et minimal pour l'ordre sous-terme, on peut itérer la construction, et obtenir une chaîne de dépendance infinie.

$\square$

Définition 6.9 (Paires de dépendance marquées) *Soit R un système de réécriture défini sur l'algèbre de termes $\mathcal{T}(\mathcal{F}, \mathcal{X})$. L'ensemble $\mathcal{F}$ est étendu par des copies marquées de chaque symbole*

$$\hat{\mathcal{F}} = \mathcal{F} \cup \{\hat{f} \mid f \in F\}$$

On note $\hat{t}$ le terme t où le symbole de tête de t a été remplacé par sa version marquée. Une paire de dépendance marquée *(u, v) de R est formée à partir d'une règle $l \rightarrow r$ de R :*
- *$u = \hat{l}$,*
- *$\hat{v}$ où v est un sous terme de r tel que $v(\Lambda) \in \mathcal{D}$.*

Une chaîne de dépendance marquée *est une séquence de paires de dépendance marquée $(u_i, v_i)_i$ telle qu'il existe une substitution σ et*

$$\forall i,\ v_i\sigma \xrightarrow[R]{(\neq\Lambda)^*} u_{i+1}\sigma$$

Théorème 6.10 (Arts et Giesel) *$\longleftarrow_R$ n'est pas bien fondée si et seulement si il existe une chaîne de dépendance* marquée *infinie.*

Démonstration La preuve du théorème 6.8 s'adapte sans problème car les étapes de réécriture $\rightarrow_R$ ont lieu uniquement dans des sous-termes stricts, et ne sont pas gênées par les marques, et les étapes de paires de dépendance deviennent des étapes marquées.
□

Corollaire 6.11 *Soit R un système de réécriture défini sur $\mathcal{T}(\mathcal{F},\mathcal{X})$, $\preceq$ un pré-ordre défini sur $\mathcal{T}(\mathcal{F},\mathcal{X})$(resp. sur $\mathcal{T}(\hat{\mathcal{F}},\mathcal{X})$. Si $\prec$ est bien fondé, $\prec$ et $\preceq$ sont stables, et $\preceq$ est monotone, toutes les règles de R décroissent pour $\preceq$ et toutes les paires de dépendance (resp. paires de dépendance marquées) décroissent pour $\prec$, alors R termine.*

6.2 Modularité de la terminaison

Comme le montre le contre-exemple de Toyama, la terminaison n'est pas modulaire. Pour prouver la modularité dans certains cas particuliers, deux approches sont possibles,
- ajouter des conditions sur les systèmes à combiner,
- renforcer la notion de terminaison.

Dans la première approche, on peut par exemple demander une propriété de commutation :

Définition 6.12 *$\rightarrow_{R_2}$ commute sur $\rightarrow R_1$, si et seulement si*

$$\forall t_1, t_2, t_3, \quad t_1 \underset{R_1}{\rightarrow} t_2 \underset{R_2}{\rightarrow} t_3 \quad \Rightarrow \quad \exists t_2', \quad t_1 \underset{R_2}{\rightarrow} t_2' \overset{*}{\underset{R_1 \cup R_2}{\rightarrow}} t_3$$

Théorème 6.13 *Si R_1 et R_2 terminent, et si $\rightarrow R_2$ commute sur $\rightarrow R_1$, alors $R_1 \cup R_2$ termine.*

D'autres limitations sont possibles, mais les résultats sont techniquement difficiles à obtenir, et peu utilisés par les outils de terminaison automatiques.

6.2.1 C_ϵ-terminaison

La notion de terminaison classique peut être renforcée par la terminaison simple et par la C_ϵ-terminaison.

Définition 6.14 (Terminaison simple) *Soit $\mathcal{T}(\mathcal{F},\mathcal{X})$ une algèbre de termes. Le système $\Pi_{\mathcal{F}}$ des projections de $\mathcal{F}$ est l'ensemble des règles de la forme*

$$f(x_1, \ldots, x_{i_1}, x_i, x_{i+1}, \ldots, x_n) \rightarrow x_i$$

pour tous les symboles f de $\mathcal{F}$, d'arité n, et tous les entiers entre 1 et n. Un système de réécriture R défini sur $\mathcal{T}(\mathcal{F},\mathcal{X})$ termine simplement *si et seulement $R \cup \Pi_{\mathcal{F}}$ termine.*

Définition 6.15 (C_ϵ-terminaison) *R termine C_ϵ si et seulement si $R \cup \Pi$ termine, où $\Pi = \{\pi(x,y) \rightarrow x, \pi(x,y) \rightarrow y\}$, où π est un nouveau symbole (c.-à-d. n'apparaissant pas dans R).*

Théorème 6.16 (Gramlich [14]) *La terminaison simple entraîne la C_ϵ-terminaison qui entraîne la terminaison. Ces inclusions sont strictes.*

La terminaison simple possède de bons résultats de modularité, mais les preuves sont assez techniques.

Nous allons nous concentrer sur la C_ϵ-terminaison, où les résultats de modularité ont des démonstrations relativement simples, et de plus sont extensibles au cas modulo AC.

Définition 6.17 (Module de réécriture)
Soit R_1 un système de réécriture defini sur $\mathcal{T}(\mathcal{F},\mathcal{X})$. On note $\mathcal{F}_1$ l'ensemble des symboles de $\mathcal{F}$ qui appaissent dans les règles de R_1. Soit R_2 un second système de réécriture sur $\mathcal{T}(\mathcal{F},\mathcal{X})$, tel qu'aucun symbole défini par R_2 n'appartient à $\mathcal{F}_1$. Soit $\mathcal{F}_2$ l'ensemble des symboles qui appaissent dans R_2, mais pas dans R_1. On dit alors que le module *$(\mathcal{F}_2, R_2)$ étend $(\mathcal{F}_1, R_1)$; on le note $(\mathcal{F}_1, R_1) \leftarrow (\mathcal{F}_2, R_2)$.*

Deux modules $(\mathcal{F}_2, R_2)$ et $(\mathcal{F}_3, R_3)$ étendent indépendamment $(\mathcal{F}_1, R_1)$ si $\mathcal{F}_2 \cap \mathcal{F}_3 = \emptyset$.

Cette définition des modules est assez naturelle dans la mesure où les développements importants se font de manière incrémentale et modulaire. Nous allons reprendre ici un exemple présenté dans un article de Xavier Urbain [30], et portant sur l'arithmétique.

Exemple 6.18 *Le système de réécriture R qui suit décrit l'addition et la multiplication des entiers positifs en notation binaire : $\#$ représente 0, $(x)0$ représente 2 fois la valeur de x et $(x)1$ représente deux fois la valeur de x plus 1. Dans ce formalisme, 6 est noté par $\#110$, sa représentation binaire usuelle précédée de $\#$. Ce système peut être décomposé en 3 parties $R = R_0 \cup R_+ \cup R_*$:*

$$R_0 = \{\#0 \to \#$$

$$R_+ = \left\{ \begin{array}{lcl} \# + x & \to & x \\ x + \# & \to & x \\ x0 + y0 & \to & (x+y)0 \\ x0 + y1 & \to & (x+y)1 \\ x1 + y0 & \to & (x+y)1 \\ x1 + y1 & \to & ((x+y)+\#1)0 \end{array} \right.$$

$$R_\times = \left\{ \begin{array}{lcl} \# \times x & \to & \# \\ x \times \# & \to & \# \\ x0 \times y & \to & (x \times y)0 \\ x1 \times y & \to & (x \times y)0 + y \end{array} \right.$$

Si on partitionne l'ensemble des symboles de fonctions en

$$\mathcal{F}_0 = \{\#, 0, 1\} \quad \mathcal{F}_+ = \{+\} \quad \mathcal{F}_\times = \{\times\},$$

on obtient la séquence d'extensions de modules

$$(\mathcal{F}_0, R_0) \leftarrow (\mathcal{F}_+, R_+) \leftarrow (\mathcal{F}_\times, R_\times)$$

Avec l'introduction de la notion de module de réécriture, on est amené à étendre la notion de paires de dépendance :

Définition 6.19 (Paires de dépendance de modules) *Soit $(\mathcal{F}_2, R_2)$ un module étendant $(\mathcal{F}_1, R_1)$. Une paire de dépendance de $(\mathcal{F}_2, R_2)$ par rapport à $(\mathcal{F}_1, R_1)$ est une paire $(\hat{l}, \hat{v})$ telle que*

– $(\hat{l}, \hat{v})$ *est une paire marquée de* $R_1 \cup R_2$,
– $l(\Lambda)$ *et* $v(\Lambda)$ *sont deux symboles de* $\mathcal{F}_2$ *définis dans* R_2.

Exemple 6.20 *Dans l'arithmétique binaire, selon l'approche classique, il y a 5 paires de dépendance pour la multiplication :*

$$\begin{array}{l}(x0 \mathbin{\hat{\times}} y,\ x \mathbin{\hat{\times}} y)\\ (x0 \mathbin{\hat{\times}} y,\ (x \times y)\hat{0})\\ (x1 \mathbin{\hat{\times}} y,\ x \mathbin{\hat{\times}} y)\\ (x1 \mathbin{\hat{\times}} y,\ (x \times y)\hat{0})\\ (x1 \mathbin{\hat{\times}} y,\ (x \times y)0 \mathbin{\hat{+}} y),\end{array}$$

mais il n'y en a que deux pour l'approche modulaire :

$$\begin{array}{l}(x0 \mathbin{\hat{\times}} y,\ x \mathbin{\hat{\times}} y)\\ (x1 \mathbin{\hat{\times}} y,\ x \mathbin{\hat{\times}} y).\end{array}$$

Le fait d'étendre la notion de paire de dépendance induit naturellement une extension de la notion de chaîne de dépendance :

Définition 6.21 (Chaîne de dépendance relative) *Soit* $(\mathcal{F}_2, R_2)$ *un module étendant* $(\mathcal{F}_1, R_1)$ *et* S *un système de réécriture quelconque. Un chaîne de dépendance de* $(\mathcal{F}_2, R_2)$ *relative à* S *est une séquence* $(u_i, v_i)_i$ *de paires de dépendance du module* $(\mathcal{F}_2, R_2)$ *et une substitution* σ *telle que pour tout* i

$$v_i\sigma \xrightarrow[S]{(\neq\Lambda)^*} u_{i+1}\sigma$$

Le théorème suivant permet de démontrer la modularité de la C_ϵ-terminaison pour l'union de systèmes, et pour l'extension de modules :

Théorème 6.22 (Urbain [30]) *Soient* $(\mathcal{F}_1, R_1) \leftarrow (\mathcal{F}_2, R_2)$ *et* $(\mathcal{F}_1, R_1) \leftarrow (\mathcal{F}_3, R_3)$ *deux extensions indépendantes de* $(\mathcal{F}_1, R_1)$. *Si*

1. $R_1 \cup R_2 \cup R_3$ *est à branchement fini,*
2. $R_1 \cup R_2$ *termine* C_ϵ,
3. *il n'y pas de chaînes de dépendance infinie de* $(\mathcal{F}_3, R_3)$ *relatives à* $R_1 \cup R_3 \cup \Pi$,

alors $R_1 \cup R_2 \cup R_3$ *termine* C_ϵ.

La démonstration de ce théorème est basée sur le lemme fondammental suivant :

Lemme 6.23 *Soit* $(\mathcal{F}_2, R_2)$ *un module étendant* $(\mathcal{F}_1, R_1)$. *Soit* S_1 *un sous-système de* R_1, *et une chaîne de dépendance infinie de* $(\mathcal{F}_1, S_1)$ *relative à* $R_1 \cup R_2$ *minimale*[2]. *Si* $R_1 \cup R_2$ *est à branchement fini, il est possible de construire une chaîne de dépendance infinie de* $(\mathcal{F}_1, S_1)$ *relative à* $R_1 \cup \Pi$, *comportant les mêmes paires, mais une nouvelle substitution et de nouvelles étapes de réécriture entre les paires.*

Démonstration Le lemme se démontre en interprétant les termes composites qui interviennent dans la chaîne de dépendance de façon à effacer les symboles de $\mathcal{F}_2$, et à recoller les sous-termes en fabriquant des «listes» grâce au symbole binaire π. Plus précisément, étant donné $>$ un ordre total arbitraire sur $\mathcal{T}(\hat{\mathcal{F}_1} \cup \{\pi, \perp\}, \mathcal{X})$, on définit un interprétation I des termes par :

[2] c'est-à-dire construite comme dans la preuve du théorème 6.8.

- $I(x) = x$ si x est une variable,
- $I(f(t_1, \ldots, t_n)) = f(I(t1), \ldots, I(t_n))$ si $f \in \hat{\mathcal{F}}_1$,
- $I(f(t_1, \ldots, t_n)) = List\_of(Red(f(t_1, \ldots, t_n)))$ si $f \in \mathcal{F}_2$, avec

$$\begin{array}{ll} List\_of(\emptyset) = \perp & \\ List\_of(\{a\} \cup E) = \pi(a, List\_of(E)) & \text{si } \forall e \in E,\ a < e \\ Red(t) = \{I(t') \mid t \rightarrow_{R_1 \cup R_2} t'\} & \end{array}$$

Cette interprétation n'est pas nécessairement bien définie sur tous les termes à cause de l'imbrication des appels récursifs de I et des étapes de réécriture modulo $R_1 \cup R_2$, alors qu'il n'y a aucune garantie que $R_1 \cup R_2$ termine.

Quelques remarques préliminaires :

Remarque 6.24 *Si t est un terme dont tous les sous-termes ayant un symbole de $\mathcal{F}_2$ en tête sont fortement normalisables pour $R_1 \cup R_2$, alors $I(t)$ est bien définie. La démonstration se fait par induction bien fondée sur le couple composé du multiensemble des sous-termes maximaux à symbole de tête dans $\mathcal{F}_2$ de t et de la taille de t, avec l'ordre $((\overset{+}{\leftarrow}_{R_1 \cup R_2})_{mul}, <)_{lex}$. Le fait que $R_1 \cup R_2$ est à branchement fini est crucial ici, car sinon le multiensemble des réduits d'un terme n'est pas nécessairement fini.*

Remarque 6.25 *On montre maintenant que pour tous termes s et t tels que $I(s)$ et $I(t)$ sont bien définies,*

$$s \xrightarrow[R_1]{p} t \quad \Longrightarrow \quad I(s) \xrightarrow[R_1 \cup \Pi]{+} I(t)$$

De plus si $p \neq \Lambda$ et $s(\Lambda) \in \hat{\mathcal{F}}_1$ alors

$$I(s) \xrightarrow[R_1 \cup \Pi]{(\neq \Lambda)^+} I(t)$$

La propriété se démontre par induction sur la longueur de p.

1. *Si $p = \Lambda$, s est une instance d'un membre gauche d'une règle $l \rightarrow r$ de R_1, donc $s = l\sigma$ et $t = r\sigma$. Par définition de I, et comme l et r ne comportent que des symboles de $\hat{\mathcal{F}}_1$ et des variables, $I(l\sigma) = lI(\sigma)$ et $I(r\sigma) = rI(\sigma)$. Donc*

$$I(s) \equiv lI(\sigma) \xrightarrow[l \rightarrow r]{\Lambda} I(t) \equiv rI(\sigma)$$

2. *Si $p = i \cdot q$, il faut distinguer deux cas :*
   (a) *Si $s(\Lambda) \in \hat{\mathcal{F}}_1$, $s = f(s_1, \ldots, s_i, \ldots, s_n)$, $I(s) = f(I(s_1), \ldots, I(s_i), \ldots, I(s_n))$, on peut appliquer l'hypothèse d'induction à s_i avec la position q et conclure.*
   (b) *Si $s(\Lambda) \in \mathcal{F}_2$, $I(s) = List\_of(Red(s))$. Par définition $I(t) \in Red(s)$, en utilisant les projections présentes dans Π, on peut «extraire» $I(t)$ de $I(s) = List\_of(Red(s))$:*

$$I(s) \xrightarrow[\Pi]{+} I(t)$$

Remarque 6.26 *De manière analogue, on montre maintenant que pour tous termes s et t tels que $I(s)$ et $I(t)$ sont bien définies,*

$$s \xrightarrow[R_2]{p} t \quad \Longrightarrow \quad I(s) \xrightarrow[\Pi]{+} I(t)$$

De plus si $s(\Lambda) \in \hat{\mathcal{F}}_1$ alors

$$I(s) \xrightarrow[\Pi]{(\neq \Lambda)^+} I(t)$$

La propriété se démontre par cas sur le symbole de tête de s et par induction sur la longueur de p.

1. *Si $s(\Lambda) \in \hat{\mathcal{F}}_1$, $s = f(s_1, \ldots, s_n)$, $I(s) = f(I(s_1), \ldots, I(s_n))$. La règle de R_2 ne peut pas s'appliquer en tête de s, elle s'appplique dans un sous-terme direct s_i ; on peut appliquer l'hypothèse d'induction à s_i et conclure.*

2. *Si $s(\Lambda) \in \mathcal{F}_2$, $I(s) = List\_of(Red(s))$. Par définition $I(t) \in Red(s)$, en utilisant les projections présentes dans Π, on peut «extraire» $I(t)$ de $I(s) = List\_of(Red(s))$:*

$$I(s) \xrightarrow[\Pi]{+} I(t)$$

Venons en maintenant à la preuve de lemme 6.23 elle-même. Soit donc une chaîne de dépendance infinie $(u_i, v_i)_i$ de $(\mathcal{F}_1, S_1)$ relative à $R_1 \cup R_2$ minimale. En particulier, la substitution σ associée est fortement normalisable pour $R_1 \cup R_2$, et la substitution $\sigma' = I(\sigma)$ est bien définie. De plus comme les termes u_i et v_i ne contiennent pas de symboles de $\mathcal{F}_2$, $I(u_i\sigma)$ et $I(v_i\sigma)$ sont bien définies. Montrons que la chaîne $(u_i, v_i)_i$ avec σ' est une chaîne de $(\mathcal{F}_1, S_1)$ relative à $R_1 \cup \Pi$. Par définition

$$v_i\sigma \xrightarrow[R_1 \cup R_2]{(\neq\Lambda)^*} u_{i+1}\sigma.$$

Toutes ces étapes se passent strictement en dessous de la racine, et conservent donc le fait que le symbole de tête soit dans $\hat{\mathcal{F}_1}$. Donc par la remarque 6.24, tous les termes intermédiaires s sont tels que $I(s)$ est bien définie car leur symbole de tête est dans $\hat{\mathcal{F}_1}$, et les sous-termes stricts sont fortement normalisables pour $R_1 \cup R_2$ (minimalité de la chaîne).

Une étape $s \longrightarrow^{(\neq\Lambda)}_{R_1 \cup R_2} t$ se projette donc en une étape ou plusieurs étapes

$$I(s) \xrightarrow[R_1 \cup \Pi]{(\neq\Lambda)^+} I(t)$$

grâce aux remarques 6.25 et 6.26, ce qui conclut la preuve.

