Démonstration automatique

Traitement de l'égalité par la récriture

Évelyne Contejean Xavier Urbain

20 janvier 2011

Table des matières

Chapitre 1

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 1.1 (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'accessibilité 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 1.2 *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 :*

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

```
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 $(\forall y,\ y\ R\ x \ \Rightarrow\ Py)$ pour prouver $\forall x\ Acc_R(x)) \Rightarrow Px$.

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

Définition 1.3 *Une relation R est bien fondée sur un ensemble $\mathscr{A}$ si tout élément de $\mathscr{A}$ est accessible par R,*

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

Théorème 1.4 *Soient $\mathscr{A}$ un ensemble, R une relation sur* bien fondée *$\mathscr{A}$ et P une prédicat sur $\mathscr{A}$. Pour prouver P sur tous les éléments de $\mathscr{A}$, il suffit de montrer :*

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

Chapitre 2

Les algèbres de termes

2.1 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.

2.2 Les termes

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

2.2.1 Signature

Définition 2.1 (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 2.2 *La signature utilisée pour les entiers de Peano est égale à $(\mathscr{S}_{peano}, \mathcal{F}_{peano}, \tau_{peano})$:*

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

τ_{peano} est définie par :

$$Zero : entier\_peano$$
$$Succ : entier\_peano \to entier\_peano$$
$$plus\_petit : entierpeano \times entier\_peano \to bool$$
$$add : entier\_peano \times entier\_peano \to entier\_peano$$
$$mul : entier\_peano \times entier\_peano \to entier\_peano$$

2.2.2 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 2.3 (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 2.4 *$mul(add(Zero, Succ(Succ(Zero))), Succ(Zero))$ est un terme bâti sur la signature $(\mathscr{S}_{peano}, \mathcal{F}_{peano}, \tau_{peano})$.*

Définition 2.5 *L'ensemble des variables Var(t) d'un terme t est défini inductivement par :*

- *si x appartient à $\mathcal{X}_s$, Var$(x) = \{x\}$.*
- *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 Var$(f(t_1, \ldots, t_n)) = \bigcup_{i=1}^{n}$ Var(t_i).*

On dira qu'un terme est linéaire *s'il est une variable ou si tous ses sous-termes sont linéaires et ont des ensembles de variables deux à deux disjoints.*

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.

2.2.3 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 $\leq_{\text{pref}}$ défini par :

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

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

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

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

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

2.2.4 Une deuxième définition des termes

Définition 2.6 (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 2.7 *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 $Succ$.

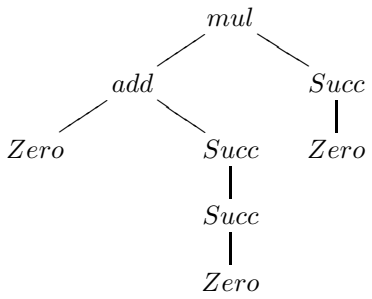
Si nous nous restreignons aux cas où l'ensemble des positions est fini, nous retrouvons les mêmes termes que ceux de la définition 2.3. Il est facile de voir que si t est un terme de sorte s au sens de la définition 2.3, alors $t(\Lambda)$ (au sens de la définition 2.6) a pour codomaine s. La profondeur 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 2.8 *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

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



2.2.5 Sous-termes

Intuitivement un sous-terme d'un terme vu comme une expression (définition 2.3) 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 2.6 :

Définition 2.9 (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 2.10 *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 2.11 *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 2.6 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 $t(p)$.

Notation 2.12 (Relation sous-terme) *On note $t \rhd s$ (ou $s \lhd t$) si s est un sous-terme propre de t, c'est-à-dire qu'il existe une position p distincte de la position en tête, telle que $t|_p = s$.*

Définition 2.13 *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 2.14 *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 2.15 *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 2.6.*
- *Le terme t et le terme $t[t|_p]_p$ sont identiques.*
- *Le terme u et le terme $(t[u]_p)|_p$ sont identiques.*

2.3 Les substitutions

Définition 2.16 (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.

2.3.1 Subsumption

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

Définition 2.17 (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 $\mathcal{T}(\mathcal{F}, \mathcal{X})$ 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 2.18 *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 $\mathcal{X} = \{x, y, z, u\}$.

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

$x\sigma$ 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 $y\sigma\theta = y\tau$.

2.3.2 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 2.19 *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 $t \stackrel{\bullet}{=} t'$.

Exercice 2.20 *Montrer que $\stackrel{\bullet}{=}$ est une relation d'équivalence.*

Exemple 2.21 *Si ρ est un renommage égal à $\{x \mapsto x', y \mapsto y'\}$, 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 $\rho' = \{x \mapsto x', y \mapsto y', z \mapsto z\}$, 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 $add(x', x')$ 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 $add(x', x') = t'\rho''$: $add(x', x')$ est une instance stricte de $add(x, y)$, et non pas un renommage.

2.3.3 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 2.22 *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* $[\![t]\!] \subseteq [\![s]\!]$.
– $s \stackrel{\bullet}{=} t$ *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.

2.4 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 2.23 *Soit $(\mathcal{S}, \mathcal{F}, \tau)$ une signature. Une $\mathcal{F}$-algèbre est constituée*
– *d'un support non vide $\mathscr{A}_s$ associé à chaque sorte s de $\mathcal{S}$,*
– *d'une interprétation $f_{\mathscr{A}}$ pour chaque symbole de fonction f de $\mathcal{F}$ telle que si $f : s_1 \times \ldots s_n \to s$, $f_{\mathscr{A}}$ est une application de $\mathscr{A}_{s_1} \times \ldots \times \mathscr{A}_{s_n} \to \mathscr{A}_s$*

Exemple 2.24 *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 2.25 *Étant donnée une signature $(\mathcal{S}, \mathcal{F}, \tau)$, 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 $f : s_1 \times \ldots s_n \to s$, 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 2.26 *Soient $\mathscr{A}$ et $\mathscr{B}$ deux $\mathcal{F}$-algèbres. Un homomorphisme de $\mathscr{A}$ dans $\mathscr{B}$ est un ensemble d'applications $\{h_s \mid s \in \mathcal{S}\}$ tel que pour toute sorte s de $\mathcal{S}$, h_s est une application de $\mathscr{A}_s$ dans $\mathscr{B}_s$, et pour tout symbole de fonction $f : s_1 \times \ldots \times s_n \to s$*

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

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 $\mathcal{X}$.

2.4.1 Algèbres quotients

Définition 2.27 (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 $f : s_1 \times \ldots \times s_n \to s$, 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 2.28 *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 3

Logique équationnelle

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

Définition 3.1 *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 3.2 *Reprenons l'exemple 2.2 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*

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

3.1 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,

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

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 2.4.1. Il est clair que par définition

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

Proposition 3.3 *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 $\{s = t\}$.

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

Définition 3.4 *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 3.5 *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 3.6 *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 3.7 *Soit $E = \{add(x, Zero) = x;\ \ add(x, Succ(y)) = Succ(add(x, y))\}$. 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 3.8 *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 3.9 *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 3.10 *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 $\mathcal{T}(\mathcal{F})/ =_E \models s = t$.*

3.3 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 :

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

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) :

$$\frac{}{s = s}$$ Réflexivité

$$\frac{s = t}{t = s}$$ Symétrie

$$\frac{s = t \qquad t = u}{s = u}$$ Transitivité

$$\frac{s = t}{u[s\sigma]_p = u[t\sigma]_p}$$ 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 :

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

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

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 3.11 (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 $E \vdash s = t$.

□

3.4 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 $E \models s = t$.

3.5 Exercices

Exercice 3.12 *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*

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

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

$$\begin{aligned} h(add(x, Zero)) &= a(h(x), (0, 1)) \\ &= a((n, m), (0, 1)) \\ &= (n + 0, m) \\ &= (n, m) \\ &= h(x) \end{aligned}$$

$$\begin{aligned} h(add(x, Succ(y))) &= a(h(x), S(h(y))) \\ &= a((n, m), (n' + 1, m')) \\ &= (n + n' + 1, m) \\ &= S(n + n', m) \\ &= S(a((n, m), (n', m'))) \\ &= S(a(h(x), h(y))) \\ &= h(Succ(add(x, y))) \end{aligned}$$

D'autre part, soit h l'assignation définie par

$$h(x) = (0, 2)$$

Cette assignation est telle que

$$\begin{aligned} h(add(Zero, x)) &= a((0, 1), h(x)) \\ &= a((0, 1), (0, 2)) \\ &= (0, 1) \\ &\neq (0, 2) = h(x) \end{aligned}$$

□

Chapitre 4

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 récriture, 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 3. 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.