□

Voici maintenant la preuve du théorème principal :

Démonstration[Théorème 6.22] Par contradiction. Supposons que $R_1 \cup R_2 \cup R_3$ ne termine pas C_ϵ, c'est-à-dire $R_1 \cup R_2 \cup R_3 \cup \Pi$ ne termine pas. En utilisant le théorème standard sur les paires de dépendance, il existe une chaîne de dépendance infinie de $R_1 \cup R_2 \cup R_3 \cup \Pi$. On peut choisir une chaîne minimale, c'est-à-dire telle que tous les sous-termes stricts du premier terme de la chaîne sont fortement normalisables, et telle que les sous-termes stricts de $v_i\sigma$ sont fortement normalisables pour chaque paire $(u_i, v_i)_i$ de la chaîne et la substitution σ associée. Il en résulte que σ est fortement normalisable pour $R_1 \cup R_2 \cup R_3 \cup \Pi$.

Les paires $(\hat{u}, \hat{v})$ qui composent la chaîne sont des types suivants :

$\{1,2\}^2$ $u(\Lambda) \in \mathcal{F}_1 \cup \mathcal{F}_2$ et $v(\Lambda) \in \mathcal{F}_1 \cup \mathcal{F}_2$,

$(3,1)$ $u(\Lambda) \in \mathcal{F}_3$ et $v(\Lambda) \in \mathcal{F}_1$,

$(3,3)$ $u(\Lambda) \in \mathcal{F}_3$ et $v(\Lambda) \in \mathcal{F}_3$.

Comme les réécritures intermédaires ne peuvent pas changer le symbole de tête des termes, le symbole de tête de u_{i+1} est égal à celui de v_i, et comme $(\mathcal{F}_2, R_2)$ et $(\mathcal{F}_3, R_3)$ sont des extensions indépendantes de $(\mathcal{F}_1, R_1)$, on ne peut pas «sauter» d'un symbole de tête de $\mathcal{F}_2$ à un symbole de $\mathcal{F}_3$, et réciproquement à l'intérieur d'une même paire. La chaîne est donc forcément composée de

1. uniquement de paires de type $\{1,2\} \times \{1,2\}$
2. uniquement de paires de type $(3,3)$
3. un nombre fini de paires de type $(3,3)$, une paire de type $(3,1)$, puis uniquement de paires de type $\{1,2\} \times \{1,2\}$.

Le dernier cas se ramène au premier en amputant la chaîne de ses premières paires.

S'il existe une chaîne qui ne contient que des paires de type $\{1,2\} \times \{1,2\}$, en appliquant le lemme 6.23 avec

- $(\mathcal{F}_1, R_1) = (\mathcal{F}_1 \cup \mathcal{F}_2, R_1 \cup R_2)$,
- $S_1 = R_1 \cup R_2$,
- $(\mathcal{F}_2, R_2) = (\mathcal{F}_3 \cup \{\pi\}, R_3 \cup \Pi)$,

on peut construire une chaîne de $(\mathcal{F}_1 \cup \mathcal{F}_2, R_1 \cup R_2)$ relative à $R_1 \cup R_2 \cup \Pi$, ce qui contredit la C_ϵ terminaison de $R_1 \cup R_2$.

S'il existe une chaîne qui ne contient que des paires de type $(3, 3)$, en appliquant le lemme 6.23 avec
- $(\mathcal{F}_1, R_1) = (\mathcal{F}_1 \cup \mathcal{F}_3, R_1 \cup R_3)$,
- $S_1 = R_3$,
- $(\mathcal{F}_2, R_2) = (\mathcal{F}_2 \cup \{\pi\}, R_2 \cup \Pi)$,

on peut construire une chaîne de $(\mathcal{F}_1 \cup \mathcal{F}_3, R_3)$ relative à $R_1 \cup R_3 \cup \Pi$, et les paires de dépendance elles-mêmes sont conservées, on obtient donc une chaîne de $(\mathcal{F}_3, R_3)$ relative à $R_1 \cup R_3 \cup \Pi$, ce qui contredit la deuxième hypothèse du théorème.

□

Chapitre 7

Ordres sur les termes

7.1 Interprétations polynômiales

Définition 7.1 *Soit $\mathcal{T}(\mathcal{F},\mathcal{X})$ une algèbre de termes. Une interprétation polynômiale $[\![\ ]\!]$ est simplement un homomorphisme de $\mathcal{T}(\mathcal{F},\mathcal{X})$ dans $\mathbb{N}$ vu comme $\mathcal{F}$-algèbre :*
- *le support de chaque sorte est égal à $\mathbb{N}$.*
- *l'interprétation de chaque symbole de fonction est un polynôme à coefficients dans $\mathbb{N}$.*

Le pré-ordre $\succeq_{[\![\ ]\!]}$ associé à cette interprétation est défini par :

$$s \succeq_{[\![\ ]\!]} t \text{ si } [\![s]\!] \geq [\![t]\!]$$

$\geq$ étant l'ordre usuel sur les entiers.

Exemple 7.2 *La taille sur les termes est une interprétation polynômiale particulière définie par*

$$P_f(X_1,\ldots,X_n) = 1 + X_1 + \ldots + X_n$$

si f est un symbole de fonction d'arité n.

Proposition 7.3 *Soit $\mathcal{T}(\mathcal{F},\mathcal{X})$ une algèbre de termes et $[\![\ ]\!]$ une interprétation polynômiale. Si pour tout symbole de fonction $f : s_1 \times \ldots \times s_n \to s$, le polynôme associé P_f a des coefficients strictement positifs pour tous ses monômes de degré 1, alors le pré-ordre $\succeq_{[\![\ ]\!]}$ associé est un pré-ordre de réécriture.*

Démonstration Quelque soit l'interprétation polynômiale, il est clair que $\succeq_{[\![\ ]\!]}$ est un pré-ordre, que $\succ_{[\![\ ]\!]}$ et $\simeq_{[\![\ ]\!]}$ sont stables, et que $\succ_{[\![\ ]\!]}$ est bien fondé quelque soit l'interprétation polynômiale. De plus $\simeq_{[\![\ ]\!]}$ est monotone : si s et t sont deux termes tels que $[\![s]\!] = [\![t]\!]$, alors

$$\begin{array}{rcl}[\![f(u_1,\ldots,u_{i-1},s,u_{i+1},\ldots,u_n)]\!] & = & P_f([\![u_1]\!],\ldots,[\![u_{i-1}]\!],[\![s]\!],[\![u_{i+1}]\!],\ldots,[\![u_n]\!]) \\ & = & P_f([\![u_1]\!],\ldots,[\![u_{i-1}]\!],[\![t]\!],[\![u_{i+1}]\!],\ldots,[\![u_n]\!]) \\ & = & [\![f(u_1,\ldots,u_{i-1},t,u_{i+1},\ldots,u_n)]\!]\end{array}$$

Si l'interprétation satisfait en outre la condition sur les coefficients des monômes de degré 1, alors on a aussi la monotonie de $\succ_{[\![\ ]\!]}$: Soient s et t deux termes tels que $s \succ_{[\![\ ]\!]} t$, c'est-à-dire $[\![s]\!] > [\![t]\!]$. Nous allons montrer que

$$[\![f(u_1,\ldots,u_{i-1},s,u_{i+1},\ldots,u_n)]\!] > [\![f(u_1,\ldots,u_{i-1},t,u_{i+1},\ldots,u_n)]\!]$$

En effet $P_f(X_1,\ldots,X_n)$ peut s'écrire $a_iX_i + P'_f(X_1,\ldots,X_n)$, avec a_i strictement positif, et P'_f polynôme à coefficients positifs :

$$\begin{array}{rcl}
[\![f(u_1,\ldots,u_{i-1},s,u_{i+1},\ldots,u_n)]\!] & = & P_f([\![u_1]\!],\ldots,[\![u_{i-1}]\!],[\![s]\!],[\![u_{i+1}]\!],\ldots,[\![u_n]\!]) \\
& = & a_i[\![s]\!] + P'_f([\![u_1]\!],\ldots,[\![u_{i-1}]\!],[\![s]\!],[\![u_{i+1}]\!],\ldots,[\![u_n]\!]) \\
& > & a_i[\![t]\!] + P'_f([\![u_1]\!],\ldots,[\![u_{i-1}]\!],[\![t]\!],[\![u_{i+1}]\!],\ldots,[\![u_n]\!]) \\
& = & P_f([\![u_1]\!],\ldots,[\![u_{i-1}]\!],[\![t]\!],[\![u_{i+1}]\!],\ldots,[\![u_n]\!]) \\
& = & [\![f(u_1,\ldots,u_{i-1},t,u_{i+1},\ldots,u_n)]\!]
\end{array}$$

□

Exemple 7.4 *D'après la proposition ci-dessus, la taille fournit un pré-ordre de réécriture.*

Considérons $\mathcal{F}$, une signature monosortée contenant une constante e, un symbole unaire I, et un symbole binaire $\cdot$ (infixe), et le système de réécriture R :

$$\begin{array}{rcl}
x\cdot e & \to & x \\
x\cdot I(x) & \to & e \\
(x\cdot y)\cdot z & \to & x\cdot(y\cdot z)
\end{array}$$

On a

$$\begin{array}{lcl}
[\![x\cdot e]\!] & = & [\![x]\!]+2 \\
& > & [\![x]\!] \\
[\![x\cdot I(x)]\!] & = & 2[\![x]\!]+2 \\
& > & 1 \\
& = & [\![e]\!] \\
[\![(x\cdot y)\cdot z]\!] & = & [\![x]\!]+[\![y]\!]+[\![z]\!]+2 \\
& = & [\![x\cdot(y\cdot z)]\!]
\end{array}$$

Les deux premières règles décroissent strictement, mais pas la dernière. La taille ne permet pas de prouver la terminaison de R. Considérons maintenant l'interprétation suivante :

$$\begin{array}{lcl}
[\![e]\!]_2 & = & 1 \\
[\![I]\!]_2(X) & = & X+1 \\
[\![\cdot]\!]_2(X_1,X_2) & = & 2\times X_1+X_2+1
\end{array}$$

On a

$$\begin{array}{lcl}
[\![x\cdot e]\!]_2 & = & 2[\![x]\!]_2+2 \\
& > & [\![x]\!]_2 \\
[\![x\cdot I(x)]\!]_2 & = & 2[\![x]\!]_2+[\![I(x)]\!]_2+1 \\
& = & 2[\![x]\!]_2+[\![x]\!]_2+2 \\
& > & 1 \\
& = & [\![e]\!]_2 \\
[\![(x\cdot y)\cdot z]\!]_2 & = & 2[\![x\cdot y]\!]_2+[\![z]\!]_2+1 \\
& = & 2(2[\![x]\!]_2+[\![y]\!]_2+1)+[\![z]\!]_2+1 \\
& = & 4[\![x]\!]_2+2[\![y]\!]_2+[\![z]\!]_2+3 \\
& > & 2[\![x]\!]_2+2[\![y]\!]_2+[\![z]\!]_2+2 \\
& = & 2[\![x]\!]_2+[\![y\cdot z]\!]_2+1 \\
& = & [\![x\cdot(y\cdot z)]\!]_2
\end{array}$$

$[\![\ ]\!]_2$ *fait décroître toutes les règles, et est un pré-ordre de réécriture, donc R termine.*

7.2 Comment combiner des ordres

7.2.1 Combinaison lexicographique

Définition 7.5 *Soient $\preceq_1, \ldots, \preceq_n$ n pré-ordres définis respectivement sur les domaines non vides $\mathscr{D}_1, \ldots, \mathscr{D}_n$. La composée lexicographique $\preceq$ de $\preceq_1, \ldots, \preceq_n$ est définie sur le domaine $\mathscr{D}_1 \times \ldots \times \mathscr{D}_n$ par l'union disjointe des deux relations $\simeq$ et $\prec$:*

$$\begin{array}{lcll} (x_1, \ldots, x_n) & \simeq & (y_1, \ldots, y_n) & \text{si } \forall i = 1, \ldots n \;\; x_i \simeq_i y_i \\ (x_1, \ldots, x_n) & \prec & (y_1, \ldots, y_n) & \text{si } \exists j \in \{1, \ldots, n\} \; \forall i = 1, \ldots j-1 \;\; x_i \simeq_i y_i \quad \wedge \quad x_j \prec_j y_j \end{array}$$

$\preceq$ est notée également $(\preceq_1, \ldots, \preceq_n)_{lex}$.

Proposition 7.6 *La composée lexicographique $(\preceq_1, \ldots, \preceq_n)_{lex}$ de n pré-ordres est un pré-ordre ; la relation d'équivalence associée est $\simeq$ et l'ordre strict est $\prec$ donnés par la définition 7.5. Si $\prec_1, \ldots, \prec_n$ sont bien fondés, $\prec$ est bien fondé.*

Démonstration Toutes les preuves se font par induction sur n car

$$(\preceq_1, \ldots, \preceq_{n+1})_{lex} \equiv (\preceq_1, (\preceq_2, \ldots, \preceq_{n+1})_{lex})_{lex}.$$

Nous supposerons donc dans perte de généralité que $n = 2$.

- Montrons que si $\preceq_1$ et $\preceq_2$ sont des pré-ordres, alors $\preceq$ est un pré-ordre :
  - $\preceq$ est réflexif car les $\simeq_i$ le sont.
  - Montrons que $\preceq$ est transitif : supposons que $(x_1, x_2) \preceq (y_1, y_2)$ et $(y_1, y_2) \preceq (z_1, z_2)$. Il faut distinguer 4 cas :
    - $(x_1, x_2) \simeq (y_1, y_2)$ et $(y_1, y_2) \simeq (z_1, z_2)$. Donc $x_1 \simeq_1 y_1 \simeq_1 z_1$ et $x_2 \simeq_2 y_2 \simeq_2 z_2$, donc $(x_1, x_2) \simeq (z_1, z_2)$.
    - $(x_1, x_2) \simeq (y_1, y_2)$ et $(y_1, y_2) \prec (z_1, z_2)$.
      - $y_1 \prec_1 z_1$, donc $x_1 \simeq_1 y_1 \prec_1 z_1$, $x_1 \prec_1 z_1$, $(x_1, x_2) \prec (z_1, z_2)$ par décroissance stricte sur la première coordonnée.
      - $y_1 \simeq_1 z_1$ et $y_2 \prec_2 z_2$, donc $x_1 \simeq_1 y_1 \simeq_1 z_1$ et $x_2 \simeq_2 y_2 \prec_2 z_2$, $(x_1, x_2) \prec (z_1, z_2)$ par décroissance stricte sur la deuxième coordonnée.
    - $(x_1, x_2) \prec (y_1, y_2)$ et $(y_1, y_2) \simeq (z_1, z_2)$.
      - $x_1 \prec_1 y_1$, donc $x_1 \prec_1 y_1 \simeq_1 z_1$, $x_1 \prec_1 z_1$, $(x_1, x_2) \prec (z_1, z_2)$ par décroissance stricte sur la première coordonnée.
      - $x_1 \simeq_1 y_1$ et $x_2 \prec_2 y_2$, donc $x_1 \simeq_1 y_1 \simeq_1 z_1$ et $x_2 \prec_2 y_2 \simeq_2 z_2$, $(x_1, x_2) \prec (z_1, z_2)$ par décroissance stricte sur la deuxième coordonnée.
    - $(x_1, x_2) \prec (y_1, y_2)$ et $(y_1, y_2) \prec (z_1, z_2)$.
      - $x_1 \prec_1 y_1$. Si $y_1 \prec_1 z_1$, par transitivité, $x_1 \prec_1 z_1$ et $(x_1, x_2) \prec (z_1, z_2)$ par décroissance stricte sur la première composante.
        Si $y_1 \simeq_1 z_1$ et $y_2 \prec_2 z_2$, on a $x_1 \prec_1 z_1$ et $(x_1, x_2) \prec (z_1, z_2)$ par décroissance stricte sur la première composante.
      - $x_1 \simeq_1 y_1$ et $x_2 \prec_2 y_2$,
        Si $y_1 \prec_1 z_1$, on a $x_1 \prec_1 z_1$ et $(x_1, x_2) \prec (z_1, z_2)$ par décroissance stricte sur la première composante.
        Si $y_1 \simeq_1 z_1$ et $y_2 \prec_2 z_2$, on a donc $x_1 \simeq_1 y_1 \simeq_1 z_1$, $x_2 \prec_2 y_2 \prec_2 z_2$ et $(x_1, x_2) \prec (z_1, z_2)$ par décroissance stricte sur la deuxième composante.
- Montrons que la relation d'équivalence associée à $\preceq$ est $\simeq$. Supposons que $x \preceq y$ et $y \preceq x$. Si l'une de ces relations est $\prec$, en appliquant le raisonnement fait ci-dessus pour la transitivité, on obtient $x \prec x$, ce qui est impossible, donc en fait, $x \simeq y$. Réciproquement, si $x \simeq y$, alors on a bien $x \preceq y$ et $y \preceq x$.

- Montrons que la relation d'ordre stricte associée à $\preceq$ est $\prec$. $\prec$ est par définition $\preceq \setminus \simeq$, et d'après le point précédent, $\simeq$ est la relation d'équivalence associée à $\preceq$, donc $\prec$ est la relation stricte associée à $\preceq$.
- Montrons que si $\prec_1$ et $\prec_2$ sont bien fondés, alors $\prec$ est bien fondé, c'est-à-dire

$$\forall x_1, x_2 \quad \underset{\prec}{Acc}(x_1, x_2)$$

On va en fait montrer une propriété un peu plus forte

$$\forall x_1, x_2, x_1' \quad x_1' \simeq_1 x_1 \quad \Rightarrow \quad \underset{\prec}{Acc}(x_1', x_2),$$

par induction bien fondée sur x_1 avec $\prec_1$. Soit donc x_1 un élément de $\mathscr{D}_1$, on va donc montrer :

$$(\forall y_1, \quad y_1 \prec_1 x_1 \quad \Rightarrow \quad (\forall x_2, x_1', \; x_1' \simeq_1 y_1 \quad \Rightarrow \quad \underset{\prec}{Acc}(x_1', x_2))) \quad \Rightarrow \quad (\forall x_2, x_1', \quad x_1' \simeq_1 x_1 \quad \Rightarrow \quad \underset{\prec}{Acc}(x_1', x_2))$$

Soit donc H_1 l'hypothèse d'induction

$$\forall y_1, \quad y_1 \prec_1 x_1 \quad \Rightarrow \quad (\forall x_2, x_1', \quad x_1' \simeq_1 y_1 \quad \Rightarrow \quad \underset{\prec}{Acc}(x_1', x_2))$$

On veut montrer

$$\forall x_2, x_1', \quad x_1' \simeq_1 x_1 \quad \Rightarrow \quad \underset{\prec}{Acc}(x_1', x_2)$$

On va le montrer par induction bien fondée sur x_2 avec $\prec_2$, soit donc x_2 un élément de $\mathscr{D}_2$, on va donc montrer

$$(\forall y_2, \quad y_2 \prec_2 x_2 \quad \Rightarrow \quad (\forall x_1', \quad x_1' \simeq_1 x_1 \quad \Rightarrow \quad \underset{\prec}{Acc}(x_1', y_2))) \quad \Rightarrow \quad (\forall x_1', \quad x_1' \simeq_1 x_1 \quad \Rightarrow \quad \underset{\prec}{Acc}(x_1', x_2))$$

Soit donc H_2 l'hypothèse d'induction

$$\forall y_2, \quad y_2 \prec_2 x_2 \quad \Rightarrow \quad (\forall x_1', \quad x_1' \simeq_1 x_1 \quad \Rightarrow \quad \underset{\prec}{Acc}(x_1', y_2))$$

Soit x_1' tel que $x_1' \simeq_1 x_1$, montrons maintenant $Acc_\prec(x_1', x_2)$, c'est-à-dire

$$\forall y_1, y_2, \quad (y_1, y_2) \prec (x_1', x_2) \quad \Rightarrow \quad \underset{\prec}{Acc}(y_1, y_2).$$

Soient donc y_1 et y_2 deux éléments tels que $(y_1, y_2) \prec (x_1', x_2)$. Distinguons deux cas :
  - $y_1 \prec_1 x_1'$. Comme $x_1' \simeq_1 x_1$, on a $y_1 \prec_1 x_1$, et il suffit d'appliquer H_1 avec y_1, y_2 et y_1.
  - $y_1 \simeq_1 x_1'$ et $y_2 \prec_2 x_2$. Il suffit d'appliquer H_2 avec y_2 et y_1.

□

7.2.2 Combinaison multi-ensemble

Nous commencerons par définir les multi-ensembles sur un support (ensemble) non vide $\mathscr{D}$. La notion de multi-ensemble est une généralisation de celle d'ensemble, un élément pouvant avoir un nombre entier positif quelconque d'occurrences dans un multi-ensemble, et plus seulement 0 ou 1. Formellement un multi-ensemble (fini) est défini comme suit :

Définition 7.7 *Soit $\mathscr{D}$ un ensemble non vide. Un multi-ensemble fini sur $\mathscr{D}$ est une application M de $\mathscr{D}$ dans $\mathbb{N}$ qui est nulle sauf sur un nombre fini d'éléments de $\mathscr{D}$. L'ensemble des multi-ensembles finis sur $\mathscr{D}$ est noté $\mathscr{M}(\mathscr{D})$. Soient M_1 et M_2 deux multi-ensembles sur $\mathscr{D}$.*

- *L'intersection* $M_1 \cap M_2$ *de* M_1 *et* M_2 *est définie par* $M_1 \cap M_2(e) = \min(M_1(e), M_2(e))$.
- *L'union* $M_1 \cup M_2$ *de* M_1 *et* M_2 *est définie par* $M_1 \cup M_2(e) = \max(M_1(e), M_2(e))$.
- *La somme* $M_1 + M_2$ *de* M_1 *et* M_2 *est définie par* $M_1 \cup M_2(e) = M_1(e) + M_2(e)$.
- *La différence* $M_1 - M_2$ *de* M_1 *et* M_2 *est définie par* $M_1 - M_2(e) = \max(0, M_1(e) - M_2(e))$.
- *L'ensemble vide* $\emptyset$ *est défini par* $\forall e \in \mathscr{D} \quad \emptyset(e) = 0$.

Définition 7.8 *Soit* $\preceq$ *un pré-ordre défini sur un domaine non vide* $\mathscr{D}$. *La composée multi-ensemble* $\preceq_{mul}$ *de* $\preceq$ *est définie sur les multi-ensembles de* $\mathscr{D}$, $\mathscr{M}(\mathscr{D})$ *par la clôture (réflexive) transitive de la relation* $\preceq^1_{mul} = \simeq^1_{mul} \cup \prec^1_{mul}$

$$\begin{array}{l} M \cup \{x\} \simeq^1_{mul} M \cup \{y\} \textit{ si } x \simeq y \\ M \cup \{y_1, \ldots, y_n\} \prec^1_{mul} M \cup \{x\} \textit{ si } \forall j \in \{1, \ldots, n\} \quad y_j \prec x \end{array}$$

Par définition, il est clair que $\preceq_{mul}$ est réflexive et transitive, donc que c'est un pré-ordre.

Proposition 7.9 $\prec_{mul}$ *est un bien fondé si et seulement si* $\prec$ *est bien fondé.*

Démonstration En identifiant les éléments de $\mathscr{D}$ et les multi-ensembles singletons, il est clair que si $\prec_{mul}$ est bien fondé, $\prec$ est fondé. Supposons maintenant que $\succ$ est bien fondé. Montrons que $\succ_{mul}$ est bien fondé. Nous allons en fait montrer que $\prec^1_{mul}$ est bien fondé.

La démonstration (constructive) se fait par 3 inductions bien fondées imbriquées, en utilisant le lemme suivant :

Lemme 7.10 *Soient* a *un élément de* $\mathscr{D}$ *et* M *et* N *deux multi-ensembles tels que* $N \prec^1_{mul} \{a\} + M$. *Alors on est dans l'un des deux cas suivants :*

1. *il existe* N' *tel que* $N = \{a\} + N'$ *et* $N' \prec^1_{mul} M$,
2. *il existe* K *tel que* $N = K + M$ *et* $K \prec^1_{mul} \{a\}$.

Ce lemme se démontre tout simplement en utilisant la définition par cas de $\prec^1_{mul}$.

La proposition $\forall M, \quad Acc_{\prec^1_{mul}}(M)$ se démontre par induction structurelle sur les multi-ensembles contruits à partir de $\emptyset$ et de l'ajout d'un singleton.

- $\emptyset$ est accessible car minimal.
- Dans le cas $\{a\} + M$, par induction, on a l'hypothèse que M est accessible pour $\prec^1_{mul}$. On va montrer que $\{a\} + M$ est accessible pour $\prec^1_{mul}$ par induction bien fondée sur a avec $\prec$:

  $$(\forall b, \quad b \prec a \quad \Rightarrow \quad (\forall M, \quad \underset{\prec^1_{mul}}{Acc}(M) \quad \Rightarrow \quad \underset{\prec^1_{mul}}{Acc}(\{b\}{+}M))) \quad \Rightarrow \quad (\forall M, \quad \underset{\prec^1_{mul}}{Acc}(M) \quad \Rightarrow \quad \underset{\prec^1_{mul}}{Acc}(\{a\}{+}M))$$

  Soit donc H_1 l'hypothèse :

  $$\forall b, \quad b \prec a \quad \Rightarrow \quad (\forall M, \quad \underset{\prec^1_{mul}}{Acc}(M) \quad \Rightarrow \quad \underset{\prec^1_{mul}}{Acc}(\{b\} + M))$$

  Nous allons montrer

  $$\forall M, \quad \underset{\prec^1_{mul}}{Acc}(M) \quad \Rightarrow \quad \underset{\prec^1_{mul}}{Acc}(\{a\} + M)$$

  par induction bien fondée sur M avec $\prec^1_{mul}$:

  $$(\forall N, \quad N \prec^1_{mul} M \quad \Rightarrow \quad \underset{\prec^1_{mul}}{Acc}(\{a\} + N)) \quad \Rightarrow \quad (\underset{\prec^1_{mul}}{Acc}(M) \quad \Rightarrow \quad \underset{\prec^1_{mul}}{Acc}(\{a\} + M))$$

  Soit donc H_2 l'hypothèse :

  $$\forall N, \quad N \prec^1_{mul} M \quad \Rightarrow \quad \underset{\prec^1_{mul}}{Acc}(\{a\} + N)$$

On sait que M est accessible, montrons que $\{a\} + M$ est accessible, c'est-à-dire que

$$\forall N, \quad N \prec^1_{mul} \{a\} + M \quad \Rightarrow \quad \underset{\prec^1_{mul}}{Acc}(N)$$

Soit donc N un multi-ensemble tel $N \prec^1_{mul} \{a\} + M$, d'après le lemme 7.10, on distingue deux cas :

1. il existe N' tel que $N = \{a\} + N'$ et $N' \prec^1_{mul} M$. Il suffit d'appliquer H_2 avec N'.
2. il existe K tel que $N = K + M$ et $K \prec^1_{mul} \{a\}$. On encore une fois une induction structurelle sur K :
   - Si $K = \emptyset$, $N = M$ est accessible par hypothèse.
   - Si $K = \{b\} + K'$, par hypothèse d'induction, $K' + M$ est accessible. Utilisons maintenant H_1 avec b et $K' + M$ pour conclure.

□

7.3 RPO : Ordre récursif sur les chemins

Outre les interprétations polynômiales, il existe un type d'ordre sur les termes bien adapté à la mécanisation, il s'agit de l'ordre récursif sur les chemins.

Définition 7.11 *Soit $\mathcal{T}(\mathcal{F}, \mathcal{X})$ une algèbre de termes, soit $\preceq$ une précédence (c'est-à-dire une relation réflexive et transitive) sur $\mathcal{F}$, et soit status une application de $\mathcal{F}$dans l'ensemble à deux éléments $\{lex, mul\}$ compatible avec $\simeq$. En outre, si $f \simeq g$ et $status(f) = status(g) = lex$, alors f et g ont la même arité. $\succ$ est étendue à $\mathcal{F} \cup \mathcal{X}$, une variable n'étant comparable à aucun élément de $\mathcal{F} \cup \mathcal{X}$ (autre qu'elle-même). L'ordre récursif sur les chemins $\preceq_{rpo}$ est défini par*

$$s \equiv f(s_1, \ldots, s_n) \preceq_{rpo} t \equiv g(t_1, \ldots, t_m)$$

si l'une des conditions ci-dessous est remplie :

1. $\exists t_j \quad s \preceq_{rpo} t_j$
2. $f \prec g$ *et* $\forall i \in \{1, \ldots, n\} \quad s_i \prec_{rpo} t$
3. $f \simeq g$, $status(f) = mul$ *et* $\{s_1, \ldots, s_n\}(\preceq_{rpo})_{mul}\{t_1, \ldots, t_m\}$
4. $f \simeq g$, $status(f) = lex$, $(s_1, \ldots, s_n)(\preceq_{rpo})_{lex}(t_1, \ldots, t_n)$ *et* $\forall i \in \{1, \ldots, n\} \quad s_i \prec_{rpo} t$.

Nous allons montrer que $\preceq_{rpo}$ est un pré-ordre de réécriture.

Lemme 7.12 *$\preceq_{rpo}$ est réflexive.*

Démonstration On montre que pour tout terme s, $s \preceq_{rpo} s$. La démonstration se fait par récurrence sur la taille de s, et en utilisant les cas 7.11.3 ou 7.11.4 suivant le statut du symbole de tête de s.
□

Lemme 7.13 *Si $s = f(s_1, \ldots, s_n) \preceq_{rpo} t$, alors pour tout $i = 1, \ldots, n$, $s_i \prec_{rpo} t$.*

Démonstration La démonstration se fait par récurrence sur la somme des tailles de s et t. On écrit $t = g(t_1, \ldots, t_m)$.

- Si $s \preceq_{rpo} t$ par sous-terme, alors il existe j tel que $s \preceq_{rpo} t_j$. Par hypothèse de récurrence sur s et t_j, pour tout i, $s_i \prec_{rpo} t_j$. Par le cas 7.11.1, on obtient $s_i \preceq_{rpo} t$. Si $t \preceq_{rpo} s_i$, par hypothèse de récurrence sur t et s_i, on obtient $t_j \prec_{rpo} s_i$, ce qui contredit $s_i \prec_{rpo} t_j$. Donc $s_i \prec_{rpo} t$.
- Si $s \preceq t$ par les cas 7.11.2 ou 7.11.4, alors $s_i \prec_{rpo} t$ par définition de l'ordre.
- Si $s \preceq t$ par le cas 7.11.3, alors pour tout i, il existe j_i tel que $s_i \preceq_{rpo} t_{j_i}$, donc $s_i \preceq_{rpo} t$ par le cas 7.11.1. Supposons que $t \preceq_{rpo} s_i$. Alors, par hypothèse de récurrence sur s_i et t, $t_{j_i} \prec_{rpo} s_i$, ce qui contredit $s_i \preceq_{rpo} t_{j_i}$, donc $s_i \prec_{rpo} t$.

□

Corollaire 7.14 *Si $s \preceq_{rpo} t$ par les cas 7.11.1 ou 7.11.2, alors $s \prec_{rpo} t$*

Démonstration La démonstration se fait par contradiction. Supposons que $t \preceq_{rpo} s$. Par le lemme 7.13, $t_j \prec_{rpo} s$ pour tous les sous-termes immédiats t_j de t.

Si $s \preceq_{rpo} t$ par le cas 7.11.1, il existe j_0 tel que $s \preceq_{rpo} t_{j_0}$, d'où une contradiction.

Si $s \preceq_{rpo} t$ par le cas 7.11.2, comme le symbole de tête de t est strictement plus grand que celui de s, la seule possibilité pour obtenir $t \preceq_{rpo} s$ est par le cas 7.11.1. D'après le cas précédent, $t \prec_{rpo} s$, ce qui contredit $s \preceq_{rpo} t$.

□

Proposition 7.15 *$\preceq_{rpo}$ est une relation transitive.*

Démonstration Supposons que $s \equiv f(s_1, \ldots, s_n) \preceq_{rpo} t \equiv g(t_1, \ldots, t_m)$ et $t \preceq_{rpo} u \equiv h(u_1, \ldots, u_p)$. La démonstration se fait par récurrence sur la somme des tailles de s, t et u.

- Si $s \preceq_{rpo} t$ par le cas 7.11.1, il existe t_{j_0} tel que $s \preceq_{rpo} t_{j_0}$. Par hypothèse de récurrence avec (s, t_{j_0}, u), on conclut $s \preceq_{rpo} u$.
- Si $t \preceq_{rpo} u$ par le cas 7.11.1, il existe u_{k_0} tel que $t \preceq_{rpo} u_{k_0}$. Par hypothèse de récurrence avec (s, t, u_{k_0}), $s \preceq_{rpo} u_{k_0}$. En appliquant le cas 7.11.1, on conclut $s \preceq_{rpo} u$.
- Si $t \preceq_{rpo} u$ par le cas 7.11.2, $g \prec h$ et pour tous les t_j, $t_j \prec_{rpo} u$.
  - Si $s \preceq_{rpo} t$ par les cas 7.11.2, 7.11.3 ou 7.11.4, $f \preceq g$, et donc $f \prec h$. De plus, par le lemme 7.13, $s_i \prec_{rpo} t$, et par hypothèse de récurrence avec (s_i, t, u), $s_i \preceq_{rpo} u$. Si $u \preceq_{rpo} s_i$, en utilisant l'hypothèse de récurrence avec (t, u, s_i), on obtient $t \preceq_{rpo} s_i$, ce qui est impossible d'après le lemme 7.13 ; donc $s_i \prec_{rpo} u$, et $s \preceq_{rpo} u$ par le cas 7.11.2.
- Si $t \preceq_{rpo} u$ par le cas 7.11.3, $g \simeq h$ et $\{t_1, \ldots, t_m\}(\preceq_{rpo})_{mul}\{u_1, \ldots, u_p\}$.
  - Si $s \preceq_{rpo} t$ par le cas 7.11.2, $g \prec h$, et donc $f \prec h$. De plus, par le lemme 7.13, $s_i \prec_{rpo} t$, et par hypothèse de récurrence avec (s_i, t, u), $s_i \preceq_{rpo} u$. Si $u \preceq_{rpo} s_i$, en utilisant l'hypothèse de récurrence avec (t, u, s_i), on obtient $t \preceq_{rpo} s_i$, ce qui est impossible d'après le lemme 7.13 ; donc $s_i \prec_{rpo} u$, et $s \preceq_{rpo} u$ par le cas 7.11.2.
  - Si $s \preceq_{rpo} t$ par le cas 7.11.3, $f \simeq g$ et $\{s_1, \ldots, s_n\}(\preceq_{rpo})_{mul}\{t_1, \ldots, t_m\}$. Par hypothèse de récurrence, $\preceq_{rpo}$ est transitive sur l'ensemble des termes $\{s_1, \ldots, s_n, t_1, \ldots, t_m, u_1, \ldots, u_p\}$, et donc $(\preceq_{rpo})_{mul}$ est transitive sur les multi-ensembles construits sur cet ensemble, donc $\{s_1, \ldots, s_n\}(\preceq_{rpo})_{mul}\{u_1, \ldots, u_p\}$, et $s \preceq_{rpo} u$ par le cas 7.11.3.
  - Le cas $s \preceq_{rpo} t$ par 7.11.4 est impossible car $f \simeq g \simeq h$ ne peut avoir à la fois le status lexicographique et multi-ensemble.
- Si $t \preceq_{rpo} u$ par le cas 7.11.4, $g \simeq h$, $(t_1, \ldots, t_m)(\preceq_{rpo})_{lex}(u_1, \ldots, u_p)$ et pour tous les t_j, $t_j \prec_{rpo} u$.
  - Si $s \preceq_{rpo} t$ par le cas 7.11.2, $g \prec h$, et donc $f \prec h$. De plus, par le lemme 7.13, $s_i \prec_{rpo} t$, et par hypothèse de récurrence avec (s_i, t, u), $s_i \preceq_{rpo} u$. Si $u \preceq_{rpo} s_i$, en utilisant l'hypothèse de récurrence avec (t, u, s_i), on obtient $t \preceq_{rpo} s_i$, ce qui est impossible d'après le lemme 7.13 ; donc $s_i \prec_{rpo} u$, et $s \preceq_{rpo} u$ par le cas 7.11.2.
  - Le cas $s \preceq_{rpo} t$ par 7.11.3 est impossible car $f \simeq g \simeq h$ ne peut avoir à la fois le status lexicographique et multi-ensemble.

– Si $s \preceq_{rpo} t$ par le cas 7.11.4, $f \simeq g$, $(s_1,\ldots,s_n)(\preceq_{rpo})_{lex}(t_1,\ldots,t_m)$ et pour tous les s_i, $s_i \prec_{rpo} t$. De même que précédemment par récurrence, $(s_1,\ldots,s_n)(\preceq_{rpo})_{lex}(t_1,\ldots,t_n)$ et $(t_1,\ldots,t_n)(\preceq_{rpo})_{lex}(u_1,\ldots,u_n)$ entraîne que $(s_1,\ldots,s_n)(\preceq_{rpo})_{lex}(u_1,\ldots,u_n)$. Par ailleurs, pour tout $i = 1,\ldots,n$, $s_i \prec_{rpo} t \preceq_{rpo} u$, donc $s_i \preceq_{rpo} u$, et $u \preceq_{rpo} s_i$ entraîne une contradiction ($t \preceq_{rpo} s_i$), donc $s_i \prec_{rpo} u$, et $s \preceq_{rpo} u$ par le cas 7.11.4.