4.1 Unification syntaxique

Définition 4.1 (Problème d'unification) *Soit $\mathcal{T}(\mathcal{F}, \mathcal{X})$ une algèbre de termes. Un problème d'unification est de la forme*

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

Toute substitution est solution de $\top$, aucune substitution n'est solution de $\bot$, 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 4.2 *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 4.3 *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.*

4.1.1 Formes résolues

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

Définition 4.4 *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 4.5 *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 4.6 *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.*

4.1.2 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 $\quad \dfrac{s = s}{\top}$

Décomposition $\quad \dfrac{f(s_1, \ldots, s_n) = f(t_1, \ldots, t_n)}{s_1 = t_1 \wedge \ldots \wedge s_n = t_n}$

Incompatibilité $\quad \dfrac{f(s_1, \ldots, s_n) = g(t_1, \ldots, t_m)}{\bot}$ si $f \neq g$

Union de deux variables $\quad \dfrac{x = y \wedge P}{x = y \wedge P\{x \mapsto y\}}$ si $x, y \in \mathcal{V}ar(P)$.

Remplacement de variable $\quad \dfrac{x = s \wedge P}{x = s \wedge P\{x \mapsto s\}}$ si $x \in \mathcal{V}ar(P) \setminus \mathcal{V}ar(s)$ et $s \notin \mathcal{X}$.

Fusion $\quad \dfrac{x = s \wedge x = t}{x = s \wedge s = t}$ si $x \in \mathcal{X}$, $s, t \notin \mathcal{X}$ et $|s| \leq |t|$.

Test d'occurrence $\quad \dfrac{x_1 = t_1[x_2]_{p_1} \wedge \ldots x_n = t_n[x_1]_{p_n}}{\bot}$ si $p_1 \cdot \ldots \cdot p_n \neq \Lambda$

4.1.3 Correction

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

Théorème 4.7 *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.
□

4.1.4 Terminaison

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

Théorème 4.8 *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 multi-ensemble 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 multi-ensemble 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 6 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	$\geq$	$>$	

□

4.1.5 Complétude

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

Théorème 4.9 *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 $j$ème 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 $i$ème 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 4.10 *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 4.5 $\mathscr{U}(P')$ 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 :

$$\begin{aligned} \theta &= \theta'\rho' \\ \theta' &= \theta\rho \end{aligned}$$

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.

□

4.1.6 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 :

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

$$\begin{array}{rl} \text{Décomposition} & \dfrac{f(x,y) = f(z,z)}{x = z \wedge y = z} \end{array}$$

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

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

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

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

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

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

4.2 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 4.11 (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 $\mathscr{U}_E(P)$.

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 4.12 (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 $\mathcal{U}_E$ (sur les variables libres de P),*

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 4.13 (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 4.14 *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.

4.3 Unification AC

La théorie de l'associativité-commutativité est celle qui a été le plus étudiée pour l'unification [34, 41, 27, 17, 18, 22, 9, 8, 1, 26, 16]. Nous rappelons que la théorie AC$(+)$ est présentée par les deux axiomes

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

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*.

4.3.1 Formes canoniques et égalité modulo AC

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

Définition 4.15 (Forme aplatie) *La forme aplatie $\overline{t}^a$ d'un terme t est définie de façon récursive comme suit :*

$$
\begin{array}{rcll}
\overline{x}^a & = & x & \text{si } x \text{ est une variable} \\
\overline{f(t_1,\ldots,t_n)}^a & = & f(\overline{t_1}^a,\ldots,\overline{t_n}^a) & \text{si } f \text{ n'est pas un symbole AC} \\
\overline{+(t_1,t_2)}^a & = & \left\{\begin{array}{ll}
+(\overline{t_1}^a,\overline{t_2}^a) & \text{si } t_1(\Lambda)\neq +, t_2(\Lambda)\neq + \\
+(\overline{t_1}^a,v_1,\ldots,v_n) & \text{si } t_1(\Lambda)\neq +, \overline{t_2}^a = +(v_1,\ldots,v_n) \\
+(u_1,\ldots,u_m,\overline{t_2}^a) & \text{si } \overline{t_1}^a = +(u_1,\ldots,u_m), t_2(\Lambda)\neq + \\
+(u_1,\ldots,u_m,v_1,\ldots,v_n) & \text{si } \overline{t_1}^a = +(u_1,\ldots,u_m), \overline{t_2}^a = +(v_1,\ldots,v_n)
\end{array}\right.
\end{array}
$$

Intuitivement, cela revient à ignorer le parenthésage 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 cela un ordre total arbitraire sur l'ensemble des termes.

Définition 4.16 (Forme canonique) *La forme canonique $\overline{t}$ d'un terme t est définie de façon récursive comme suit :*

$$
\begin{array}{rcll}
\overline{x} & = & x & \text{si } x \text{ est une variable} \\
\overline{f(t_1,\ldots,t_n)} & = & f(\overline{t_1},\ldots,\overline{t_n}) & \text{si } f \text{ n'est pas un symbole AC} \\
\overline{+(t_1,t_2)} & = & \left\{\begin{array}{ll}
+(Sort(\overline{t_1},\overline{t_2})) & \text{si } t_1(\Lambda)\neq +, t_2(\Lambda)\neq + \\
+(Sort(\overline{t_1},v_1,\ldots,v_n)) & \text{si } t_1(\Lambda)\neq +, \overline{t_2} = +(v_1,\ldots,v_n) \\
+(Sort(u_1,\ldots,u_m,\overline{t_2})) & \text{si } \overline{t_1} = +(u_1,\ldots,u_m), t_2(\Lambda)\neq + \\
+(Sort(u_1,\ldots,u_m,v_1,\ldots,v_n)) & \text{si } \overline{t_1} = +(u_1,\ldots,u_m), \overline{t_2} = +(v_1,\ldots,v_n)
\end{array}\right.
\end{array}
$$

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 récriture. 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 4.17 *Deux termes s et t sont égaux modulo AC si et seulement si*

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

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

4.3.2 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 aplatis.

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 4.18 *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*

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

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 4.17, 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 4.19 *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 4.17, les formes aplaties de $s\sigma$ et $t\sigma$ ont le même nombre d'occurrences 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 [40, 34, 41] :

Théorème 4.20 (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 4.19, $(u_1^k, \ldots, u_n^k)$ 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

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

□

4.4 Résolution d'équations diophantiennes linéaires

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

$$a_1x_1 + \cdots + a_nx_n = k$$

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*

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

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 [24], améliorée par Lambert [32]. 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 [10], puis, nous verrons comment Contejean et Devie ont étendu cette méthode au cas de systèmes d'équations [12]. Les deux algorithmes ont en outre l'avantage de s'adapter facilement au cas inhomogène.

4.4.1 Algorithme de Clausen-Fortenbacher

Le cas homogène

Notation 4.21 *On notera e_i le $i$ème vecteur de la base canonique de $\mathbb{N}^n$. Étant donné un vecteur $v \in \mathbb{N}^n$, $v(i)$ désigne la $i$ème composante de v.*

Définition 4.22 *Le* défaut *du vecteur $c \equiv (c_1, \ldots, c_n) \in \mathbb{N}^n$ par rapport à l'équation diophantienne linéaire $e \equiv a_1x_1 + \cdots + a_nx_n = 0$ ($a_1, \ldots, a_n \in \mathbb{Z}$) est l'entier relatif $a_1c_1 + \cdots + a_nc_n$. On le note $D(c, e)$ ou plus simplement $D(c)$, s'il n'y a pas d'ambiguïté.*

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 4.23 *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 4.1 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
  - $S_{i+1} = S_i \cup \{v \in M_i \mid D(v, e) = 0\}$
  - $M_{i+1} = \{v + e_i \mid \nexists s \in S_{i+1}$ t.q. $v \geq^n s$, et $D(v, e) \times D(e_i, e) < 0\}$.
- On retourne le premier S_k tel que $M_k = \emptyset$.

FIG. 4.1 – L'algorithme de Clausen-Fortenbacher, pour une équation homogène, première version.

Proposition 4.24 *Étant donné une équation diophantienne linéaire*

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

l'algorithme de la figure 4.1 termine toujours et calcule l'ensemble S des solutions non-nulles, positives, minimales pour $\geq^n$ de e.

Démonstration La complétude découle du lemme 4.23. 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 4.1 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 4.2 où un vecteur n'est jamais engendré deux fois.