□

Proposition 7.16 *$\prec_{rpo}$ et $\simeq_{rpo}$ sont monotones.*

Démonstration Supposons que $s \preceq_{rpo} t$. Il est facile de voir que $\preceq_{rpo}$ est monotone en utilisant les cas 7.11.3 ou 7.11.4 suivant le statut de f pour conclure que

$$f(u_1,\ldots,u_{i-1},s,u_{i+1},\ldots,u_n) \preceq_{rpo} f(u_1,\ldots,u_{i-1},t,u_{i+1},\ldots,u_n).$$

On en déduit que $\simeq_{rpo}$ est monotone.

Supposons maintenant que $s \prec_{rpo} t$. D'après ce qui précède, $u_s = f(u_1,\ldots,u_{i-1},s,u_{i+1},\ldots,u_n) \preceq_{rpo} u_t = f(u_1,\ldots,u_{i-1},t,u_{i+1},\ldots,u_n)$. Si on a $u_t \preceq_{rpo} u_s$, on distingue suivant les cas :

– Si $u_t \preceq_{rpo} u_s$ par le cas 7.11.1, il existe un sous-terme direct de u_s qui est plus grand que u_t. Ça ne peut pas être un u_j, donc c'est s, et donc $t \prec_{rpo} u_t \preceq_{rpo} s$, ce qui contredit $s \prec_{rpo} t$. Ce cas est impossible.
– $u_t \preceq_{rpo} u_s$ par le cas 7.11.2 est impossible car u_s et u_t ont le même symbole de tête.
– Si $u_t \preceq_{rpo} u_s$ par le cas 7.11.3, $\{u_1,\ldots,u_{i-1},t,u_{i+1},\ldots,u_n\}(\preceq_{rpo})_{mul}\{u_1,\ldots,u_{i-1},s,u_{i+1},\ldots,u_n\}$, donc nécessairement $t \preceq_{rpo} s$, ce qui contredit $s \prec_{rpo} t$. Ce cas est impossible.
– Si $u_t \preceq_{rpo} u_s$ par le cas 7.11.4, on conclut de même.

Finalement, $u_t \preceq_{rpo} u_s$ est impossible, et donc $u_s \prec_{rpo} u_t$.
□

Proposition 7.17 *$\prec_{rpo}$ et $\simeq_{rpo}$ sont stables.*

Démonstration Soient $s = f(s_1,\ldots,s_n)$, $t = g(t_1,\ldots,t_m)$ deux termes. On montre par récurrence sur la somme des tailles de s et t que si $s \preceq_{rpo} t$ alors pour toute substitution σ, on a $s\sigma \preceq_{rpo} t\sigma$, et que $s \prec_{rpo} t$ alors pour toute substitution σ, on a $s\sigma \prec_{rpo} t\sigma$.

– Si $s \preceq_{rpo} t$ par le cas 7.11.1, il existe j tel que $s \preceq_{rpo} t_j$. Par hypothèse de récurrence $s\sigma \preceq_{rpo} t_j\sigma$, et donc $s\sigma \preceq_{rpo} t\sigma$ par le cas 7.11.1. D'après le corollaire 7.14, $s\sigma \prec_{rpo} t\sigma$.
– si $s \preceq_{rpo} t$ par le cas 7.11.2, $f \prec g$, et pour tout $i = 1,\ldots,n$, $s_i \prec_{rpo} t$. Par hypothèse de récurrence, $s_i\sigma \prec_{rpo} t\sigma$, et donc $s\sigma \preceq_{rpo} t\sigma$ par le cas 7.11.2. D'après le corollaire 7.14, $s\sigma \prec_{rpo} t\sigma$.
– si $s \preceq_{rpo} t$ par le cas 7.11.3, $f \simeq g$, et $\{s_1,\ldots,s_n\}(\preceq_{rpo})_{mul}\{t_1,\ldots,t_m\}$. Par hypothèse de récurrence, $\{s_1\sigma,\ldots,s_n\sigma\}(\preceq_{rpo})_{mul}\{t_1\sigma,\ldots,t_m\sigma\}$, et donc $s\sigma \preceq_{rpo} t\sigma$ par le cas 7.11.3. Si en outre $s \prec_{rpo} t$, $\{s_1,\ldots,s_n\}(\preceq_{rpo})_{mul}\{t_1,\ldots,t_m\}$ est une relation stricte, donc par hypothèse de récurrence, $\{s_1\sigma,\ldots,s_n\sigma\}(\preceq_{rpo})_{mul}\{t_1\sigma,\ldots,t_m\sigma\}$ est stricte également. on ne peut donc pas avoir $t\sigma \preceq_{rpo} s\sigma$ par le cas 7.11.3. Ça n'est pas possible non plus par les cas 7.11.1 et 7.11.2, sinon on aurait $t\sigma \prec_{rpo} s\sigma$, ce qui contredit $s\sigma \preceq_{rpo} t\sigma$. Comme le statut de f est multi-ensemble, le cas 7.11.4 n'est pas à envisager.
– si $s \preceq_{rpo} t$ par le cas 7.11.4, $f \simeq g$, $(s_1,\ldots,s_n)(\preceq_{rpo})_{lex}(t_1,\ldots,t_m)$, et pour tout $i = 1,\ldots,n$, $s_i \prec_{rpo} t$. Par hypothèse de récurrence, $(s_1\sigma,\ldots,s_n\sigma)(\preceq_{rpo})_{lex}(t_1\sigma,\ldots,t_m\sigma)$ et pour tout $i = 1,\ldots,n$, $s_i\sigma \prec_{rpo} t\sigma$. Donc $s\sigma \preceq_{rpo} t\sigma$ par le cas 7.11.4. Si en outre $s \prec_{rpo} t$, la relation $(s_1,\ldots,s_n)(\preceq_{rpo})_{lex}(t_1,\ldots,t_m)$ est stricte, donc $(s_1\sigma,\ldots,s_n\sigma)(\preceq_{rpo})_{lex}(t_1\sigma,\ldots,t_m\sigma)$ est stricte également, et donc on ne peut pas avoir $t\sigma \preceq_{rpo} s\sigma$ par le cas 7.11.4. Ça n'est pas possible non plus par les cas 7.11.1 et 7.11.2, sinon on aurait $t\sigma \prec_{rpo} s\sigma$, ce qui contredit $s\sigma \preceq_{rpo} t\sigma$. Comme le statut de f est lexicographique, le cas 7.11.3 n'est pas à envisager.

□

Proposition 7.18 *$\prec_{rpo}$ est bien fondé.*

La démonstration est basée sur le lemme suivant :

Lemme 7.19 *Soit $\mathcal{T}(\mathcal{F},\mathcal{X})$ une algèbre de termes, $f \in \mathcal{F}$ un symbole de fonction d'arité n, et $(s_1,\ldots,s_n)$ un n-uplet de termes accessible pour $\prec_{rpo}$. Alors si $\prec$ est bien fondée, $f(s_1,\ldots,s_n)$ est accessible pour $\prec_{rpo}$.*

Démonstration Soit $\mathscr{A}$ l'ensemble

$$\{(f,(t_1,\ldots,t_n)) \in \mathcal{F} \times (\mathcal{T}(\mathcal{F},\mathcal{X}))^n \mid arit(f) = n \quad \wedge \quad \forall i, \quad 1 \leq i \leq n \quad \Rightarrow \quad \underset{\prec_{rpo}}{Acc}(t_i)\}$$

et $\prec\!\!\prec_{rpo}$ l'ordre défini sur $\mathscr{A}$ par

$$(f,(s_1,\ldots,s_n)) \prec\!\!\prec_{rpo} (g,(t_1,\ldots,t_m))$$
$$\text{si et seulement si}$$
$$\begin{cases} f \prec g \\ \text{ou} \quad f \simeq g \in Mul, \{s_1,\ldots,s_n\}(\prec_{rpo})_{mul}\{t_1,\ldots,t_m\} \\ \text{ou} \quad f \simeq g \in Lex, n = m, (s_1,\ldots,s_n)(\prec_{rpo})_{lex}(t_1,\ldots,t_m) \end{cases}$$

Par définition de $\mathscr{A}$, si $\prec$ est bien fondée, $\prec\!\!\prec_{rpo}$ est une relation bien fondée de $\mathscr{A}$.

Nous allons montrer

$$\forall (f,(t_1,\ldots,t_n)) \in A, \quad \underset{\prec_{rpo}}{Acc}(f(t_1,\ldots,t_n))$$

par induction bien fondée sur $\mathscr{A}$avec $\prec\!\!\prec_{rpo}$. Soit donc H_1 l'hypothèse d'induction

$$\forall (g,(s_1,\ldots,s_m)) \in \mathscr{A}, \quad (g,(s_1,\ldots,s_m)) \prec\!\!\prec_{rpo} (f,(t_1,\ldots,t_n)) \quad \Rightarrow \quad \underset{\prec_{rpo}}{Acc}(g(s_1,\ldots,s_m))$$

Nous allons montrer $Acc_{\prec_{rpo}}(f(t_1,\ldots,t_n))$, c'est-à-dire

$$\forall s \in \mathcal{T}(\mathcal{F},\mathcal{X}), \quad s \prec_{rpo} f(t_1,\ldots,t_n) \quad \Rightarrow \quad \underset{\prec_{rpo}}{Acc}(s)$$

Nous allons montrer cette propriété par induction bien fondée sur la taille de t. Soit donc H_2 l'hypothèse d'induction

$$\forall u \in \mathcal{T}(\mathcal{F},\mathcal{X}), \quad |u| < |s| \quad \Rightarrow \quad u \prec_{rpo} f(t_1,\ldots,t_n) \quad \Rightarrow \quad \underset{\prec_{rpo}}{Acc}(u)$$

- Si $s \prec_{rpo} f(t_1,\ldots,t_n)$ par le cas 7.11.1, il existe i tel que $s \preceq_{rpo} t_i$. Par hypothèse t_i est accessible pour $\prec_{rpo}$, donc s également.
- Si $s \prec_{rpo} f(t_1,\ldots,t_n)$ par le cas 7.11.2, alors $s = g(s_1,\ldots,s_m)$, $g \prec f$, et pour tout $j = 1,\ldots,m$, $s_j \prec_{rpo} f(t_1,\ldots,t_n)$. Par H_2, les s_j sont accessible pour $\prec_{rpo}$, et donc $(g,(s_1,\ldots,s_m))$ est dans $\mathscr{A}$, et $(g,(s_1,\ldots,s_m)) \prec\!\!\prec_{rpo} (f,(t_1,\ldots,t_n)$, donc par H_1, $g(s_1,\ldots,s_m)$ est accessible pour $\prec_{rpo}$.
- Si $s \prec_{rpo} f(t_1,\ldots,t_n)$ par le cas 7.11.3, alors $s = g(s_1,\ldots,s_m)$, $f \simeq g$, et $\{s_1,\ldots,s_m\}(\preceq_{rpo})_{mul}\{t_1,\ldots,t_n\}$. Pour tout s_j, il existe t_i tel que $s_j \preceq_{rpo} t_i$, et comme t_i est accessible pour $\prec_{rpo}$, s_j l'est aussi. On conclut comme précédemment que $(g,(s_1,\ldots,s_m))$ est dans $\mathscr{A}$, et $(g,(s_1,\ldots,s_m)) \prec \prec_{rpo} (f,(t_1,\ldots,t_n)$, donc par H_1, $g(s_1,\ldots,s_m)$ est accessible pour $\prec_{rpo}$.
- Si $s \prec_{rpo} f(t_1,\ldots,t_n) \succ_{rpo}$ par le cas 7.11.4, alors $s = g(s_1,\ldots,s_n)$, $f \simeq g$, et pour tout $j = 1,\ldots,n$, $s_j \prec_{rpo} f(t_1,\ldots,t_n)$. Par H_2, les s_j sont accessible pour $\prec_{rpo}$, et donc $(g,(s_1,\ldots,s_m))$ est dans $\mathscr{A}$, et $(g,(s_1,\ldots,s_m)) \prec\!\!\prec_{rpo} (f,(t_1,\ldots,t_n)$, donc par H_1, $g(s_1,\ldots,s_m)$ est accessible pour $\prec_{rpo}$.

□

Proposition 7.20 *Si $\prec$ est bien fondée, $\prec_{rpo}$ est bien fondé.*

Démonstration On montre que tout terme t est accessible pour $\prec_{rpo}$ par récurrence sur la taille de t. Soit $t = f(t_1, \ldots, t_n)$ un terme : par hypothèse de récurrence, les t_is sont accessibles pour $\prec_{rpo}$, et par le lemme 7.19, t est aussi accessible pour $\prec_{rpo}$.
□

D'après les propositions ci-dessus, nous pouvons conclure :

Théorème 7.21 *$\preceq_{rpo}$ est un pré-ordre de réécriture.*

Exemple 7.22 *Considérons la même signature et le même système de réécriture qu'à l'exemple 7.4. Soit le RPO défini par $e < I < \cdot$, $\cdot$ ayant un statut lexicographique de gauche à droite. On a alors*

$$
\begin{array}{lll}
x & <_{rpo} \; x \cdot e & \textit{par le cas 7.11.1.} \\
e & <_{rpo} \; x \cdot I(x) & \textit{par le cas 7.11.2.} \\
x \cdot (y \cdot z) & <_{rpo} \; (x \cdot y) \cdot z & \textit{par le cas 7.11.4.}
\end{array}
$$

Détaillons la dernière ligne :
- $(x, y \cdot z)(<_{rpo})_{lex}(x \cdot y, z)$ *car* $x <_{rpo} x \cdot y$ *par le cas 7.11.1.*
- $x <_{rpo} (x \cdot y) \cdot z$ *par le cas 7.11.1 itéré deux fois*
- $y \cdot z <_{rpo} (x \cdot y) \cdot z$ *par le cas 7.11.4. :*
  - $(y, z)(<_{rpo})_{lex}(x \cdot y, z)$ *car* $y <_{rpo} x \cdot y$ *par le cas 7.11.1.*
  - $y <_{rpo} (x \cdot y) \cdot z$ *par le cas 7.11.1 itéré deux fois.*
  - $z <_{rpo} (x \cdot y) \cdot z$ *par le cas 7.11.1.*

7.4 KBO : Knuth-Bendix Ordering

L'ordre KBO [21] est antérieur au RPO et est défini à partir d'une précédence $\prec$ sur les symboles de fonction, et d'une fonction de poids $w : \mathcal{F} \longrightarrow \mathbb{N}$. Le poids est étendu aux termes de la façon suivante :

$$w(f(t_1, \ldots, t_n)) = w(f) + \Sigma_{i=1}^{n} w(t_i)$$

KBO est alors défini par :

$$\frac{w(s) < w(t)}{s <_{kbo} t}$$

$$\frac{w(f(s_1, \ldots, s_n)) = w(g(t_1, \ldots, t_m)) \text{ et } f \prec g}{f(s_1, \ldots, s_n) <_{kbo} g(t_1, \ldots, t_m)}$$

$$\frac{w(f(s_1, \ldots, s_n)) = w(f(t_1, \ldots, t_n)) \text{ et } (s_1, \ldots, s_n)(<_{kbo})_{lex}(t_1, \ldots, t_n)}{f(s_1, \ldots, s_n) <_{kbo} f(t_1, \ldots, t_n)}$$

Pour que cette relation soit un ordre de réécriture, il faut en outre des conditions dites de compatiblité :
- toutes les constantes de $\mathcal{F}$ ont un poids strictement positif.
- il y a au plus une fonction unaire de poids nul, et s'il y en a une, elle est maximale pour $\prec$.

Chapitre 8

Unification

L'unification est un mécanisme clef de la démonstration automatique, elle intervient au cours de la vérification de la confluence locale d'un système de réécriture, de la complétion et dans la résolution par paramodulation (cf. PROLOG). Nous avons déjà vu la notion de problème d'unification modulo un ensemble d'équations au chapitre 4. Dans le présent chapitre, nous allons élargir nos premières définitions, et traiter des problèmes d'unification modulo un ensemble vide d'équations (unification syntaxique), modulo l'associativité et la commutativité d'un seul symbole de fonction (unification AC), et finalement nous verrons comment combiner des algorithmes d'unification pour des théories disjointes.

8.1 Unification syntaxique

Définition 8.1 (Problème d'unification) *Soit $\mathcal{T}(\mathcal{F},\mathcal{X})$ une algèbre de termes. Un problème d'unification est de la forme*

- $\top$
- $\perp$
- $s_1 = t_1 \wedge \ldots \wedge s_n = t_n$

Toute substitution est solution de $\top$, aucune substitution n'est solution de $\perp$, et σ est solution de $s_1 = t_1 \wedge \ldots \wedge s_n = t_n$ si pour tout $i = 1, \ldots, n$, $s_i\sigma = t_i\sigma$. L'ensemble des solutions d'un problème d'unification P sera noté $\mathscr{U}(P)$.

Théorème 8.2 *Soit $\mathcal{T}(\mathcal{F},\mathcal{X})$une algèbre de termes et soit P un problème d'unification dans cette algèbre. Alors $\mathscr{U}(P)$ vérifie l'une des propriétés suivantes :*

- *$\mathscr{U}(P)$ est vide,*
- *$\mathscr{U}(P)$ est non vide, et admet une substitution principale, unique à renommage près. Cette substitution est appelé unificateur principal de P, et est notée $mgu(P)$ (pour* most general unifier*).*

Nous allons démontrer ce théorème en exhibant un problème d'unification équivalent dont les solutions sont immédiates.

Définition 8.3 *Soient P_1 et P_2 deux problèmes d'unification sur la même algèbre. P_1 et P_2 sont équivalents si leurs ensembles de solutions $\mathscr{U}(P_1)$ et $\mathscr{U}(P_2)$ sont identiques.*

8.1.1 Formes résolues

Les problèmes dont les solutions sont immédiates sont les problèmes en forme résolue :

Définition 8.4 *Soit P un problème d'unification sur une algèbre de termes $\mathcal{T}(\mathcal{F},\mathcal{X})$. P est en forme résolue si P est égal à*
- $\top$
- $\bot$
- $x_1 = t_1 \wedge \ldots \wedge x_n = t_n$ *où les x_i sont des variables deux à deux distinctes, et les t_j sont des termes qui ne contiennent pas les variables x_i.*

Proposition 8.5 *Soit $P \equiv x_1 = t_1 \wedge \ldots \wedge x_n = t_n$ un problème d'unification en forme résolue, et soit θ la substitution*

$$\{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$$

L'ensemble des solutions de P est égal à l'ensemble des instances de θ :

$$\mathscr{U}(P) = \{\theta\sigma \mid \sigma \textit{ est une substitution}\}$$

Démonstration Soit σ une solution de P, nous allons montrer que σ est une instance de θ par elle-même, $\sigma = \theta\sigma$. En effet, soit x_i une variable du domaine de θ :

$$\begin{array}{rcll} x_i(\theta\sigma) & = & (x_i\theta)\sigma & \\ & = & t_i\sigma & \\ & = & x_i\sigma & \text{car } \sigma \text{ est solution de } P. \end{array}$$

Soit y une variable qui n'appartient pas au domaine de θ :

$$\begin{array}{rcll} y(\theta\sigma) & = & (y\theta)\sigma & \\ & = & y\sigma & \text{car } y\theta = y. \end{array}$$

Montrons maintenant que toute instance $\theta\sigma$ est une solution de P :

$$\begin{array}{rcll} x_i(\theta\sigma) & = & (x_i\theta)\sigma & \\ & = & t_i\sigma & \\ & = & t_i\theta\sigma & \text{car } t_i \text{ ne contient pas de variables du domaine de } \theta. \end{array}$$

□

D'après ce qui précède la substitution associée à un problème en forme résolue est une substitution principale parmi l'ensemble de ses solutions, ce qui nous amène à introduire la définition suivante :

Définition 8.6 *Si P est un problème d'unification équivalent à la forme résolue $x_1 = t_1 \wedge \ldots \wedge x_n = t_n$, la substitution $\{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$ est appelée unificateur principal de P, ou unificateur plus général de P.*

8.1.2 Règles de transformation

Nous allons voir maintenant comment transformer étape par étape un problème d'unification en un problème en forme résolue. Ces transformations sont données comme dans le cas de la logique équationnelle sous forme de règles de déduction :

Trivial	$\dfrac{s = s}{\top}$	
Décomposition	$\dfrac{f(s_1, \ldots, s_n) = f(t_1, \ldots, t_n)}{s_1 = t_1 \wedge \ldots \wedge s_n = t_n}$	
Incompatibilité	$\dfrac{f(s_1, \ldots, s_n) = g(t_1, \ldots, t_m)}{\perp}$	si $f \neq g$
Union de deux variables	$\dfrac{x = y \wedge P}{x = y \wedge P\{x \mapsto y\}}$	si $x, y \in \mathcal{V}ar(P)$.
Remplacement de variable	$\dfrac{x = s \wedge P}{x = s \wedge P\{x \mapsto s\}}$	si $x \in \mathcal{V}ar(P) \setminus \mathcal{V}ar(s)$ et $s \notin \mathcal{X}$.
Fusion	$\dfrac{x = s \wedge x = t}{x = s \wedge s = t}$	si $x \in \mathcal{X}$, $s, t \notin \mathcal{X}$ et $\lvert s\rvert \leq \lvert t\rvert$.
Test d'occurrence	$\dfrac{x_1 = t_1[x_2]_{p_1} \wedge \ldots x_n = t_n[x_1]_{p_n}}{\perp}$	si $p_1 \cdot \ldots \cdot p_n \neq \Lambda$

8.1.3 Correction

Tout d'abord, il convient de s'assurer que les règles ci-dessus transforment un problème d'unification en un problème équivalent.

Théorème 8.7 *Les règles* **Trivial**, **Décomposition**, **Incompatibilité**, **Union de deux variables**, **Remplacement de variable**, **Fusion** *et* **Test d'occurrence** *transforment un problème d'unification en un problème équivalent.*

Démonstration Les règles **Trivial**, **Union de deux variables**, **Remplacement de variable** et **Fusion** sont des conséquences du fait que l'égalité est une congruence. Les règles **Décomposition** et **Incompatibilité** proviennent du fait que l'égalité modulo $\emptyset$ sur les termes est une congruence sur les termes à partir d'un ensemble d'axiomes vide. Enfin, **Test d'occurrence** signifie qu'un terme ne peut pas être égal à un de ses sous termes strict.
□

8.1.4 Terminaison

Dans un deuxième temps, il convient de vérifier que les règles de transformation terminent.

Théorème 8.8 *Quel que soit P, problème d'unification, il n'existe pas de séquence infinie*

$$P \equiv P_0 \rightarrow_U P_1 \rightarrow_U \ldots \rightarrow_U P_n \rightarrow_U P_{n+1} \ldots$$

telle que P_{n+1} est obtenu à partir de P_n en appliquant une des règles d'unification.

Démonstration Nous allons donner une interprétation des problèmes d'unification qui décroît strictement pour chaque transformation des règles d'unification. Cette interprétation Φ est un triplet (Φ_1, Φ_2, Φ_3) :

- Une variable x est dite *résolue* dans le problème P si x apparaît exactement une fois dans P, comme membre d'une équation : $P \equiv x = s \wedge Q$, et x n'apparaît pas dans s ni dans Q. $\Phi_1(P)$ est égal au nombre de variables non résolues dans P.
- La taille d'une équation $s = t$ est égale au couple $(\max(|s|, |t|))$, et $\Phi_2(P)$ est égal au multiensemble des tailles de ses équations.
- $\Phi_3(P)$ est le nombre d'équations de P dont l'un des membres (au moins) est une variable.

Deux interprétations (r_1, M_1, v_1) et (r_2, M_2, v_2) sont comparées par $\geq$, composition lexicographique de l'ordre usuel sur $\mathbb{N}$, l'extension multiensemble de l'ordre naturel sur $\mathbb{N}$, et l'ordre naturel sur $\mathbb{N}$. L'ordre strict $>$ associé à $\geq$ est bien fondé (voir le chapitre 7 sur les ordres).

L'ordre défini sur les problèmes d'unification par

$$P > Q \text{ si } \Phi(P) > \Phi(Q)$$

Le sens de variation de (Φ_1, Φ_2, Φ_3) par l'application des règles d'unification est donné par le tableau ci dessous.

	Φ_1	Φ_2	Φ_3
Trivial	$\geq$	$>$	
Décomposition	$\geq$	$>$	
Incompatibilité	$\geq$	$>$	
Union de deux variables	$>$		
Remplacement de variable	$>$		
Fusion	$\geq$	$\geq$	$>$
Test d'occurrence	$\geq$	$>$	

□

8.1.5 Complétude

Finalement, on vérifie que les problèmes en forme normale pour l'ensemble des règles d'unification sont en forme résolue.

Théorème 8.9 *Soit P un problème d'unification sur lequel aucune des règles d'unification ne peut s'appliquer. Alors P est en forme résolue.*

Démonstration Soit P un problème en forme normale, que l'on supposera différent de $\top$ et $\bot$. Comme ni **Décomposition** ni **Incompatibilité** ne s'appliquent, P ne contient que des équations dont au moins un membre est une variable : $P \equiv x_1 = t_1 \wedge \ldots \wedge x_n = t_n$. Comme **Fusion** ne s'applique pas, les x_i sont deux à deux distincts. Si l'un des x_i a au moins deux occurrences dans P :

- x_i apparaît comme sous terme strict dans t_i : impossible car **Test d'occurrence** ne s'applique pas.
- x_i est égal à t_i : impossible car **Trivial** ne s'applique pas.
- x_i est un sous-terme strict de l'un des t_j, $i \neq j$: impossible car **Remplacement de variable** ne s'applique pas.
- x_i est égal à l'un des t_j, $j \neq i$: la $j$ième équation est $x_j = x_i$.
  - t_i n'est pas une variable : impossible car **Remplacement de variable** devrait s'appliquer avec x_i.
  - t_i est une variable qui apparaît ailleurs que dans la $i$ième équation : impossible car **Union de deux variables** ne s'applique pas.

– t_i est une variable x'_i qui n'apparaît pas ailleurs : on convient de regarder P comme

$$P \equiv x_1 = t_1 \wedge \ldots x_{i-1} = t_{i-1} \wedge x'_i = x_i \wedge x_{i+1} = t_{i+1} \wedge \ldots \wedge x_n = t_n$$

Dans tous les cas, P est une conjonction d'équations $x_1 = t_1 \wedge \ldots \wedge x_n = t_n$ où les x_i sont deux à deux distincts et ont une seule occurrence.

□

Corollaire 8.10 *Soit P un problème d'unification, et $\mathscr{U}(P)$ l'ensemble de ses solutions. Alors $\mathscr{U}(P)$ vérifie l'une des propriétés suivantes :*
- *$\mathscr{U}(P)$ est vide,*
- *$\mathscr{U}(P)$ est non vide, et admet une substitution principale, unique à renommage près.*

Démonstration D'après les théorèmes qui précèdent, en appliquant (un nombre fini de fois) les règles d'unification à P, on obtient P', un problème équivalent à P et en forme résolue. Trois cas sont possibles :
- $P' \equiv \top$: n'importe quelle substitution est solution de P', donc de P. Soit id la substitution qui a un domaine vide, $\mathscr{U}(P)$ est non vide, et admet id comme substitution principale.
- $P' \equiv \bot$: ni P ni P' n'ont de solutions et $\mathscr{U}(P)$ est vide.
- $P' \equiv x_1 = t_1 \wedge \ldots \wedge x_n = t_n$. Soit θ la substitution $\{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$. D'après la proposition 8.5 $\mathscr{U}(P')$ est égal à l'ensemble des instances de θ, donc $\mathscr{U}(P)$ est non vide et admet θ comme substitution principale.

Si $\mathscr{U}(P)$ est non vide, et admet θ et θ' comme substitutions principales, par définition il existe deux substitutions ρ et ρ' telles que :

$$\begin{array}{rcl} \theta & = & \theta'\rho' \\ \theta' & = & \theta\rho \end{array}$$

On a donc $\theta = \theta'\rho' = \theta\rho\rho'$, $\rho\rho'$ est l'identité sur les variables du codomaine de θ (*i.e.* les variables apparaissant dans les $x\theta$, $x \in \mathcal{D}om(\theta)$). De même $\rho'\rho$ est l'identité sur les variables du codomaine de θ', donc θ' est un renommage de θ.

□

8.1.6 Exemples et exercices

Dans toute cette section, nous utiliserons la signature monosortée $\mathcal{F} = \{f, g, a, b\}$, où f est binaire, g unaire, et a et b sont constantes, et l'ensemble de variables $\mathcal{X} = \{x, y, z\}$.

Voici des exemples de dérivations partant de problèmes d'unification quelconques et s'arrêtant sur des problèmes en forme résolue :

$$\begin{array}{rc} & f(a,b) = f(a,a) \\ \text{Décomposition} & \overline{\quad a = a \wedge b = a \quad} \\ \text{Trivial} & \overline{\quad b = a \quad} \\ \text{Incompatibilité} & \overline{\quad \bot \quad} \end{array}$$

$$\begin{array}{rc} & f(x,y) = f(z,z) \\ \text{Décomposition} & \overline{\quad x = z \wedge y = z \quad} \end{array}$$

$$\begin{array}{rc} & f(g(x), f(y,z)) = f(x,a) \\ \text{Décomposition} & \overline{\quad g(x) = x \wedge a = f(y,z) \quad} \\ \text{Test d'occurrence} & \overline{\quad \bot \quad} \end{array}$$

$$
\begin{array}{rc}
 & f(x, f(a,b)) = f(f(y,z),y) \\ \cline{2-2}
\text{Décomposition} & x = f(y,z) \wedge f(a,b) = y \\ \cline{2-2}
\text{Remplacement de variable} & x = f(f(a,b),z) \wedge f(a,b) = y
\end{array}
$$

$$
\begin{array}{rc}
 & f(x, f(a,y)) = f(f(b,z),x) \\ \cline{2-2}
\text{Décomposition} & x = f(b,z) \wedge f(a,y) = x \\ \cline{2-2}
\text{Fusion} & x = f(b,z) \wedge f(a,y) = f(b,z) \\ \cline{2-2}
\text{Décomposition} & x = f(b,z) \wedge a = b \wedge y = z \\ \cline{2-2}
\text{Incompatibilité} & \bot
\end{array}
$$

$$
\begin{array}{rc}
 & f(g(a), f(y,z) = f(x,x) \\ \cline{2-2}
\text{Décomposition} & g(a) = x \wedge f(y,z) = x \\ \cline{2-2}
\text{Fusion} & g(a) = x \wedge f(y,z) = g(a) \\ \cline{2-2}
\text{Incompatibilité} & \bot
\end{array}
$$

$$
\begin{array}{rc}
 & f(f(a,y), f(y,z) = f(x,x) \\ \cline{2-2}
\text{Décomposition} & f(a,y) = x \wedge f(y,z) = x \\ \cline{2-2}
\text{Fusion} & f(a,y) = x \wedge f(y,z) = f(a,y) \\ \cline{2-2}
\text{Décomposition} & f(a,y) = x \wedge y = a \wedge z = y \\ \cline{2-2}
\text{Remplacement de variable} & f(a,a) = x \wedge y = a \wedge z = a
\end{array}
$$

$$
\begin{array}{rc}
 & f(x, f(x,v)) = f(f(y,z),y) \\ \cline{2-2}
\text{Décomposition} & x = f(y,z) \wedge f(x,z) = y \\ \cline{2-2}
\text{Test d'occurrence} & \bot
\end{array}
$$

8.2 Quelques définitions pour l'unification modulo

Soit $\mathcal{T}(\mathcal{F}, \mathcal{X})$ une algèbre de termes, et E un ensemble d'équations définies sur $\mathcal{T}(\mathcal{F}, \mathcal{X})$. Un problème d'unification modulo E est défini d'une façon un peu plus générale que dans le cas syntaxique :

Définition 8.11 (Problème d'unification modulo)

1. *$\top$ et $\bot$ sont des problèmes d'unification modulo E,*
2. *$s = t$ est un problème d'unification modulo E,*
3. *si P est un problème d'unification modulo E, $\exists x, P$ en est un,*
4. *si P et Q sont des problèmes d'unification modulo E, $P \wedge Q$ et $P \vee Q$ en sont également.*

Toute substitution est solution de $\top$, aucune substitution n'est solution de $\bot$, σ est solution de $s = t$ si $s\sigma =_E t\sigma$, σ est solution de $\exists x, P$ s'il existe un terme t tel que σ est solution de $P\{x \mapsto t\}$, σ est une solution de $P \wedge Q$ si σ est une solution de P et une solution de Q, σ est une solution de $P \vee Q$ si σ est une solution de P ou une solution de Q. L'ensemble des solutions d'un problème d'unification P sera noté $\mathcal{U}_E(P)$.

Contrairement au cas syntaxique, l'unification modulo n'est en général pas unitaire. En effet modulo la commutativité de $+$, le problème $x + y = a + b$ a pour solutions

$$\{x \mapsto a; y \mapsto b\} \text{ et } \{x \mapsto b; y \mapsto a\}$$

Il n'existe pas de substitution qui soit plus générale que les deux solutions à la fois. Mais cependant l'ensemble des solutions peut parfois être représenté par un ensemble fini de solutions.

Définition 8.12 (Ensemble complet d'unificateurs) *Soit P un problème d'unification modulo E. Un ensemble $\mathcal{U}_E$ de substitutions est un ensemble complet d'unificateurs de P si*

1. *tout élément de $\mathcal{U}_E$ est une solution de P,*
2. *toute solution de P est instance modulo E d'un élément de $\mathcal{U}_E$,*

Cet ensemble est dit minimal *si en outre ses éléments sont deux à deux incomparables pour l'ordre de subsumption modulo E.*

Comme dans le cas syntaxique, on a des problèmes en forme résolue :

Définition 8.13 (Forme résolue) *Un problème d'unification est en forme résolue si il est égal à $\top$, $\bot$ ou de la forme*

$$\exists y_1, \ldots, y_p, x_1 = t_1 \wedge \ldots \wedge x_n = t_n$$

les x_is étant deux à deux distincts, distincts de tous les y_js et n'apparaissant pas dans les t_ks. Si de plus pour toutes les équations $x = y$ avec x et y variables, x et y sont libres, le problème est en forme résolue compacte.

Il est toujours possible d'obtenir une forme résolue compacte équivalente à partir d'une forme résolue en appliquant autant que possible la règle suivante :

Replace-nv	$\dfrac{\exists y_1, \ldots, y_p, y\ P \wedge x = y}{\exists y_1, \ldots, y_p\ P\{y \mapsto x\}}$

Proposition 8.14 *Soit $P \equiv \exists y_1, \ldots, y_p,\ x_1 = t_1 \wedge \ldots \wedge x_n = t_n$ un problème d'unification en forme résolue, et soit θ la substitution*

$$\{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$$

L'ensemble singleton $\{\theta\}$ est un ensemble complet d'unificateurs minimal pour P.

La preuve est calquée sur celle du cas syntaxique.

8.3 Unification AC

La théorie de l'associativité-commutativité est celle qui a été le plus étudiée pour l'unification [24, 28, 20, 12, 13, 16, 5, 4, 1, 19, 11]. Nous rappelons que la théorie $AC(+)$ est présentée par les deux axiomes

$$\begin{array}{rclr} (x + y) + z & = & x + (y + z) & (A) \\ x + y & = & y + x & (C) \end{array}$$

En général, il peut y avoir plusieurs symboles AC, et des symboles de fonction quelconques.

L'unification AC n'est pas unitaire, mais elle est finitaire, c'est-à-dire que tout problème d'unification modulo AC soit n'admet pas de solutions, soit admet un ensemble complet d'unificateurs *fini*.

8.3.1 Formes canoniques et égalité modulo AC

Les termes qui contiennent des symboles AC sont généralement représentés sous une forme applatie, et parfois même grâce à un ordre total arbitraire sur les termes, sous une forme dite canonique. C'est sur des termes applatis que fonctionne les algorithmes classiques d'unification AC, et les formes canoniques permettent de tester facilement l'égalité modulo AC de deux termes.

Définition 8.15 (Forme applatie) *La forme applatie $\overline{\overline{t}}$ d'un terme t est définie de façon récursive comme suit :*

$$
\begin{array}{rcl}
\overline{\overline{x}} & = & x \qquad\qquad \textit{si } x \textit{ est une variable} \\
\overline{\overline{f(t_1,\dots,t_n)}} & = & f(\overline{\overline{t_1}},\dots,\overline{\overline{t_n}}) \qquad \textit{si } f \textit{ n'est pas un symbole AC} \\
\overline{\overline{+(t_1,t_2)}} & = & \begin{cases}
+(\overline{\overline{t_1}},\overline{\overline{t_2}}) & \textit{si } t_1(\Lambda) \neq +, t_2(\Lambda) \neq + \\
+(\overline{\overline{t_1}},v_1,\dots,v_n) & \textit{si } t_1(\Lambda) \neq +, \overline{\overline{t_2}} = +(v_1,\dots,v_n) \\
+(u_1,\dots,u_m,\overline{\overline{t_2}}) & \textit{si } \overline{\overline{t_1}} = +(u_1,\dots,u_m), t_2(\Lambda) \neq + \\
+(u_1,\dots,u_m,v_1,\dots,v_n) & \textit{si } \overline{\overline{t_1}} = +(u_1,\dots,u_m), \overline{\overline{t_2}} = +(v_1,\dots,v_n)
\end{cases}
\end{array}
$$

Intuitivement, celà revient à ignorer le parenthèsage pour les symboles AC, ce qui est correct grâce à l'associativité.