Proposition 4.25 *Étant donné une équation diophantienne linéaire $e \equiv a_1x_1 + \cdots + a_nx_n = 0$, l'algorithme de la figure 4.2 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 4.3 montre le fonctionnement de l'algorithme sur l'équation $x + 2y - 3x = 0$.

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

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.

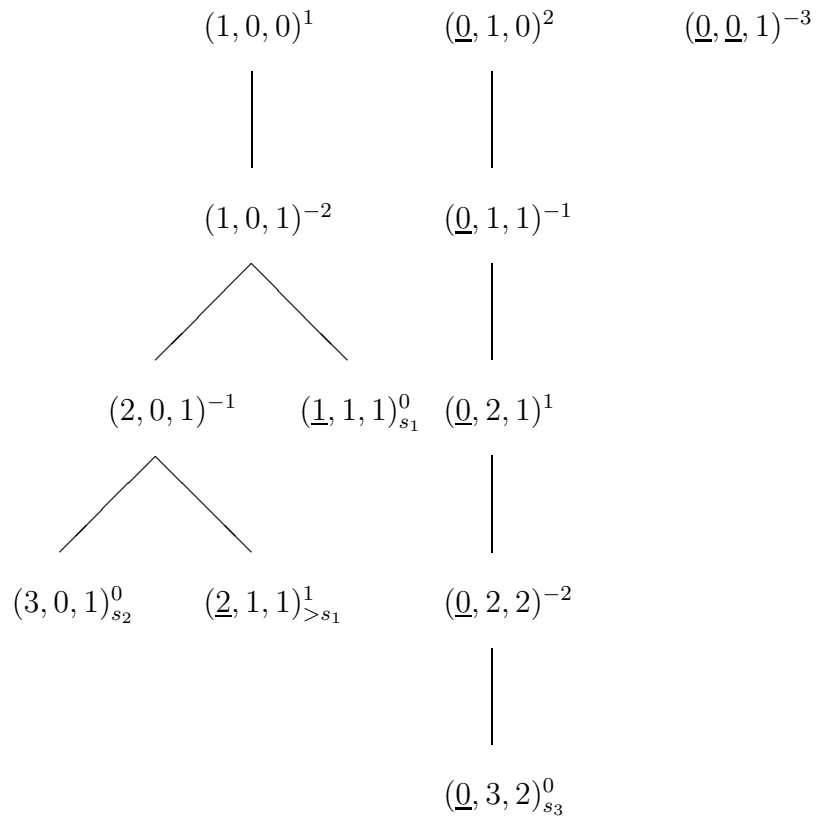
4.4.2 Algorithme de Contejean-Devie

Contejean et Devie ont adapté l'algorithme précédent à la résolution directe de systèmes d'équations, à partir d'une interprétation géométrique simple de l'algorithme de Clausen-Fortenbacher [11, 12].

La notion de défaut s'étend naturellement au cas de systèmes d'équations :

Définition 4.26 *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 :*

$$\begin{array}{ccccccc} a_{11}x_1 & + & \cdots & + & a_{1n}x_n & = & 0 \\ \cdot & & & & \cdot & & \cdot \\ \cdot & & & & \cdot & & \cdot \\ \cdot & & & & \cdot & & \cdot \\ a_{m1}x_1 & + & \cdots & + & a_{mn}x_n & = & 0 \end{array}$$



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. 4.3 – Exemple d'exécution de l'algorithme de la figure 4.2.

où les a_{ij} sont dans $\mathbb{Z}$ est le 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 $(0, \ldots, 0)$.

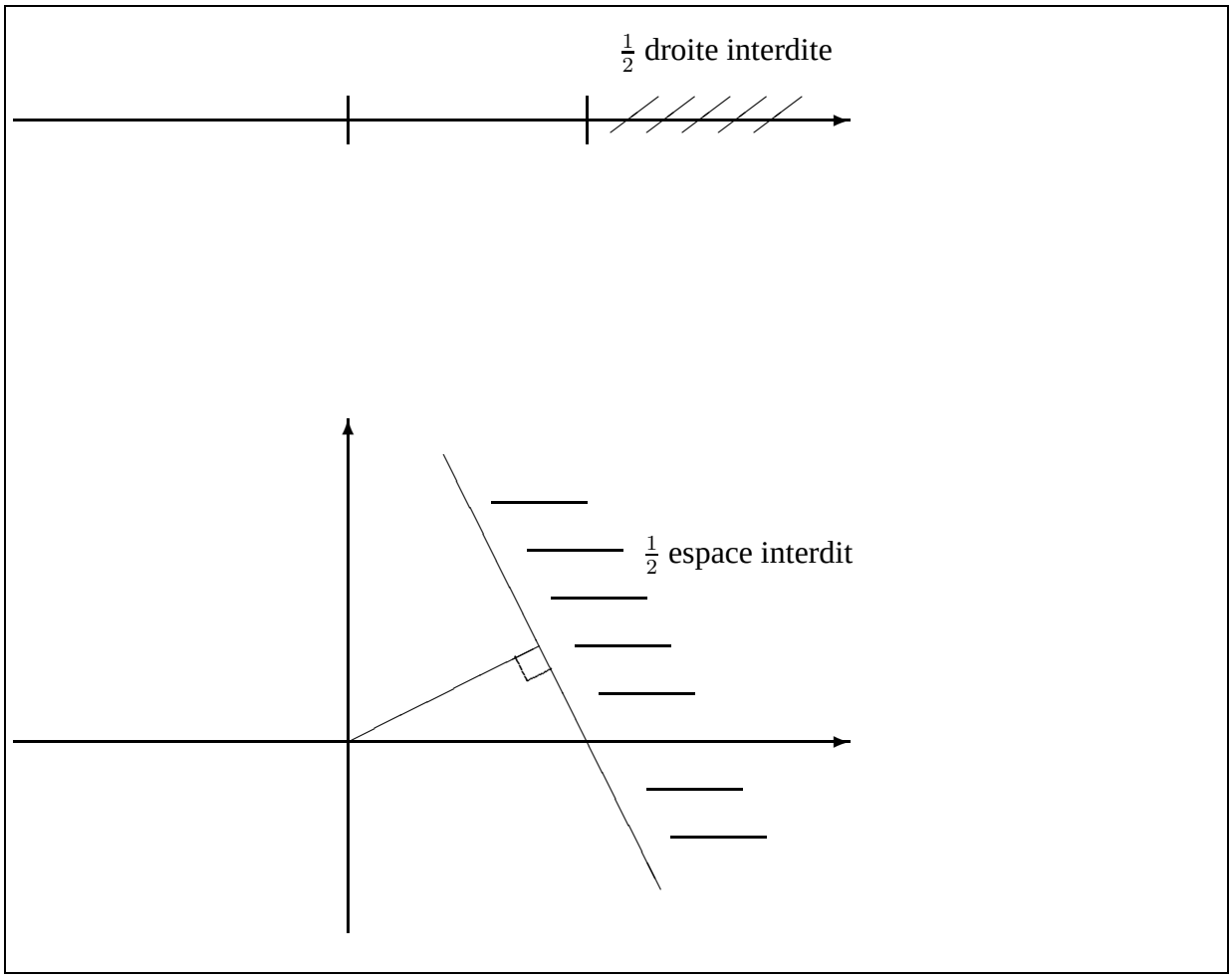


FIG. 4.4 – La restriction géométrique imposée sur les défauts.

La figure 4.4 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 4.27 *Soit $\mathcal{S}$ le système d'équations diophantiennes linéaires*

$$\begin{array}{ccccccc} a_{11}x_1 & + & \cdots & + & a_{1n}x_n & = & 0 \\ \cdot & & & & \cdot & & \cdot \\ \cdot & & & & \cdot & & \cdot \\ \cdot & & & & \cdot & & \cdot \\ a_{m1}x_1 & + & \cdots & + & a_{mn}x_n & = & 0 \end{array}$$

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 $D(e_i) \cdot D(c) < 0$.

Entrée : le système d'équations diophantiennes linéaires S

$$\begin{array}{ccccccc} a_{11}x_1 & + & \cdots & + & a_{1n}x_n & = & 0 \\ \cdot & & & & \cdot & & \cdot \\ \cdot & & & & \cdot & & \cdot \\ \cdot & & & & \cdot & & \cdot \\ a_{m1}x_1 & + & \cdots & + & a_{mn}x_n & = & 0 \end{array}$$

- 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. 4.5 – L'algorithme de Contejean-Devie pour un système d'équations diophantiennes linéaires

On obtient l'algorithme de la figure 4.5 qui est évidemment complet par le lemme 4.27, mais dont la terminaison n'est pas triviale,contrairement au cas d'une équation. On pourra se reporter à [12] pour la preuve de terminaison.

4.5 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. Nous

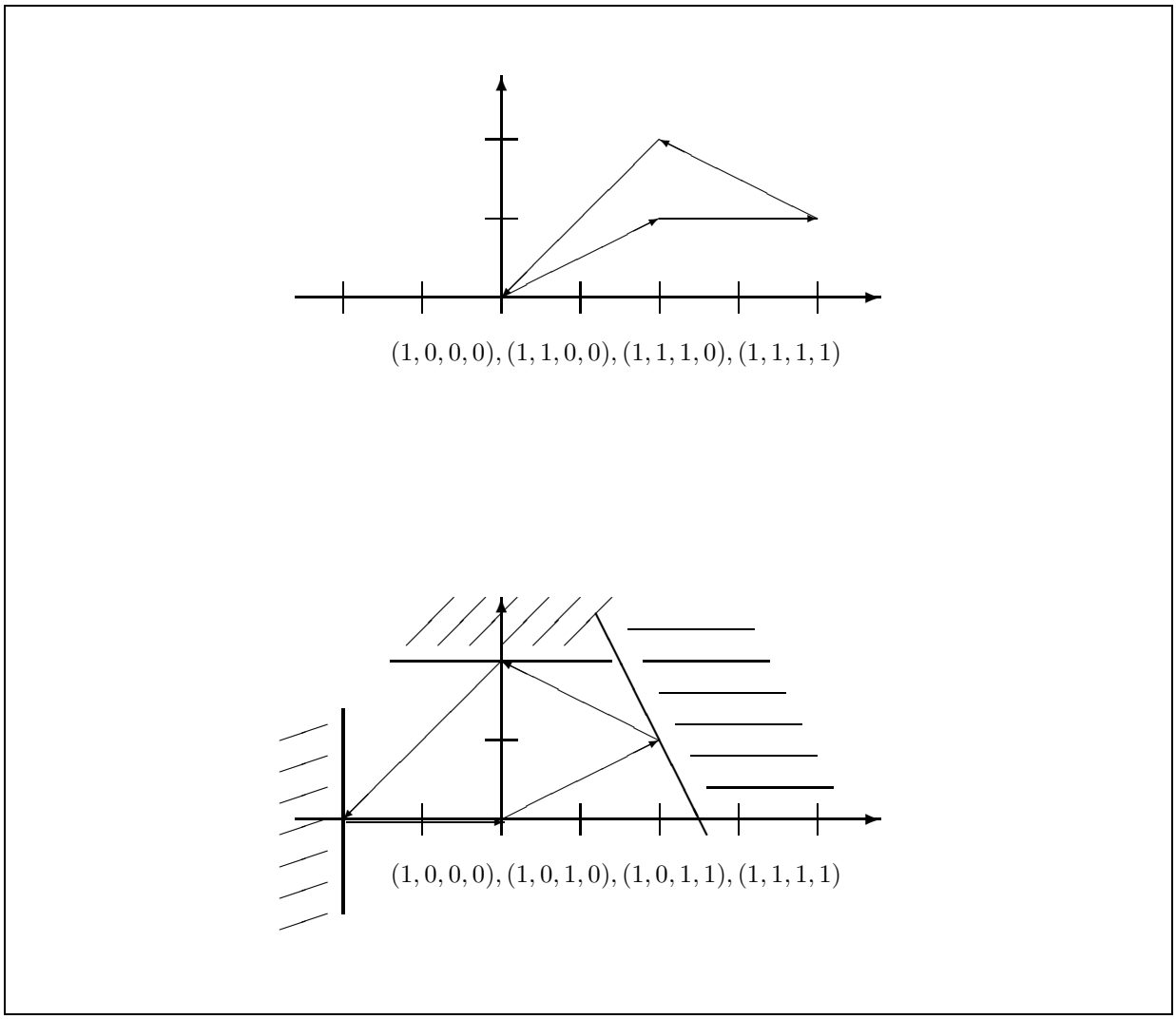


FIG. 4.6 – Deux séquences de défauts correspondant à la solution $(1,1,1,1)$ du système

$$\begin{array}{rcrcrcrcl} 2x_1 & + & 2x_2 & - & 2x_3 & - & 2x_4 & = & 0 \\ x_1 & & & + & x_3 & - & 2x_4 & = & 0 \end{array}$$

Seule la deuxième sera construite par l'algorithme de Contejean-Devie, car la première viole la restriction géométrique.

supposerons que la signature $\mathcal{F}$ est finie, et sans perte de généralité qu'elle est partitionnée en *deux* parties $\mathcal{F}_1 \cup \mathcal{F}_2$, 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 [7]. 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.

4.5.1 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 4.28 (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 $i \neq j$.*

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 4.29 *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 sur les variables de s et t.*

Démonstration La preuve de ce lemme repose sur la complétion close ordonnée. Si γ est une solution de $s = t$ modulo $E_1 \cup E_2$, alors

$$s\gamma =_{E_1 \cup E_2} t\gamma$$

Sans perte de généralité, on pourra considérer toutes les variables qui apparaissent dans cette preuve d'égalité comme des constantes (ce qui entraîne en particulier que γ est vue comme une substitution close). On se donne un ordre de récriture total sur les termes clos (étendus avec les variables « constantifiées » de la preuve $s\gamma =_{E_1 \cup E_2} t\gamma$) ; un tel ordre existe, par exemple RPO. Cet ordre permet de faire tourner une exécution équitable de la complétion ordonnée et d'obtenir un système de récriture (potientiellement infini) R à partir de $E_1 \cup E_2$ tel que

$$s\gamma \longrightarrow_R^* \longleftarrow_R^* t\gamma$$

De plus, comme les symboles de $\mathcal{F}_1$ et $\mathcal{F}_2$ sont disjoints et que les théories sont consistantes (pas d'égalité $x = y$ où x et y sont des variables), il n'y a pas de paires critiques entre E_1 et E_2, et R peut être partitionné en $R_1 \cup R_2$ où R_i ne comporte que des symboles de $\mathcal{F}_i$.

Soit h une bijection des formes normales de $\mathcal{T}(\mathcal{F},\mathcal{X})$ modulo $\rightarrow_R$ dans $\mathcal{X}$ (c'est possible car $\mathcal{X}$ est infini dénombrable, l'ensemble des termes $\mathcal{T}(\mathcal{F},\mathcal{X})$ est dénombrable, et les formes normales de $\mathcal{T}(\mathcal{F},\mathcal{X})$ sont aussi infinies dénombrables, car les variables sont des formes normales deux à deux distinctes). h est étendue à toute l'algèbre $\mathcal{T}(\mathcal{F},\mathcal{X})$ de la façon usuelle :

$$h(t)h(t\downarrow_R)$$

La partie homogène dans $\mathcal{F}_1$ d'un terme t, $\overline{t}^1$ est définie récursivement par :

$$\begin{array}{rcll} \overline{x}^1 & = & h(x) & \text{si } x \text{ est une variable} \\ \overline{f(s_1,\ldots,s_n)}^1 & = & f(\overline{s_1}^1,\ldots,\overline{s_n}^1) & \text{si } f\in\mathcal{F}_1 \\ \overline{f(s_1,\ldots,s_n)}^1 & = & h(f(s_1,\ldots,s_n)) & \text{si } f\notin\mathcal{F}_1 \end{array}$$

Cette définition est étendue aux substitutions :

$$\overline{\sigma}^1=\{x\mapsto\overline{x\sigma}^1 \mid x\in\mathcal{D}om(\sigma)\}$$

De façon duale, on définit la « substitution » universelle ρ_u par

$$\rho_u(h(t))=t\downarrow_R$$

Strictement parlant, ρ_u a un domaine infini, mais ne sera jamais utilisée que sur un sous-domaine fini.

De par la définition de ρ_u, il est clair que

$$\forall t,\ \ \overline{t}^1\rho_u=_{E_1\cup E_2} t$$

Commençons par étendre explictement γ par l'identité sur toutes les variables de s et t qui ne sont pas dans son domaine, et définissons maintenant σ par

$$\sigma=\overline{\gamma}^1$$

Il est clair que σ est une substitution de $\mathcal{T}(\mathcal{F}_1,\mathcal{X})$, et que γ est une $(E_1\cup E_2)$-instance (par ρ_u) de σ sur les variables de s et t.

Montrons que σ est une E_1-solution de $s=t$.

Supposons sans perte de généralité que γ est en forme normale pour R.