La commutativité permet en outre de trier les arguments des symboles AC en définissant une forme canonique. On se donne pour celà un ordre total arbitraire sur l'ensemble des termes.

Définition 8.16 (Forme canonique) *La forme canonique $\overline{t}$ d'un terme t est définie de façon récursive comme suit :*

$$
\begin{array}{rcl}
\overline{x} & = & x \qquad\qquad \textit{si } x \textit{ est une variable} \\
\overline{f(t_1,\dots,t_n)} & = & f(\overline{t_1},\dots,\overline{t_n}) \qquad \textit{si } f \textit{ n'est pas un symbole AC} \\
\overline{+(t_1,t_2)} & = & \begin{cases}
+(Sort(\overline{t_1},\overline{t_2})) & \textit{si } t_1(\Lambda) \neq +, t_2(\Lambda) \neq + \\
+(Sort(\overline{t_1},v_1,\dots,v_n)) & \textit{si } t_1(\Lambda) \neq +, \overline{t_2} = +(v_1,\dots,v_n) \\
+(Sort(u_1,\dots,u_m,\overline{t_2})) & \textit{si } \overline{t_1} = +(u_1,\dots,u_m), t_2(\Lambda) \neq + \\
+(Sort(u_1,\dots,u_m,v_1,\dots,v_n)) & \textit{si } \overline{t_1} = +(u_1,\dots,u_m), \overline{t_2} = +(v_1,\dots,v_n)
\end{cases}
\end{array}
$$

où $Sort$ est une fonction de tri sur les listes de termes qui utilise l'ordre arbitraire cité plus haut.

Le théorème qui suit fait partie du folklore dans la communauté de la réécriture. Une forme équivalente à la première partie a été énoncée par Hullot et Contejean a donné une preuve formelle en Coq de la seconde partie.

Théorème 8.17 *Deux termes s et t sont égaux modulo AC si et seulement si*

1. *les symboles de tête de $\overline{\overline{s}}$ et $\overline{\overline{t}}$ sont identiques,*
2. *si le symbole de tête commun n'est pas AC, les sous-termes directs de $\overline{\overline{s}}$ et $\overline{\overline{t}}$ sont deux à deux égaux modulo AC.*
3. *si le symbole de tête commun est AC, les deux listes de sous-termes directs de $\overline{\overline{s}}$ et $\overline{\overline{t}}$ sont identiques à permutation près, et modulo AC.*

Deux termes sont égaux modulo AC si et seulement si leurs formes canoniques sont identiques.

8.3.2 Unification AC élémentaire

Nous supposerons dans le reste de cette section que la signature sur laquelle sont bâtis les termes contient un seul symbole binaire $+$ qui sera noté de façon infixe. Les termes seront en outre supposés applatis.

Nous allons donner une méthode pour l'unification AC-élémentaire, mais auparavant, nous introduisons une notation pour les substitutions (formes résolues) qui est adaptée à ce problème.

Définition 8.18 *Soient $v_1, \ldots, v_n$ des variables, $t_1, \ldots, t_m$ des termes, et pour $i \in \{1, \ldots, m\}, j \in \{1, \ldots, n\}$, $a_{ij} \in \mathbb{N}$. Le tableau suivant*

$$T \equiv \begin{array}{c|cccccc} & v_1 & \cdots & v_i & \cdots & v_n \\ \hline t_1 & a_{11} & \cdots & a_{1i} & \cdots & a_{1n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ t_m & a_{m1} & \cdots & a_{mi} & \cdots & a_{mn} \end{array}$$

où, pour tout $j \in \{1, \ldots, n\}$ $\Sigma_{i=1}^{m} a_{ij} \neq 0$ désigne la substitution qui à chaque v_i associe le terme

$$\underbrace{t_1 + \cdots + t_1}_{a_{1i} \text{ fois}} + \cdots + \underbrace{t_m + \cdots + t_m}_{a_{mi} \text{ fois}}$$

Les sous-tableaux admissibles *de T sont les sous-tableaux (au sens où l'on a enlevé certaines lignes) dont la somme des coefficients d'aucune colonne n'est nulle. (Cette condition est nécessaire pour qu'un sous-tableau représente une substitution bien formée).*

Grâce au théorème 8.17, on peut se restreindre aux problèmes ne contenant que des équations de la forme

$$E \equiv \underbrace{x_1 + \cdots + x_1}_{a_1 \text{ fois}} + \cdots + \underbrace{x_n + \cdots + x_n}_{a_n \text{ fois}} = \underbrace{y_1 + \cdots + y_1}_{b_1 \text{ fois}} + \cdots + \underbrace{y_m + \cdots + y_m}_{b_m \text{ fois}}$$

où $\{x_1, \ldots, x_n\} \cap \{y_1, \ldots, y_m\} = \emptyset$. En effet, l'équation $x + s = x + t$ a les mêmes solutions que $s = t$ modulo AC.

À l'équation E ci-dessus, on associe l'équation diophantienne linéaire

$$(e) \equiv a_1 v_1 + \cdots + a_n v_n - b_1 w_1 - \cdots - b_m w_m = 0$$

Le lemme suivant établit un lien entre E et (e) :

Lemme 8.19 *Soit σ une solution de E. Soit z une variable introduite par σ. Soient $c_1, \ldots, c_n, d_1, \ldots, d_m$ les nombres respectifs d'occurrences de z dans $x_1\sigma, \ldots, x_n\sigma, y_1\sigma, \ldots, y_m\sigma$. Alors $(c_1, \ldots, c_n, d_1, \ldots, d_m)$ est une solution de (e).*

Démonstration Grâce au théorème 8.17, les formes applaties de $s\sigma$ et $t\sigma$ ont le même nombre d'occurences de z, et ces nombres sont respectivement $a_1c_1 + \cdots + a_nc_n$ et $b_1d_1 + \cdots + b_md_m$.
□

Ce lemme se généralise évidemment à des problèmes d'unification composés de plusieurs équations, qui sont en correspondance avec des systèmes de plusieurs équations diophantiennes linéaires.

Nous verrons dans la suite que l'ensemble des solutions d'un système d'équations diophantiennes linéaires possède un nombre fini de solutions minimales pour l'ordre $\leq^n$, et que les solutions minimales forment une base. En outre cette base peut se calculer, et elle permet d'obtenir un ensemble complet d'unificateurs modulo AC. Le résultat suivant a été trouvé indépendamment par Stickel et par Livesey et Siekmann [29, 24, 28] :

Théorème 8.20 (Unification AC élémentaire) *Soit P le problème d'unification modulo AC composé de p équations $E_1, \ldots, E_p$ où interviennent les n variables $x_1, \ldots, x_n$. Soit*

$$S = \{(c_{1,1}, \ldots, c_{1,n}), \ldots, (c_{m,1}, \ldots, c_{m,n})\}$$

l'ensemble des solutions minimales pour $\leq^n$, positives, non nulles du système composé des p équations diophantiennes linéaires associées respectivement à $E_1, \ldots, E_p$. Alors, les sous-tableaux admissibles de

$$T \equiv \begin{array}{c|ccc} & x_1 & \cdots & x_n \\ \hline z_1 & c_{1,1} & \cdots & c_{1,n} \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ z_m & c_{m,1} & \cdots & c_{m,n} \end{array}$$

où $z_1, \ldots, z_m$ sont de nouvelles variables, forment un ensemble complet d'unificateurs modulo AC de P.

Démonstration Correction : Tout sous-tableau admissible est une substitution bien formée. Cette substitution σ est une solution de P car le membre gauche et le membre droit de chaque équation, une fois instanciés par σ, ont le même nombre d'occurrences de chaque z_j, et sont donc égaux modulo AC.
Complétude : Soit θ une solution de P. Soit $W = \{w_1, \ldots, w_l\}$ l'ensemble des variables introduites par θ. Soit $w_k \in W$, et u_i^k le nombre d'occurrences de w_k dans $x_i\theta$; par le lemme 8.19, $(u_1^k, \ldots, u_n^k)$ est solution de $(e_1 \wedge \ldots \wedge e_p)$. Comme S est un ensemble générateur des solutions de $(e_1 \wedge \ldots \wedge e_p)$,

$$(u_1^k, \ldots, u_n^k) = \alpha_1^k s_1 + \cdots + \alpha_m^k s_m$$

où $s_j = (c_{j,1}, \ldots, c_{j,n}) \in S$. On a donc

$$\begin{array}{rcl} u_i^k & = & \alpha_1^k c_{1,i} + \cdots + \alpha_m^k c_{m,i} \\ & = & \Sigma_{j=1}^{j=m} \alpha_j^k c_{j,i} \end{array}$$

Montrons que θ est instance d'un sous-tableau admissible de T :

$$\begin{array}{rcl} x_i\theta & = & u_i^1 w_1 + \ldots + u_i^k w_k + \ldots + u_i^l w_l \\ & = & \Sigma_{k=1}^{k=l} u_i^k w_k \\ & = & \Sigma_{k=1}^{k=l} (\Sigma_{j=1}^{j=m} \alpha_j^k c_{j,i}) w_k \\ & = & \Sigma_{j=1}^{j=m} (\Sigma_{k=1}^{k=l} \alpha_j^k c_{j,i} w_k) \\ & = & \Sigma_{j=1}^{j=m} c_{j,i} (\Sigma_{k=1}^{k=l} \alpha_j^k w_k) \end{array}$$

Soit J le sous-ensemble d'indices j de $\{1, \ldots, m\}$ tel que $\Sigma_{k=1}^{k=l} \alpha_j^k \neq 0$. La substitution $\rho = \{z_j \mapsto \Sigma_{k=1}^{k=l} \alpha_j^k w_k \mid j \in J\}$ est bien définie et

$$\begin{array}{rcl} x_i\theta & = & \Sigma_{j=1}^{j=m} c_{j,i} (\Sigma_{k=1}^{k=l} \alpha_j^k w_k) \\ & = & \Sigma_{j \in J} c_{j,i} (z_j \rho) \\ & = & (\Sigma_{j \in J} c_{j,i} z_j) \rho \end{array}$$

Pour tout i, le terme $\Sigma_{j \in J} c_{j,i} z_j$ est bien formé, et donc le sous-tableau σ de T où l'on a conservé que les lignes de J est admissible, et

$$\theta =_{AC} \sigma\rho$$

□

8.4 Résolution d'équations diophantiennes linéaires

Le problème consiste à trouver les solutions entières, positives à des systèmes d'équations de la forme

$$a_1 x_1 + \cdots + a_n x_n = k$$

où $a_1, \ldots, a_n, k \in \mathbb{Z}$. Pour cela, il faut calculer d'une part les solutions minimales de l'équation, et d'autre part l'ensemble S des solutions non-nulles, positives et minimales (pour $\leq^n$) de l'*équation homogène associée*

$$a_1 x_1 + \cdots + a_n x_n = 0$$

Toute solution est alors somme d'une solution minimale et d'une combinaison linéaire de solutions de S.

Huet a proposé une borne sur les solutions minimales [17], améliorée par Lambert [22]. Disposant d'une telle borne, il suffit d'énumérer $\mathbb{N}^n$ jusqu'à la borne, et de retenir les solutions minimales.

Nous allons ici décrire une solution moins coûteuse en pratique pour calculer S. La méthode consiste également à énumérer $\mathbb{N}^n$, mais d'une façon astucieuse qui permet de réduire considérablement l'espace de recherche tout en préservant la complétude.

Nous commençons par présenter un algorithme pour une équation qui a été proposé par Clausen et Fortenbacher [6], puis, nous verrons comment Contejean et Devie ont étendu cette méthode au cas de systèmes d'équations [8]. Les deux algorithmes ont en outre l'avantage de s'adapter facilement au cas inhomogène.

8.4.1 Algorithme de Clausen-Fortenbacher

Le cas homogène

Notation 8.21 *On notera e_i le ième vecteur de la base canonique de $\mathbb{N}^n$. Etant donné un vecteur $v \in \mathbb{N}^n$, $v(i)$ désigne la ième composante de v.*

Définition 8.22 *Le* défaut *du vecteur $c \equiv (c_1, \ldots, c_n) \in \mathbb{N}^n$ par rapport à l'équation diophantienne linéaire $e \equiv a_1 x_1 + \cdots + a_n x_n = 0$ ($a_1, \ldots, a_n \in \mathbb{Z}$) est l'entier relatif $a_1 c_1 + \cdots + a_n c_n$. On le note $D(c, e)$ ou plus simplement $D(c)$, s'il n'y a pas d'ambigüité.*

D'une certaine façon, le défaut mesure combien un vecteur *n'est pas* solution d'une équation. En effet, $(c_1, \ldots, c_n) \in \mathbb{N}^n$ est solution d'une équation e si et seulement si son défaut par rapport à e est égal à zéro.

La clef de la méthode réside dans le lemme trivial suivant :

Lemme 8.23 *Soit e l'équation diophantienne linéaire $e \equiv a_1x_1 + \cdots + a_nx_n = 0$. Si $c = (c_1, \ldots, c_n) \in \mathbb{N}^n$ n'est pas solution de e, alors, pour toute solution d, plus grande que c pour l'ordre produit cartésien $>^n$, il existe un vecteur e_i de la base canonique de $\mathbb{N}^n$ tel que $d \geq^n c + e_i$ avec $D(e_i) \times D(c) < 0$.*

Démonstration Supposons que $b = (b_1, \ldots, b_n) \geq^n c$ soit solution. Alors $D(b) = 0$, et $D(b) = D(c) + D(b - c)$. $D(c)$ et $D(b - c)$ sont donc de signes opposés. Il existe au moins un vecteur de base e_i plus petit que $b - c$ et tel que $D(e_i)$ et $D(b - c)$ ont même signe.
□

La complétude de l'algorithme de la figure 8.1 découle du lemme précédent.

Entrée : l'équation $e \equiv a_1x_1 + \cdots + a_nx_n = 0$
- Soient
  - $S_0 = \emptyset$
  - $M_0 = \{e_1, \ldots, e_n\}$.
- On définit
  - $S_{i+1} = S_i \cup \{v \in M_i \mid D(v, e) = 0\}$
  - $M_{i+1} = \{v + e_i \mid \nexists s \in S_{i+1} \text{ t.q. } v \geq^n s,$
    $\text{et } D(v, e) \times D(e_i, e) < 0\}$.
- On retourne le premier S_k tel que $M_k = \emptyset$.

FIG. 8.1 – L'algorithme de Clausen-Fortenbacher, pour une équation homogène, première version.

Proposition 8.24 *Etant donné une équation diophantienne linéaire*

$$e \equiv a_1x_1 + \cdots + a_nx_n = 0$$

l'algorithme de la figure 8.1 termine toujours et calcule l'ensemble S des solutions non-nulles, positives, minimales pour $\geq^n$ de e.

Démonstration La complétude découle du lemme 8.23. La terminaison est garantie par le fait que comme chaque vecteur engendré est obtenu à partir du précédent par l'ajout d'un vecteur e_i ayant un défaut de signe opposé, la valeur absolue des défauts des vecteurs engendrés est toujours inférieure ou égale à celle des coefficients. Soit k la plus grande valeur absolue des coefficients. Si, partant d'un vecteur de base e_i, on ajoute $2k$ fois un vecteur de base ayant un défaut de signe opposé à celui du vecteur courant, par le principe des tiroirs, on engendrera deux vecteurs v et w tels que $v <^n w$ et $D(v) = D(w)$. Mais alors, $w - v$ est solution. Comme l'algorithme engendre les vecteurs en ordre non décroissant pour $>^n$, et qu'il est complet, $w - v$ appartient à l'ensemble des solutions déjà calculées avant d'engendrer w. Donc on n'incrémentera jamais w, et l'algorithme termine après au plus $2k$ itérations.
□

On remarquera, en appliquant l'algorithme de la figure 8.1 à l'équation $x + 2y - 3z = 0$ que la solution $(1, 1, 1)$ est engendrée 4 fois. Pour éviter d'engendrer plusieurs fois une solution, on peut, lorsque un vecteur v peut être incrémenté par e_i et e_j avec $i < j$, engendrer les vecteurs $v + e_i$ et $v + e_j$ où la $i$ème composante de $v + e_j$ est *gelée*, c'est-à-dire qu'on s'interdit de jamais l'incrémenter. L'algorithme ainsi modifié est évidemment plus efficace, et il reste complet pour la raison suivante.

Soit $v \in M_i$ un vecteur engendré par l'algorithme, non plus grand qu'une solution minimale, et $S_{>v}$ l'ensemble des solutions minimales plus grandes que v. Soient $e_{l_1}, \ldots, e_{l_k}$ les vecteurs de base ayant un défaut de signe opposé. Pour $1 \leq h \leq k$, soit S_h l'ensemble des solutions supérieures ou égales à $v + e_{i_h}$. Alors les ensembles $S_1, S_v \setminus S_1, S_v \setminus (S_1 \cup S_2), \ldots, S_v \setminus (S_1 \cup \cdots \cup S_{k-1})$ forment une partition de $S_{>v}$. Donc, il n'est pas nécessaire de considérer les solutions plus grandes que $v + e_i$ issues de $v + e_j$, car elles sont déjà engendrées à partir de $v + e_i$. On obtient ainsi l'algorithme de la figure 8.2 où un vecteur n'est jamais engendré deux fois.

Entrée : l'équation $e \equiv a_1x_1 + \cdots + a_nx_n = 0$
- Soient
  - $S_0 = \emptyset$
  - $M_0 = \{e_1, \ldots, e_n\}$

    où les composantes $e_i(1), \ldots, e_i(i-1)$ de e_i sont gelées pour $i > 1$.
- On définit
  - $S_{i+1} = S_i \cup \{v \in M_i \mid D(v, e) = 0\}$
  - $M_{i+1} = \{v + e_i \mid \nexists s \in S_{i+1}$ t.q. $v \geq^n s$,

    et $D(v, e) \times D(e_i, e) < 0$,

    et $v(i)$ n'est pas gelée$\}$

    où, si l'on a engendré $v + e_i$ et $v + e_j$ avec $i < j$ à partir de v, la $i$ème composante de $v(e_j)$ est gelée.
- On retourne le premier S_k tel que $M_k = \emptyset$.

FIG. 8.2 – L'algorithme de Clausen-Fortenbacher, pour une équation homogène, deuxième version.

Proposition 8.25 *Etant donné une équation diophantienne linéaire $e \equiv a_1x_1 + \cdots + a_nx_n = 0$, l'algorithme de la figure 8.2 termine toujours et calcule l'ensemble S des solutions non-nulles, positives, minimales pour $\geq^n$ de e. Chaque vecteur engendré au cours du calcul ne l'est qu'une fois.*

La figure 8.3 montre le fonctionnement de l'algorithme sur l'équation $x + 2y - 3x = 0$.