Montrons la propriété suivante :

$$\forall s',t',\gamma'\ \ \gamma' \text{ en f.n. } \wedge\, s',t'\in\mathcal{T}(\mathcal{F}_1,\mathcal{X})\wedge s'\gamma'=_{E_1\cup E_2}t'\gamma'\Longrightarrow s'\overline{\gamma'}^1=_{E_1}t'\overline{\gamma'}^1$$

par induction sur la paire $\{s'\gamma',t'\gamma'\}$ avec pour ordre bien fondé l'extension multiensemble de l'ordre utilisé pour la complétion ordonnée.

Si $s\gamma$ et $t\gamma$ sont en forme normale pour R, comme R est convergent et que $s\gamma=_R t\gamma$, $s\gamma$ et $t\gamma$ sont identiques ; leurs parties homogènes le sont aussi et

$$s\sigma=s\overline{\gamma}^1=\overline{s\gamma}^1=\overline{t\gamma}^1=t\sigma$$

Si par exemple $s\gamma$ n'est pas en forme normale, alors choisissons un redex de $s\gamma$ minimal pour l'ordre sous-terme :

$$s\gamma|_p=l\tau$$

avec $l=r\in R$ et $l\tau>r\tau$. Supposons sans perte de généralité que les variables de l et r sont disjointes de celles de s et t (éventuellement en renommant l et r), et que le domaine de τ est exactement égal à l'union des variables de l et r.

Comme γ est en forme normale, $s\gamma(p)\notin\mathcal{F}_2$, et $s\gamma|_p$ n'est pas une une variable (sinon l est une variable, et comme les théories sont cohérentes, r est un terme qui contient l, ce qui contredit $l\tau>r\tau$), donc $s\gamma(p)\in\mathcal{F}_1$ et $l=r$ est une équation de R_1. De plus p est une position de s, car γ est supposée en forme normale.

Comme le redex choisi dans $s\gamma$ est minimal, tous les sous termes stricts de $l\tau$ sont en forme normale. Si r contient une variable x qui n'apparaît pas dans l, on peut choisir de réduire $x\tau$ en forme normale pour R, ce qui donne une substitution τ' définie par

$$\tau' = \{x \mapsto (x\tau) \downarrow_R |\ x \in \mathcal{D}om(\tau)\}$$

τ' est en forme normale et coïncide avec τ sur les variables de l.
On peut maintenant appliquer l'hypothèse d'induction à $s[r]_p$, t et $\gamma \cup \tau'$, donc

$$s\overline{\gamma}^1[r\overline{\tau'}^1]_p =_{E_1} t\overline{\gamma}^1$$

Il reste à montrer que

$$s\overline{\gamma}^1 =_{E_1} s\overline{\gamma}^1[r\overline{\tau'}^1]_p$$

$$\begin{array}{lll}
s\overline{\gamma}^1 = \overline{s\gamma}^1 & & \text{car } s \text{ est dans } \mathcal{T}(\mathcal{F}_1, \mathcal{X}) \\
= \overline{s\gamma[l\tau]_p}^1 & & \\
= s\overline{\gamma}^1[l\overline{\tau}^1]_p & \text{car } p \text{ est une position de } s \text{ et } l \text{ est dans } \mathcal{T}(\mathcal{F}_1, \mathcal{X}) & \\
= s\overline{\gamma}^1[l\overline{\tau'}^1]_p & & \\
=_{E_1} s\overline{\gamma}^1[r\overline{\tau'}^1]_p & & \text{car } l = r \text{ est dans } R_1, \text{ donc } l =_{E_1} r
\end{array}$$

□

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	$\dfrac{\exists z_1, \ldots, z_p\ x = s_1 \wedge x = s_2 \wedge P}{\bot}$	si $x \in \mathcal{X}$, $s_1(\Lambda) \in \mathcal{F}_1$ et $s_2(\Lambda) \in \mathcal{F}_2$.
Conflit-2	$\dfrac{\exists z_1, \ldots, z_p\ s_1 = s_2 \wedge P}{\bot}$	$s_1(\Lambda) \in \mathcal{F}_1$ 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	$\dfrac{\exists y_1, \ldots, y_p\ x_1 = t_1[x_2]_{p_1} \wedge \ldots x_n = t_n[x_1]_{p_n}}{\bot}$	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 débarrasser 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.

4.5.2 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 multi-ensemble $\{SV(P_1), SV(P_2)\}$: 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 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 $x \in Var(P_i)$ tels qu'il existe $y \in Var(P_i)$ 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 occurrence 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.

Exemple 4.30

P_V	P_1	P_2
$x = y$	$x = f(u)$ $y = f(v)$ $z = a$	$u = u_1 + u_2$ $v = v_1 + v_2$

$$SV(P) = \{4, 2\}$$

Var-Rep

P_V	P_1	P_2
$x = y$	$y = f(u)$ $y = f(v)$ $z = a$	$u = u_1 + u_2$ $v = v_1 + v_2$

$$SV(P) = \{2, 2\}$$

E-Res

P_V	P_1	P_2
$x = y$ $u = v$	$y = f(u)$ $z = a$	$u = u_1 + u_2$ $v = v_1 + v_2$

$$SV(P) = \{0, 2\}$$

Var-Rep

P_V	P_1	P_2
$x = y$ $u = v$	$y = f(u)$ $z = a$	$u = u_1 + u_2$ $u = v_1 + v_2$

$$SV(P) = \{0, 0\}$$

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 4.31 E-Res *ne fait pas croître* $SV(P)$.

Démonstration La démonstration se fait en examinant tous les cas possibles pour les variables partagées de P'. Supposons sans perte de généralité que **E-Res** s'applique à P_1, donc

$$\begin{array}{rcll} P & = & \exists \vec{y} & P_V \wedge P_H \wedge P_1 \wedge P_2 \\ P' & = & \exists \vec{y}\, \exists \vec{x'} & P_{V'} \cup P_V \wedge P_H \wedge P'_1 \wedge P_2 \end{array}$$

- Une nouvelle variable x' ne peut être partagée car elle n'apparaît que dans P'_1 par définition d'une forme résolue compacte.
- Si y est une variable partagée de P_2, il existe z une variable de P_2 telle que $y \approx_2^{P'} z$. Toutes les étapes de la chaîne $y \approx_2^{P'} z$ sont possibles dans P' :
  1. si $x_1 =_{P_{V'}} x_2$ alors $x_1 \sim_1^P x_2$ (définition d'une forme résolue compacte),
  2. et si $x_1 \sim_1^{P'} x_2$, soit ni x_1 ni x_2 ne sont des nouvelles variables et $x_1 \sim_1^P x_2$, soit l'une au moins est une nouvelle variable, et la preuve de $y \approx_2^{P'} z$ peut être racourcie.

  Dans tous les cas, y était déjà partagée dans P_2.
- Si y est une variable partagée de P_1, il existe z une variable de P_1 telle que $y \approx_1^{P'} z$. Les étapes $=_{P_V}$ et $\sim_2^{P'}$ étaient des étapes possibles dans P. Isolons la ou les premières étapes (de gauche à droite) de $=_{P_{V'}}$ dans la séquence $y \approx_1^{P'} z$:
  1. Il n'y a aucune étape $=_{P_{V'}}$, $y \approx_1^P z$, et y était déjà partagée.
  2. Il y a au moins une telle étape mais ce n'est pas la plus à gauche, donc $y \approx_1^P x \approx_1^{P'} z$; y était déjà partagée dans P car x est membre d'une équation de $P_{V'}$ donc est une variable de P_1.
  3. Il y a exactement un bloc de $=_{P_{V'}}$, et il est complétement à gauche : $y =_{P_{V'}} x \approx_1^P z$. Dans ce cas x était partagée dans P_1, mais ne l'est plus dans P'_1 car n'apparaît plus dans les variables de P'_1 (forme résolue compacte).
  4. Il y a au moins deux blocs de $=_{P_{V'}}$, le premier étant complétement à gauche : $y =_{P_{V'}} x \approx_1^P x' =_{P_{V'}} x'' \approx_1^{P'} z$. Encore une fois, et pour les mêmes raisons, x et x' étaient partagées dans P_1, mais x ne l'est plus dans P'_1.

  Dans le cas où y est une variable de P_1 nouvellement partagée dans P'_1, elle « efface » au moins une variable partagée de P_1. Si deux variables y_1 et y_2 effacent la même variable, elles sont égales modulo $y =_{P_{V'}}$, et donc par définition d'une forme résolue compacte y_1 et y_2 sont identiques.