Extension au cas inhomogène

L'algorithme de Clausen-Fortenbacher s'étend facilement au cas inhomogène. En effet, toute solution positive de l'équation

$$e \equiv a_1x_1 + \cdots + a_nx_n = k$$

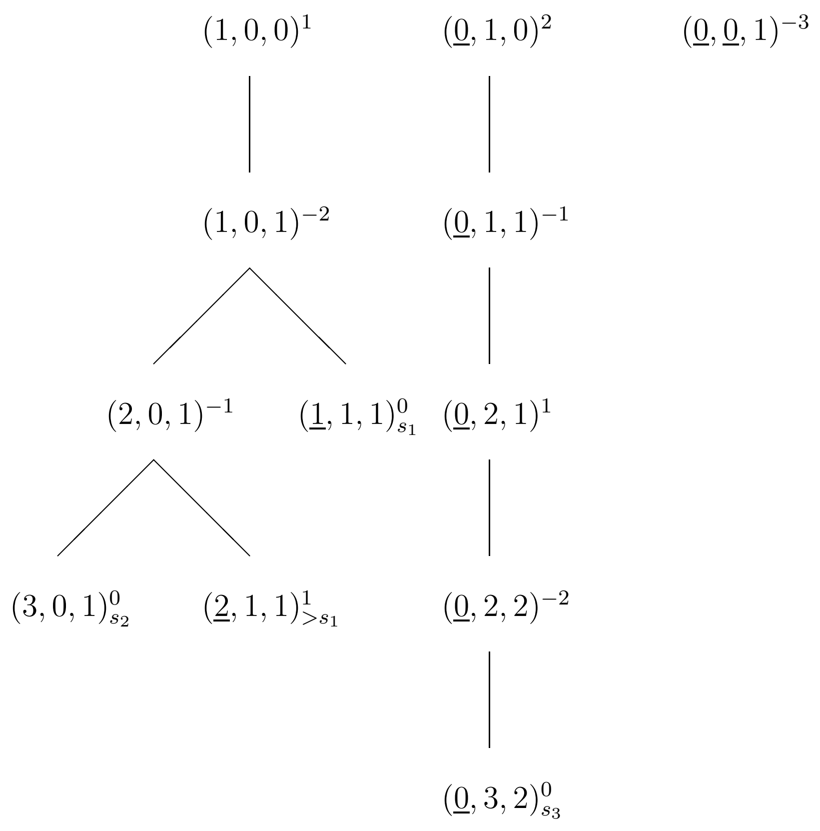
est la somme d'une solution positive, minimale, non nulle de e et d'une combinaison linéaire des solutions positives, minimales, non nulles de l'équation homogène associée

$$a_1x_1 + \cdots + a_nx_n = 0$$

Si l'on applique l'algorithme à l'équation homogène

$$e' \equiv a_1x_1 + \cdots + a_nx_n - kx' = 0$$

les solutions où x' vaut 1, sont les solutions positives, minimales, non nulles de e, et celles oú x' vaut 0 sont les solutions positives, minimales, non nulles de l'équation homogène associée. On pourra donc utiliser l'algorithme précédent en ajoutant une nouvelle variable x' et en gelant la composante associée dès qu'elle atteint 1.



Calcul de l'algorithme de Clausen-Fortenbacher sur l'équation $x + 2y - 3z = 0$. Les défauts des vecteurs figurent en exposant et les composantes gelées sont soulignées.

FIG. 8.3 – Exemple d'exécution de l'algorithme de la figure 8.2.

8.4.2 Algorithme de Contejean-Devie

Contejean et Devie ont adapté l'algorithme précédent à la résolution directe de systèmes d'équations, à partir d'une interprétation géometrique simple de l'algorithme de Clausen-Fortenbacher [7, 8].

La notion de défaut s'étend naturellement au cas de systèmes d'équations :

Définition 8.26 *Le* défaut *du vecteur $c \equiv (c_1, \ldots, c_n) \in \mathbb{N}^n$ par rapport au système $\mathcal{S}$ d'équations diophantiennes linéaires :*

$$\begin{array}{ccccccc} a_{11}x_1 & + & \cdots & + & a_{1n}x_n & = & 0 \\ . & & & & . & & . \\ . & & & & . & & . \\ . & & & & . & & . \\ a_{m1}x_1 & + & \cdots & + & a_{mn}x_n & = & 0 \end{array}$$

où les a_{ij} sont dans $\mathbb{Z}$ est ve vecteur $D(v) \in \mathbb{Z}^m$

$$(\Sigma_{i=1}^n a_{1i}c_i, \ldots, \Sigma_{i=1}^n a_{mi}c_i)$$

Un vecteur $c \equiv (c_1, \ldots, c_n) \in \mathbb{N}^n$ est alors solution de $\mathcal{S}$ si et seulement si son défaut est $(0, \ldots, 0)$.

La figure 8.4 montre comment on peut traduire la restriction de Clausen-Fortenbacher sur les défauts des variables dans le cas de plusieurs équations. Dans le cas d'une équation, la condition que le défaut du vecteur de base que l'on rajoute ait un signe contraire à celui du défaut du vecteur courant impose de n'incrémenter que les composantes dont le défaut est tel que l'on *revient vers l'origine*. Cela revient, dans le cas d'un système à n'incrémenter que les composantes dont le défaut est dans le demi-espace ouvert délimité par l'hyperplan orthogonal au défaut courant et contenant l'origine. Cette restriction préserve la complétude pour la même raison que dans le cas d'une équation : si l'on incrémente un vecteur v tel que $D(v) \neq 0$ uniquement en ses composantes qui ne vérifient pas cette restriction, on ne reviendra pas dans le demi-espace contenant l'origine. On peut donc adapter l'algorithme de Clausen-Fortenbacher à la résolution de systèmes en n'incrémentant le vecteur courant v de e_i que si le produit scalaire $D(v) \cdot D(e_i)$ est strictement négatif.

Lemme 8.27 *Soit $\mathcal{S}$ le système d'équations diophantiennes linéaires*

$$\begin{array}{ccccccc} a_{11}x_1 & + & \cdots & + & a_{1n}x_n & = & 0 \\ . & & & & . & & . \\ . & & & & . & & . \\ . & & & & . & & . \\ a_{m1}x_1 & + & \cdots & + & a_{mn}x_n & = & 0 \end{array}$$

Si $c = (c_1, \ldots, c_n) \in \mathbb{N}^n$ n'est pas solution de $\mathcal{S}$, alors, pour toute solution d, plus grande que c pour l'ordre produit cartésien $>^n$, il existe un vecteur e_i de la base canonique de $\mathbb{N}^n$ tel que $d \geq^n c + e_i$ avec $D(e_i) \cdot D(c) < 0$.

On obtient l'algorithme de la figure 8.5 qui est évidemment complet par le lemme 8.27, mais dont la terminaison n'est pas triviale,contrairement au cas d'une équation. On pourra se reporter à [8] pour la preuve de terminaison.

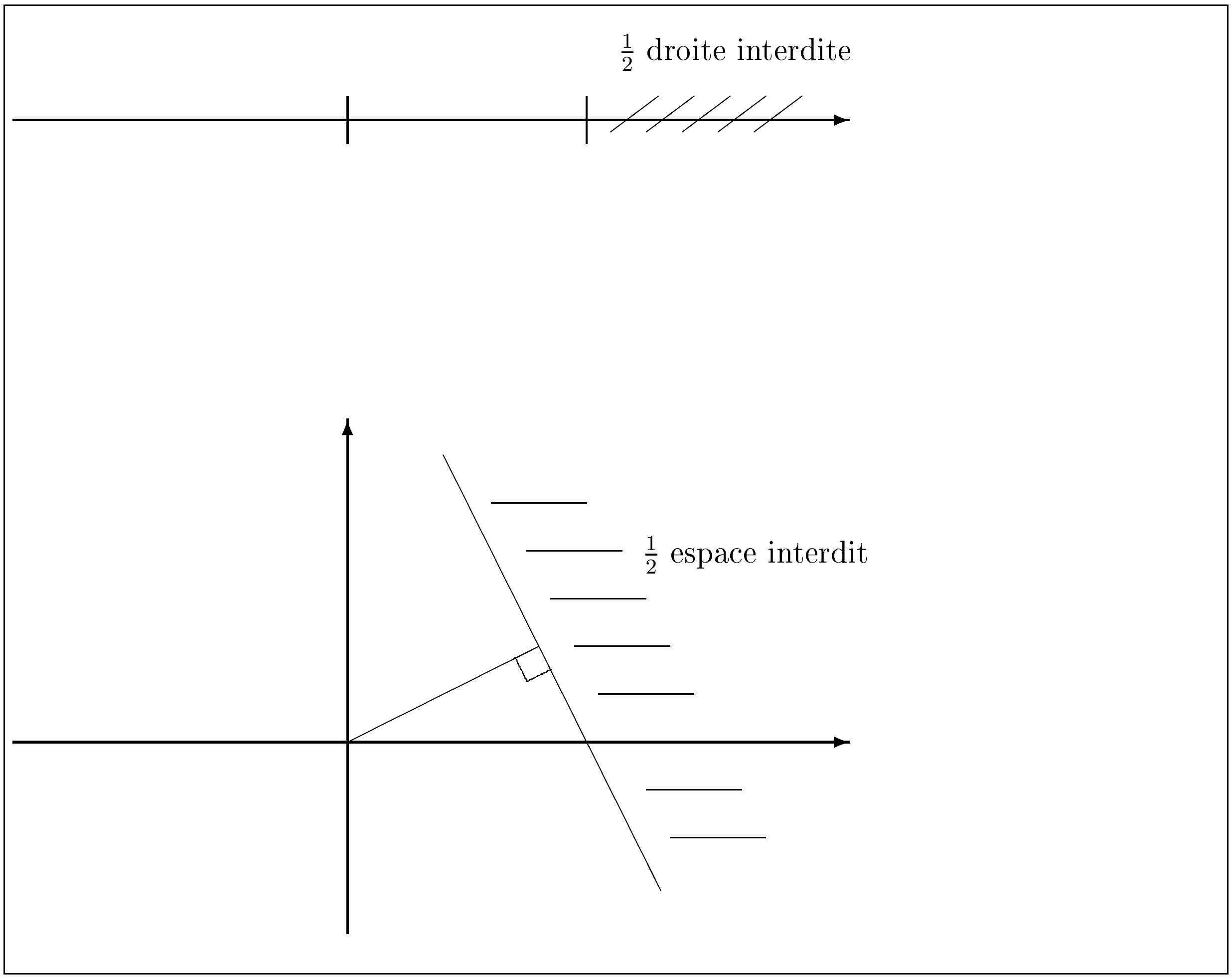


FIG. 8.4 – La restriction géométrique imposée sur les défauts.

Entrée : le système d'équations diophantiennes linéaires S

$$\begin{array}{ccccccc} a_{11}x_1 & + & \cdots & + & a_{1n}x_n & = & 0 \\ \vdots & & & & \vdots & & \vdots \\ a_{m1}x_1 & + & \cdots & + & a_{mn}x_n & = & 0 \end{array}$$

- Soient
  - $S_0 = \emptyset$
  - $M_0 = \{e_1, \ldots, e_n\}$

    où les composantes $e_i(1), \ldots, e_i(i-1)$ de e_i sont gelées pour $i > 1$.
- On définit
  - $S_{i+1} = S_i \cup \{v \in M_i \mid D(v, e) = (0, \ldots, 0)\}$
  - $M_{i+1} = \{v + e_i \mid \nexists s \in S_{i+1}$ t.q. $v \geq^n s$,

    et $D(v) \cdot D(e_i) < 0$,

    et $v(i)$ n'est pas gelée$\}$

    où, si l'on a engendré $v + e_i$ et $v + e_j$ avec $i < j$ à partir de v, la $i$ème composante de $v(e_j)$ est gelée.
- On retourne le premier S_k tel que $M_k = \emptyset$.

FIG. 8.5 – L'algorithme de Contejean-Devie pour un système d'équations diophantiennes linéaires

8.5 Combinaison d'algorithmes d'unification

Dans le but de construire des algorithmes d'unification pour des théories équationnelles présentées par des ensembles d'axiomes qui concernent des ensembles de symboles disjoints, nous allons montrer comment construire un algorithme d'unification à partir d'algorithmes pour les sous-théories disjointes. Sans perte de généralité, nous supposerons que la signature $\mathcal{F}$ est partitionnée en $\mathcal{F}_1 \cup \mathcal{F}_2$, et que la théorie E est présentée par $E_1 \cup E_2$ où les égalités de E_1 (respectivement de E_2) ne portent que sur des termes construits à partir de symboles de $\mathcal{F}_1$ (respectivement $\mathcal{F}_2$) et de variables.

Nous nous placerons dans un cadre un peu restreint, plus simple à expliquer que le cas général [3]. Nous supposerons que E_1 et E_2 sont des théories simples, c'est-à-dire que les équations de la forme $s = t$ où t est un sous-terme propre de s, n'ont pas de solutions modulo E_i, $i = 1, 2$.

Nous supposerons que l'on dispose d'algorithmes d'unification modulo E_i, $i = 1, 2$ pour résoudre des problèmes construits sur les termes de $\mathcal{T}(\mathcal{F}_i, \mathcal{X})$. L'algorithme pour la combinaison est décrit par des règles d'inférence.

8.5.1 Règles d'inférence

La première règle vise à transformer le problème de départ en un problème équivalent, mais sur lequel on pourra appliquer les algorithmes d'unification connus pour les sous-théories.

Définition 8.28 (Sous-terme étranger) *Un sous-terme u de t à la position p est* étranger *si p est de la forme $q \cdot n$, et $t(q) \in \mathcal{F}_i$ et $t(q \cdot n) \in \mathcal{F}_j$ avec $i \neq j$.*

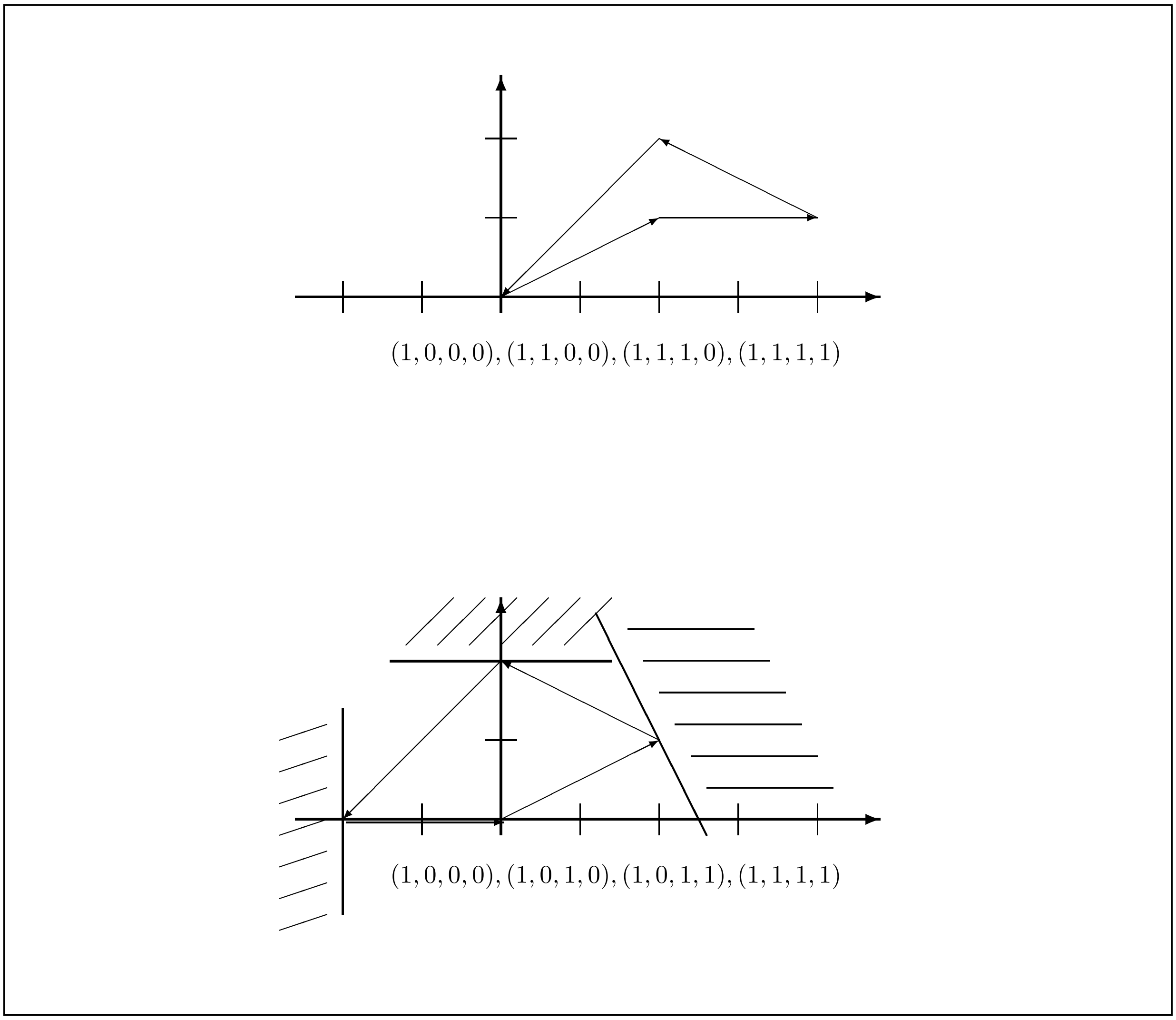


FIG. 8.6 – Deux séquences de défauts correspondant à la solution $(1,1,1,1)$ du système

$$\begin{array}{rcrcrcrcl} 2x_1 & + & 2x_2 & - & 2x_3 & - & 2x_4 & = & 0 \\ x_1 & & & + & x_3 & - & 2x_4 & = & 0 \end{array}$$

Seule la deuxième sera construite par l'algorithme de Contejean-Devie, car la première viole la restriction géometrique.

VA	$\dfrac{\exists y_1, \ldots, y_p\ s[u]_p = t \wedge P}{\exists x, y_1, \ldots, y_p\ s[x]_p = t \wedge x = u \wedge P}$	si u est un sous-terme étranger de $s[u]_p$ à la position p.

Il est clair que VA préserve l'ensemble des solutions.

Après une ou plusieurs applications de VA, il est possible que certaines équations soient bâties uniquement sur des termes de $\mathcal{T}(\mathcal{F}_i, \mathcal{X})$, on peut alors appliquer les algorithmes connus pour les sous-théories :

E-Res	$\dfrac{P_i}{P_i'}$	si P_i n'est pas en forme résolue et P_i' est une forme résolue compacte de P_i.

Il est clair que la règle (non déterministe) E-res est correcte, mais sa complétude est plus difficile à démontrer. Elle repose sur le lemme suivant :

Lemme 8.29 *Soient s et t deux termes de $\mathcal{T}(\mathcal{F}_1, \mathcal{X})$ et γ une substitution solution de $s = t$ modulo $E_1 \cup E_2$. Alors il existe une substitution σ de $\mathcal{T}(\mathcal{F}_1, \mathcal{X})$, solution de $s = t$ modulo E_1 et telle que γ est une $E_1 \cup E_2$ instance de σ.*

La preuve de ce lemme repose sur la complétion close ordonnée.

Les différents sous-problèmes communiquent uniquement par le biais de variables, c'est pourquoi la règle suivante est cruciale :

Var-Rep	$\dfrac{\exists z_1, \ldots, z_p\ x = y \wedge P}{\exists z_1, \ldots, z_p\ x = y \wedge P\{x \mapsto y\}}$	si x et y sont des variables de P, et x est quantifié existentiellement ou y est libre.

Comme les théories sont supposées simples, les conflits de théorie sont insolubles :

Conflit-1	$\dfrac{\exists z_1, \ldots, z_p\ x = s_1 \wedge x = s_2 \wedge P}{\perp}$	si $x \in \mathcal{X}$, $s_1(\Lambda) \in \mathcal{F}_1$ et $s_2(\Lambda) \in \mathcal{F}_2$.
Conflit-2	$\dfrac{\exists z_1, \ldots, z_p\ s_1 = s_2 \wedge P}{\perp}$	$s_1(\Lambda) \in \mathcal{F}_1$ et $s_2(\Lambda) \in \mathcal{F}_2$.

De même, les cycles composés n'ont pas de solutions, car le nombre maximal d'alternances de théories le long d'un chemin dans un terme est un invariant des classes modulo $E_1 \cup E_2$.

Cycle	$\dfrac{\exists y_1, \ldots, y_p\ x_1 = t_1[x_2]_{p_1} \wedge \ldots x_n = t_n[x_1]_{p_n}}{\perp}$	si $n \geq 2$, il existe i et j tels que $s_i(\Lambda) \in \mathcal{F}_1$ et $s_j(\Lambda) \in \mathcal{F}_2$.

Finalement une dernière règle permet de se débarasser des variables existentiellement quantifiées «inutiles» :

EQE	$\dfrac{\exists x, y_1, \ldots, y_p\ x = s \wedge P}{\exists y_1, \ldots, y_p\ P}$	si x n'apparaît ni dans s ni dans P.

8.5.2 Terminaison

De façon classique, on montre que l'application non-déterministe des règles VA, E-Res, Var-Rep, EQE, Conflit-1, Conflit-2, Cycle termine en utilisant une mesure qui décroît strictement à chaque application d'une règle. Cette mesure est un 5-uplet $(Th(P), SV(P), USP(P), PVR(P), PEQE(P))$. Les composantes sont définies ci-après. Tout d'abord, les problèmes considérés sont mis sous la forme

$$\exists y_1, \ldots, y_p\ P_V \wedge P_H \wedge P_1 \wedge P_2,$$

où P_V ne contient que des équations entre variables, P_H contient les équations hétérogènes du problème, et P_1 (respectivement P_2) contient les équations homogènes entre termes non tous deux variable de $\mathcal{T}(\mathcal{F}_1, \mathcal{X})$ (respectivement $\mathcal{T}(\mathcal{F}_2, \mathcal{X})$).

$Th(P)$ est le nombre de sous-termes étrangers présents dans P.

$SV(P)$ est le multiensemble $\{SV(P_1), SV(P_2)\}$: deux variables x et y sont en relation pour $\sim_Q$ si $x, y \in Var(Q)$, deux variables x et y sont en relation pour $\approx_i$ si $x, y \in Var(P_i)$ et x et y sont dans la clôture transitive de $=_V \cup \sim_{P_H} \cup \sim_{P_{(i \bmod 2)+1}}$. $SV(P_i)$ est le nombre de variables partagées de P_i, c'est-à-dire les x tels qu'il existe y et $x \approx_i y$.

$USP(P)$ est le nombre de problèmes homogènes de P qui ne sont pas résolus.

$PVR(P)$ est le nombre potentiel d'applications de Var-Rep à P, c'est-à-dire le nombre d'équations $x = y$ dans P_V telles que x et y ont tous deux au moins un autre occurence dans le problème.

$PEQE(P)$ est le nombre potentiel d'applications de EQE à P, c'est à dire le nombre de variables existentiellement quantifiées x telles que x n'apparaît que dans une seule équation $x = s$ de P.

Le tableau ci-dessous résume la preuve de terminaison :

	Th	SV	USP	PVR	PEQE
VA	$\downarrow$				
E-Res	$\downarrow =$	$\downarrow =$	$\downarrow$		
Var-Rep(1)	$\downarrow =$	$\downarrow =$	$\downarrow =$	$\downarrow$	
Var-Rep(2)	$\downarrow =$	$\downarrow$	$\uparrow$	$\downarrow$	
EQE	$\downarrow =$	$\downarrow =$	$\downarrow =$	$\downarrow =$	$\downarrow$

Cette preuve repose sur les lemmes non triviaux suivants :

Lemme 8.30 *E-Res ne fait pas croître $SV(P)$.*

Lemme 8.31 *Var-Rep ne fait pas croître SV(P). De plus, si Var-Rep est appliqué à une équation $x = u$ telle que x et y apparaissent tous deux dans P_i, alors SV(P) décroît strictement.*

8.5.3 Obtenir des formes résolues

Si un problème est tel qu'aucune de règles VA, E-Res, Var-Rep, EQE, Conflit-1, Conflit-2, Cycle ne s'applique, il est dans une forme dite séquentiellement résolue, et il suffit d'appliquer le remplacement de variable standard pour obtenir une forme résolue.

Chapitre 9

Compilation du fitrage

Dans tout ce chapitre, on suppose fixée une algèbre de termes $\mathcal{T}(\mathcal{F}, \mathcal{X})$.

9.1 Réseaux de discrimination

Les réseaux de dicrimination sont des structures qui permettent de stocker de façon compacte des ensembles de termes (linéaires) assez gros. En outre, on peut leur associer un automate qui calcule sur une entrée s (un terme) l'ensemble des filtres pour s dans l'ensemble T sous-jacent :

$$\{t \in T \mid \exists \sigma t\sigma = s\}$$

Définition 9.1 (Squelette, frange) *Un squelette S est un terme (clos) bien formé de $\mathcal{T}(\mathcal{F} \cup \{\square\})$, où $\square$ est une constante spéciale qui n'apparaît pas dans $\mathcal{F}$. La frange d'un squelette S est l'ensemble de positions de S tel que*

$$F(S) = \{p \in \mathcal{P}os(S) \mid S(p) = \square\}$$

Un terme t est compatible avec un squelette S si et seulement si

$$\forall p \in \mathcal{P}os(S),\ p \notin F(S) \Longrightarrow \begin{cases} p \in \mathcal{P}os(t) \wedge S(p) = t(p) \\ ou \\ \exists q \in \mathcal{P}os(t),\ q \leq p \wedge t(q) \in \mathcal{X} \end{cases}$$

Un ensemble de termes T est compatible avec un squelette si chacun de ses éléments l'est.

p est une position de discrimination pour S et T si T est compatible avec S, et il existe p une position de $F(S)$ et un terme de T tels que $t(p) \in \mathcal{F}$.

Définition 9.2 (Arbre de filtrage, Réseau de discrimination) *Un* arbre de filtrage *est un arbre*

1. *dont les arêtes sont dans $\mathcal{F} \cup \{\omega\}$,*
2. *dont les nœuds u sont étiquetés par S_u et M_u, un squelette et un ensemble de termes compatibles, et si u n'est pas une feuille une position p_u.*

tel que

1. *à la racine de l'arbre, le squelette est égal à $\square$,*
2. *si u n'est pas une feuille, p_u est une position de discrimination pour S_u et M_u,*

3. *si* (u, v) *est une arête étiquetée par* f,

$$S_v = S_u[f(\Box, \ldots, \Box)]_{p_u},$$

4. M_v *est l'ensemble des éléments de* M_u *compatibles avec* S_v.

Un réseau de discrimination pour T *est un arbre de filtrage maximal dont la racine est étiquetée par* $(\Box, T, \_)$.

Soit T un ensemble de termes linéaires et D un reseau de discrimination pour T. Soit A l'automate de filtrage associé à D : ses états sont les nœuds de D, et ses transitions les arêtes de D. Sur une entrée s (un terme), la transition $u \rightarrow^f v$ a lieu si

1. $s(p_u) = f$,
2. ou bien $s(p_u) \in \mathcal{X}$ et $f = \omega$.

Les automates de filtrage ainsi définis sont *déterministes*. En outre, si S atteint l'état final M_v sur l'entrée s, alors

$$M_v = \{t \in T \mid \exists\sigma \ \ t\sigma = s\}$$

Si A échoue sur l'entrée s alors

$$\{t \in T \mid \exists\sigma \ \ t\sigma = s\} = \emptyset.$$

Pour un ensemble de termes non-linéaires, on obtient un algorithme de préfiltrage en partant des termes linéarisés.

9.2 Réseaux de discrimination non déterministes

Ces réseaux sont *a priori* moins intéressants que les autres, cependant en pratique, ils sont moins gros, et plus faciles à mettre à jour. Ils sont donc parfois utilisés dans des processus comme la complétion, où l'ensemble de règles de réécriture varie.

Chapitre 10

Complétion

Nous avons vu que le problème du mot est décidable dans une théorie équationnelle E lorsque l'on dispose d'un système de réécriture R équivalent et convergent. La convergence locale de R se teste facilement sur les paires critiques, et pour prouver la terminaison, il suffit de trouver un préordre de réécriture qui fait décroître les règles de R. En général, on connaît E, et on essaie de trouver R qui convient en orientant de façon intuitive les équations de E en règles. Par exemple à partir des groupes, qui sont définis par

$$E = \{x \cdot e = x; x \cdot I(x) = e; (x \cdot y) \cdot z = x \cdot (y \cdot z)\}$$

on obtiendra

$$R = \{x \cdot e \to x; x \cdot I(x) \to e; (x \cdot y) \cdot z \to x \cdot (y \cdot z)\}$$

On peut essayer de trouver un préordre de réécriture qui fait décroître les règles de R. Ici, on pourra choisir $\geq$ le RPO associé à la précédence $I \succ \cdot \succ e$, où tous les symboles ont un statut lexicographique (de gauche à droite).

Cependant lorsque l'on teste la confluence locale de R, la paire critique

$$\begin{tikzcd}
 & (x \cdot e) \cdot z \arrow[dl] \arrow[dr] & \\
x \cdot (e \cdot z) & & x \cdot z
\end{tikzcd}$$

n'est pas convergente.

L'idée de la complétion est de rajouter cette équation, une fois orientée par $>$ au système R. Il faut noter qu'en ajoutant une règle à R, on crée de nouvelles paires critiques dont il faudra aussi vérifier la confluence.

10.1 Règles de complétion

Le processus de complétion est décrit comme un ensemble de règles d'inférence. Les données sont un ensemble d'équations E_0 définissant une théorie équationnelle sur une algèbre de termes $\mathcal{T}(\mathcal{F}, \mathcal{X})$, et un préordre de réécriture $\geq$.

Initial	$\dfrac{}{E_0 \; ; \; \emptyset}$	
Paire Critique	$\dfrac{E \; , \; R}{E \cup \{e\} \; , \; R}$	si e est une paire critique de R
Oriente	$\dfrac{E \cup \{s = t\} \; , \; R}{E \; , \; R \cup \{s \to t\}}$	si $s > t$
Trivial	$\dfrac{E \cup \{s = s\} \; , \; R}{E \; , \; R}$	
Simplifie	$\dfrac{E \cup \{s = t\} \; , \; R}{E \cup \{s' = t'\} \; , \; R}$	si $s \to_R s'$ et/ou $t \to_R t'$
Compose	$\dfrac{E \; , \; R \cup \{l \to r\}}{E \; , \; R \cup \{l \to r'\}}$	si $r \to_R r'$
Collapse	$\dfrac{E \; , \; R \cup \{l \to r\}}{E \cup \{l' = r\} \; , \; R}$	si $l \to_{g \to d} l', g \to d \in R$, et $\lvert l\rvert > \lvert g\rvert$ ou ($\lvert l\rvert = \lvert g\rvert$ et $r > d$).

Lemme 10.1 *Chacune des règles d'inférences de l'ensemble ci-dessus :*

$$\frac{E \; , \; R}{E' \; , \; R'}$$

conserve la théorie équationnelle : $=_{E \cup E_R} \equiv =_{E' \cup E_{R'}}$.

Définition 10.2 *Une séquence de complétion est un succès si partant de* $(E_0, \emptyset)$*, on aboutit en appliquant un nombre fini de fois les règles de complétion à* $(\emptyset, R)$*. Dans ce cas,* R *est un système de réécriture confluent équivalent à* E*, qui permet de décider le problème du mot modulo* E*.*

La complétion peut être entièrement automatisée si l'on dispose d'un algorithme pour décider le préordre de réécriture choisi (c'est le cas pour les préordres de type RPO). Voici deux exemples de la complétion des groupes par le logiciel CiME (http ://www.lri.fr/∼demons/cime.html). Dans le premier exemple, le niveau de verbosité est bas, et le système n'indique que le type de règle d'inférence appliquée, dans le deuxième exemple, le niveau est plus haut, et le système donne l'état courant de l'ensemble des équations et des règles.

10.2 La complétion vue comme réécriture de preuves

Bibliographie

[1] Mohamed Adi and Claude Kirchner. AC-unification race : the system solving approach. In *Proc. Int. Symposium on Design and Implementation of Symbolic Computation Systems, LNCS 429*, 1990.

[2] Thomas Arts and Jürgen Giesl. Termination of term rewriting using dependency pairs. *Theoretical Computer Science*, 236 :133–178, 2000.

[3] Alexandre Boudet. Combining unification algorithms. *Journal of Symbolic Computation*, 16 :597–626, 1993.

[4] Alexandre Boudet, Evelyne Contejean, and Hervé Devie. A new AC-unification algorithm with a new algorithm for solving diophantine equations. In *Proc. 5th IEEE Symp. Logic in Computer Science, Philadelphia*, pages 289–299. IEEE Computer Society Press, June 1990.

[5] Jim Christian and Patrick Lincoln. Adventures in associative-commutative unification. Technical Report ACA-ST-275-87, MCC, AI Program Austin, Austin, October 1987.

[6] M. Clausen and A. Fortenbacher. Efficient solution of linear diophantine equations. *Journal of Symbolic Computation*, 8(1&2) :201–216, 1989.

[7] Evelyne Contejean and Hervé Devie. Résolution de systèmes linéaires d'équations diophantiennes. *Comptes-Rendus de l'Académie des Sciences de Paris*, 313 :115–120, 1991. Série I.

[8] Evelyne Contejean and Hervé Devie. An efficient algorithm for solving systems of diophantine equations. *Information and Computation*, 113(1) :143–172, August 1994.

[9] Max Dauchet. Simulation of a turing machine by a left-linear rewrite rule. In *Proc. 3rd Rewriting Techniques and Applications, Chapel Hill, LNCS 355*, 1989.

[10] Nachum Dershowitz. Termination of rewriting. *Journal of Symbolic Computation*, 3(1) :69–115, February 1987.

[11] Eric Domenjoud. AC unification through order-sorted AC1 unification. *Journal of Symbolic Computation*, 14(6) :537–556, December 1992.

[12] François Fages. Associative-commutative unification. *Journal of Symbolic Computation*, 3(3), June 1987.

[13] A. Fortenbacher. An algebraic approach to unification under associativity and commutativity. In *Proc. Rewriting Techniques and Applications 85, Dijon, LNCS 202*. Springer-Verlag, May 1985.

[14] Bernhard Gramlich. Generalized sufficient conditions for modular termination of rewriting. *Applicable Algebra in Engineering, Communication and Computing*, 5 :131–158, 1994.

[15] Bernhard Gramlich. Abstract relations between restricted termination and confluence properties of rewrite systems. *Fundamenta Informaticæ*, 24 :3–23, 1995.

[16] Alexander Herold and Jörg H. Siekmann. Unification in abelian semi-groups. *Journal of Automated Reasoning*, 3(3) :247–283, 1987.

[17] Gérard Huet. An algorithm to generate the basis of solutions to homogeneous linear diophantine equations. *Information Processing Letters*, 7(3), April 1978.

[18] Gérard Huet and Dallas S. Lankford. On the uniform halting problem for term rewriting systems. Research Report 283, INRIA, March 1978.

[19] D. Kapur and P. Narendran. Double-exponential complexity of computing a complete set of *ac*-unifiers. In *Proc. 7th IEEE Symp. Logic in Computer Science, Santa Cruz*, June 1992.

[20] Claude Kirchner. Méthodes et outils de conception systématique d'algorithmes d'unification dans les théories equationnelles. Thèse d'Etat, Univ. Nancy, France, 1985.

[21] Donald E. Knuth and Peter B. Bendix. Simple word problems in universal algebras. In J. Leech, editor, *Computational Problems in Abstract Algebra*, pages 263–297. Pergamon Press, 1970.

[22] J. L. Lambert. Une borne pour les générateurs des solutions entières positives d'une équation diophantienne linéaire. *Comptes Rendus de l'Académie des Sciences de Paris*, 305 :39,40, 1987. Série I.

[23] Dallas S. Lankford and A. M. Ballantyne. Decision procedures for simple equational theories with permutative axioms : Complete sets of permutative reductions. Research Report Memo ATP-37, Department of Mathematics and Computer Science, University of Texas, Austin, Texas, USA, August 1977.

[24] M. Livesey and Jörg H. Siekmann. Unification of bags and sets. Research report, Institut fur Informatik I, Universität Karlsruhe, West Germany, 1976.

[25] M. H. A. Newman. On theories with a combinatorial definition of 'equivalence'. *Ann. Math.*, 43(2) :223–243, 1942.

[26] Enno Ohlebusch. On the modularity of termination of term rewriting systems. *Theoretical Computer Science*, 136 :333–360, 1994.

[27] Gerald E. Peterson and Mark E. Stickel. Complete sets of reductions for some equational theories. *Journal of the ACM*, 28(2) :233–264, April 1981.

[28] M. Stickel. A unification algorithm for associative-commutative functions. *Journal of the ACM*, 28(3) :423–434, 1981.

[29] M. E. Stickel. A complete unification algorithm for associative-commutative functions. In *Proceedings 4th International Joint Conference on Artificial Intelligence, Tbilissi (USSR)*, pages 71–76, 1975.

[30] Xavier Urbain. Modular and incremental automated termination proofs. *Journal of Automated Reasoning*, 32 :315–355, 2004.