□

Lemme 4.32 Var-Rep *ne fait pas croître SV(P). De plus, si Var-Rep est appliqué à une équation* $x = y$ *telle que* x *et* y *apparaissent tous deux dans* P_i*, alors SV(P) décroît strictement.*

Démonstration Supposons que **Var-Rep** remplace toutes les occurrences de x par y sauf dans l'équation $x = y$. Remarquons tout d'abord que :

1. La variable x n'est partagée dans P' car elle n'a plus qu'une seule occurrence.
2. Si $z \approx_i^{P'} z'$, alors $z \approx_i^{P} z'$ car $=_V$ est inchangée et les étapes $v \sim_j^{P'} y$ rendues possibles par le remplacement de x par y peuvent s'expanser en $v \sim_j^{P} x =_V y$.

Examinons maintenant les variables partagées de P'_i :

1. Soit u une variable partagée de P'_i différente de y : u est dans P'_i (et donc dans P_i) et il existe une variable v de P'_i telle que $u \approx_i^{P'} v$. On sait que $u \approx_i^{P} v$. Si v est dans P_i, u est partagée dans P_i, sinon, c'est que v est égale à y, et dans ce cas, u est également partagée dans P_i, cette fois-ci avec x, par la séquence $u \approx_i^{P} v \equiv y =_V x$.
2. Si y est partagée dans P'_i, alors il existe une variable z de P'_i, distincte de y telle que $y \approx_i^{P'} z$, donc $y \approx_i^{P} z$, et z est une variable de P_i distincte de x et y.
   - Si y n'apparaît pas dans P_i, c'est que x apparaît dans P_i, et donc x est partagée dans P_i, mais ne l'est plus dans P'_i (bilan nul).
   - Si y apparaît dans P_i, y était partagée dans P_i. Si x et y apparaissent tous deux dans P_i, tous deux étaient partagés dans P_i, mais x ne l'est plus dans P'_i.

□

4.5.3 Correction

Toutes les règles d'inférence données à la section 4.5.1 sont correctes, c'est-à-dire que si

R $\quad \dfrac{P}{P'} \quad$ si C.

est une règle, si C est satisfaite, toute solution de P' est une solution de P.

Il est facile de vérifier que c'est bien le cas pour toutes les règles, car l'égalité $=_{E_1 \cup E_2}$ est une congruence, et par définition d'une solution dans le cas des problèmes de la forme $\exists x, P$ (pour **EQE**).

4.5.4 Complétude

Toutes les règles d'inférence données à la section 4.5.1 sont complètes, c'est-à-dire que si

R $\quad \dfrac{P}{P'} \quad$ si C.

est une règle, si C est satisfaite, toute solution de P est une solution de P'.

4.5.5 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 5

Récriture

Nous avons vu qu'il est difficile de savoir si deux termes sont égaux en utilisant le remplacement des égaux par des égaux. La récriture 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 récriture

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

Définition 5.1 *Soit $\mathcal{T}(\mathcal{F},\mathcal{X})$ une algèbre de termes. Une règle de récriture 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 récriture R est un ensemble de règles de récriture. 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 récriture 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*

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

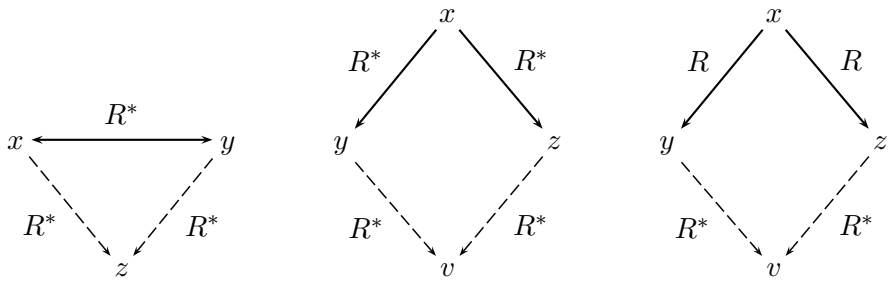
– *est* confluente *si et seulement si*

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

– *est* localement confluente *si et seulement si*

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

Les définitions ci-dessus peuvent se voir plus graphiquement :



Church-Rosser Confluence Confluence locale

Remarque : dans le cas de la récriture, par abus de langage, on dira d'un système de règles de récriture R qu'il 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 $\leftarrow_R$:

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

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

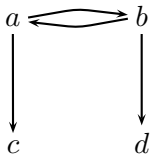
$$x \overset{*}{\underset{R}{\rightarrow}} v' \overset{*}{\underset{R}{\leftarrow}} v \overset{*}{\underset{R}{\leftarrow}} t.$$

□

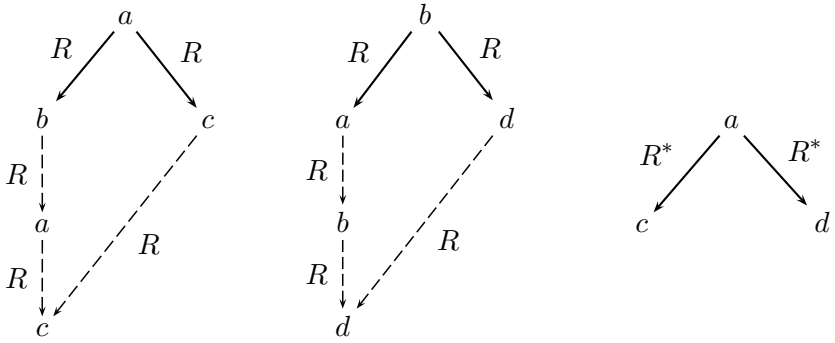
Par contre, la confluence et la confluence locale sont deux notions qui ne sont pas équivalentes. Considérons l'ensemble $\mathscr{A} = \{a, b, c, d\}$ et la relation

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

On a alors le graphe suivant :



La relation est localement confluente, mais pas confluente :



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 récriture R est convergent si la relation associée $\rightarrow_R$ l'est.

[1] Voir les rappels du chapitre 1.

Lemme 5.7 *Soit $\mathscr{A}$ un ensemble et $\rightarrow_R$ 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 1.4) 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 $y'\downarrow$:

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

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

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

  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 [36] :

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

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

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

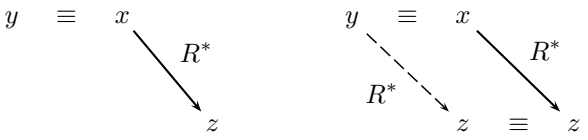
avec

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

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

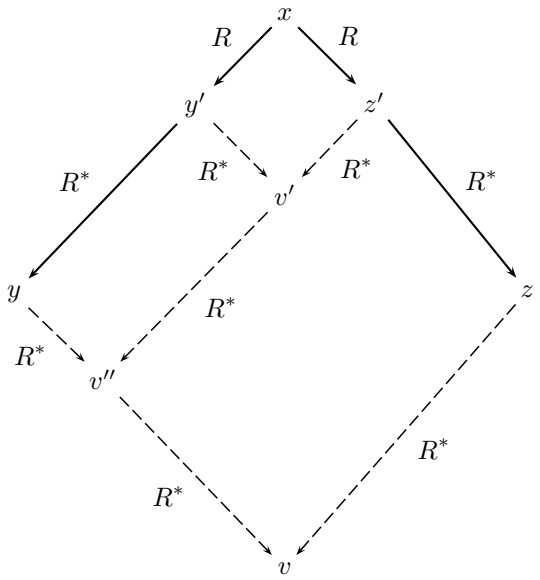
$$y \overset{*}{\underset{R}{\leftarrow}} x \overset{*}{\underset{R}{\rightarrow}} z$$

- Si x n'a pas de prédécesseurs pour $\leftarrow_R$ (*i.e.* pas de successeurs pour $\rightarrow_R$), $P(x)$ 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 $\rightarrow_R$ avec z en un certain nombre de pas, et z en relation par $\rightarrow_R$ avec z en 0 pas : on peut prendre z en guise de v.



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



□

5.3 Paires critiques

Dans le cas où l'on étudie une relation de récriture $\rightarrow_R$ 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 récriture. Une paire critique est une paire de termes obtenue à partir de deux règles de R $\{l \rightarrow r; g \rightarrow d\}$ 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*

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

Considérons la règle de récriture $(x \cdot y) \cdot z \to x \cdot (y \cdot z)$. 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 :

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

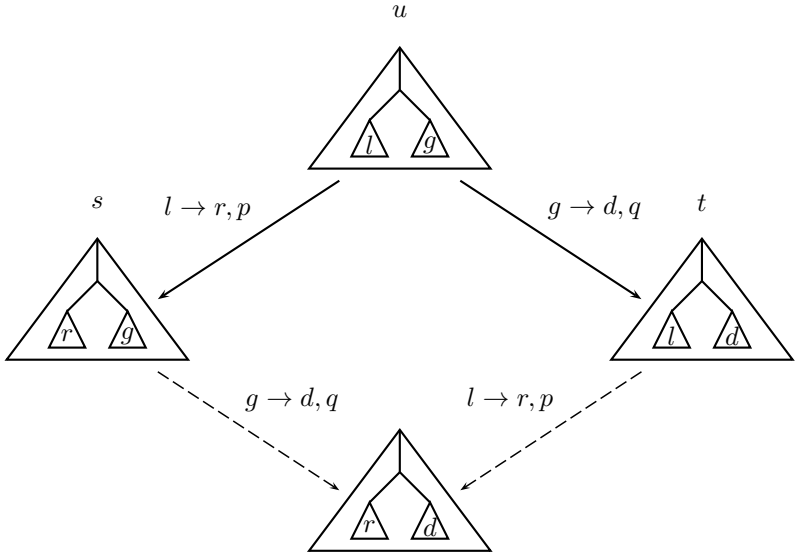
Considérons la règle $x \cdot e \to x$. Cette règle se superpose à la règle $(x \cdot y) \cdot z \to x \cdot (y \cdot z)$ à la racine : $x' \cdot e$ s'unifie à $(x \cdot y) \cdot z$ avec pour unificateur principal $\{x' \mapsto x \cdot y; z \mapsto e\}$. La paire critique est donc :

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

Théorème 5.11 *Un système de récriture 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 récritures commutent, en effet, $s|_q = (u[r\sigma]_p)|_q = u|_q = g\sigma$, donc s peut se récrire 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 récrire 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 récrire 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

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

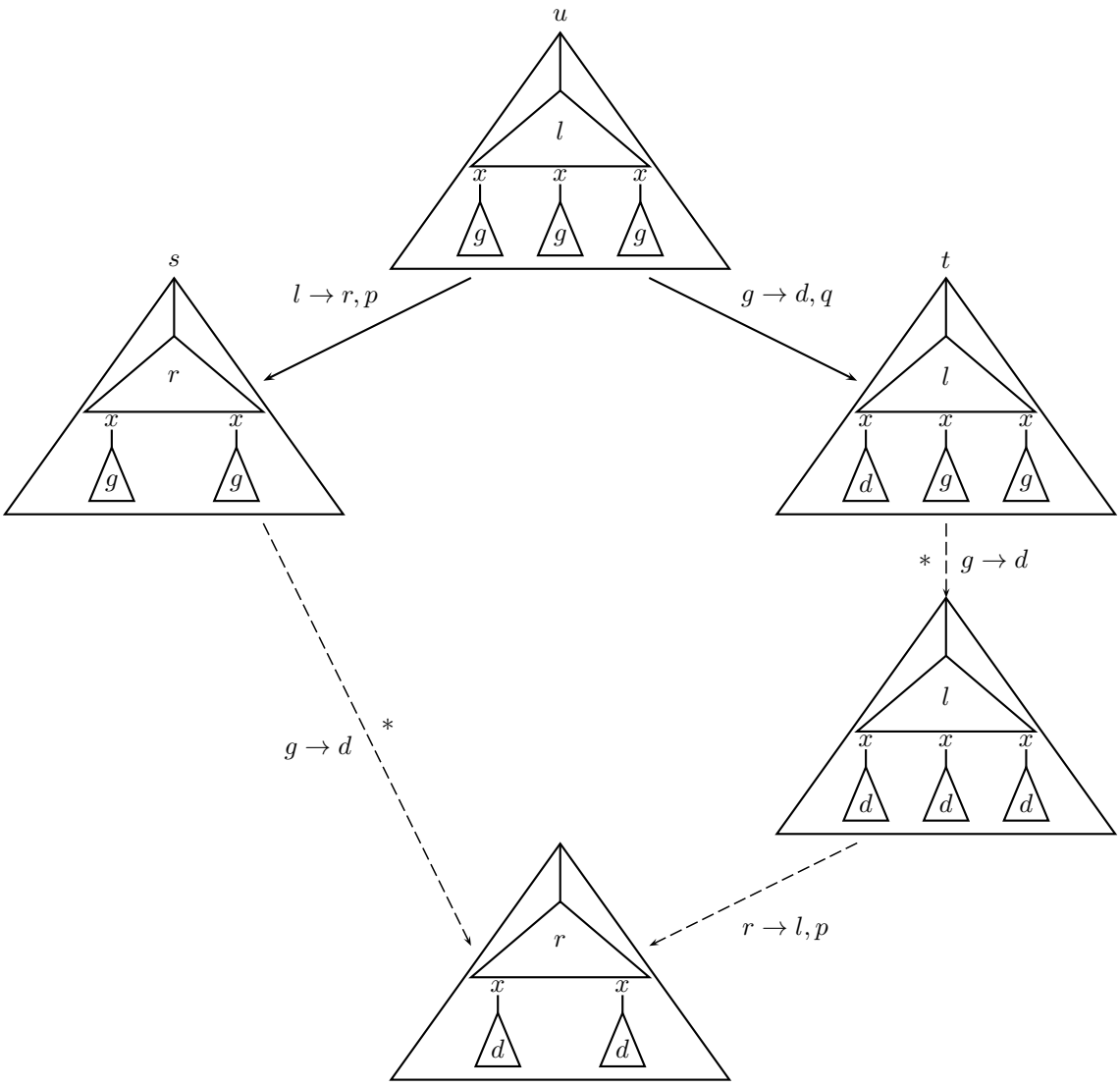
On note σ' la substitution dont le domaine est identique à celui de σ et telle que

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

On obtient alors

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

et bien sûr, $u[l\sigma']_p$ peut se récrire en $u[r\sigma']_p$. Par ailleurs $s \equiv u[r\sigma]_p$ peut se récrire 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 3.9). 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

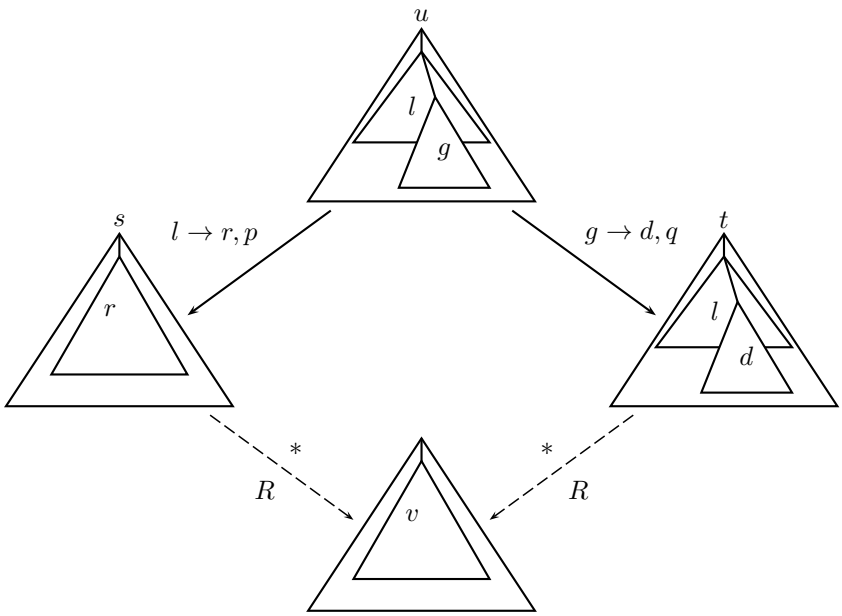
$$\begin{array}{rcl} r\theta & \rightarrow_R^* & v \\ l\theta[d\theta]_{q'} & \rightarrow_R^* & v \end{array}$$

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

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

Finalement, on conclut que

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



□

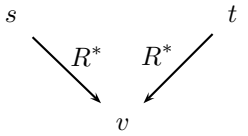
Corollaire 5.12 *Soit R un système de récriture tel que la relation $\leftarrow_R$ est bien fondée. R est convergent si et seulement si pour toute paire critique $s = t$ de R, il existe $s\downarrow$ une forme normale de s et $t\downarrow$ une forme normale de t, telles que $s\downarrow \equiv t\downarrow$. 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

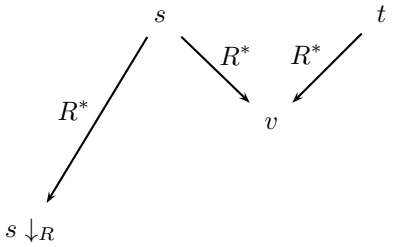
Dans le cas d'un système de récriture R convergent (et fini), le problème du mot relatif à l'ensemble d'équations associé E_R est décidable :

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

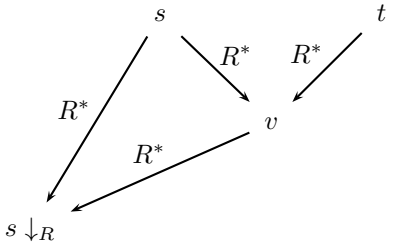
Démonstration Soient s et t deux termes. Il est clair que si leurs formes normales pour $\rightarrow_R$ sont identiques, s et t sont égaux modulo E_R. Supposons maintenant que s et t sont égaux modulo E_R : il existe une séquence d'étapes de $\rightarrow_R \cup \leftarrow_R$ entre s et t, et comme $\rightarrow_R$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



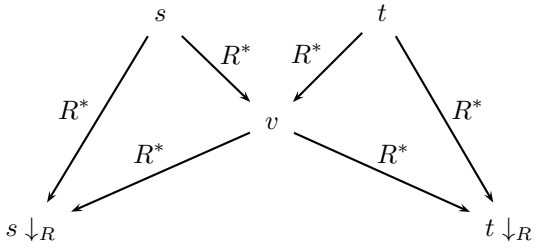
Soit par ailleurs une forme normale $s\downarrow_R$ de s :



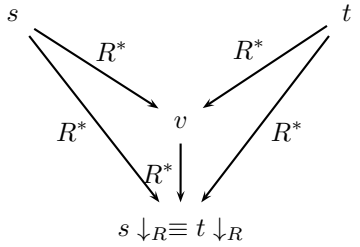
R est confluent, donc il existe une preuve par récriture entre $s\downarrow_R$ et v, et comme $s\downarrow_R$ est en forme normale, v se récrit en $s\downarrow_R$:



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



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



□

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 récriture 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 récriture canoniques

Soit E un ensemble d'équations sur une algèbre de termes $\mathcal{T}(\mathcal{F}, \mathcal{X})$, et soit R un système de récriture tel que E_R et E définissent la même théorie équationnelle. Si R est convergent, on peut trouver un système de récriture 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 récriture 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.*

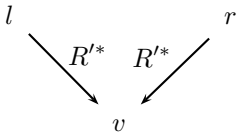
Un système de récriture est canonique si il est convergent et interréduit.

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

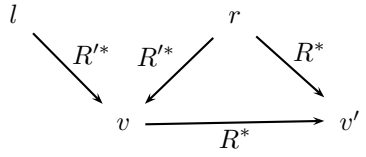
- *les deux relations $=_{E_R}$ et $=_{E_{R'}}$ sont égales.*
- *R et R' sont canoniques.*
- *$\leftarrow_{R \cup R'}$ est bien fondée.*

Alors R et R' sont identiques à renommage près.

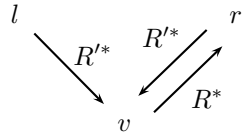
Démonstration Soit $l \to r$ une règle de R. Nous allons montrer que R' contient un renommage de cette règle. l et r sont égaux modulo E_R, donc modulo $E_{R'}$. Comme R' est convergent, il existe une preuve



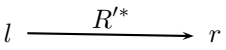
r et v sont égaux modulo $E_{R'}$, donc modulo E_R, comme R est convergent, il existe une preuve en V entre r et v :



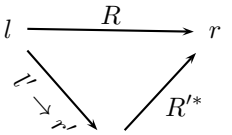
Comme R est interréduit, r est en forme normale pour R, et donc r et v' sont identiques :



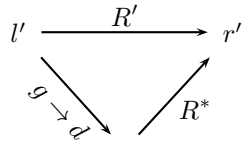
Or $\leftarrow_{R \cup R'}$ est bien fondée, donc r et v sont identiques :



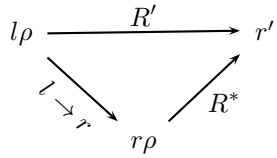
Comme $\leftarrow_R$ est bien fondée, l et r ne sont pas identiques, donc il existe au moins une étape de récriture $\rightarrow_{R'}$ entre l et r. Soit $l' \rightarrow r'$ la règle utilisée lors de la première étape :



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



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 :



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 Récriture modulo

Le principe de la récriture 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 récriture 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 récriture 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 Récriture dans les classes, récriture étendue

Le principe de la récriture 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 [33] :

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

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

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

Pour utiliser effectivement cette récriture, il faut pouvoir énumérer la classe d'un terme modulo S, c'est-à-dire qu'elle soit finie. En outre, cela peut conduire à une combinatoire rédhibitoire. En pratique, on utilise une relation plus faible introduite par Peterson et Stickel [38].

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

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

5.6.2 Confluence de la récriture modulo

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

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*

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

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

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

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

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

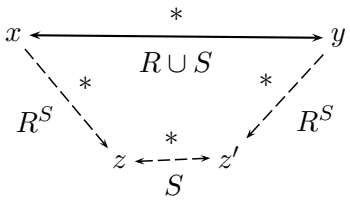
- *$\rightarrow_{R^S}$ est cohérente modulo $\rightarrow_S$ si et seulement si*

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

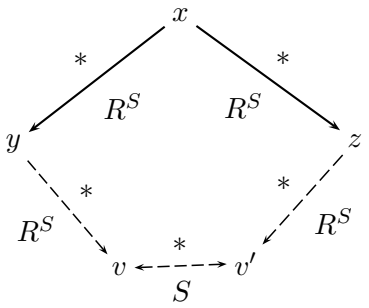
- *$\rightarrow_{R^S}$ est localement cohérente modulo $\rightarrow_S$ si et seulement si*

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

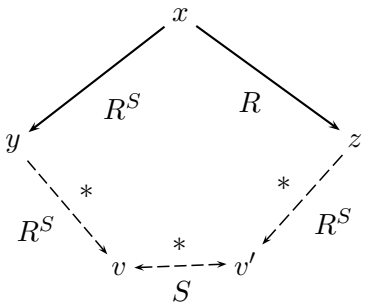
Vision graphique des définitions :



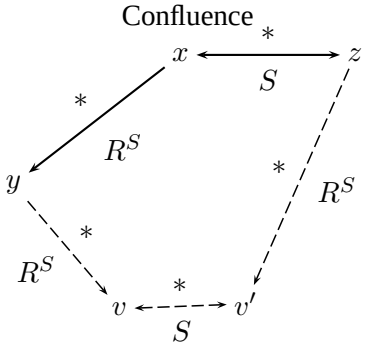
Church-Rosser



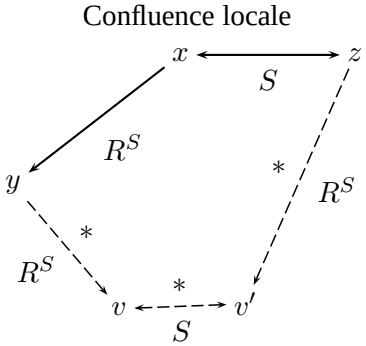
Confluence



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 $\rightarrow_{R^S}$, $x' \longleftrightarrow^*_S y'$.*

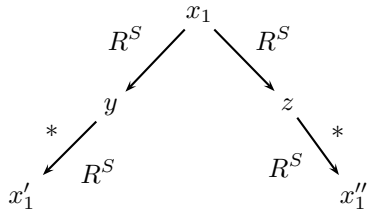
Démonstration Il est évident que $1. \Rightarrow 2. \Rightarrow 3.$.
Montrons maintenant que $3. \Rightarrow 4.$

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

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

par induction bien fondée avec l'extension multi-ensemble 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 :



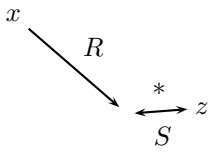
   $\rightarrow_{R^S}$ est comprise entre $\rightarrow_R$ et $\rightarrow_{R/S}$, donc $x_1 \ \rightarrow_{R/S}\ z$. Le pas $x_1 \ \rightarrow_{R/S}\ z$ peut se décomposer par définition en

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

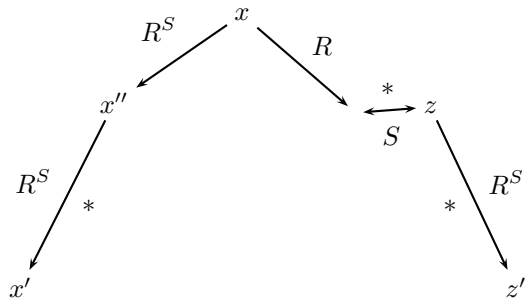
   On va montrer par induction sur m la propriété suivante :

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

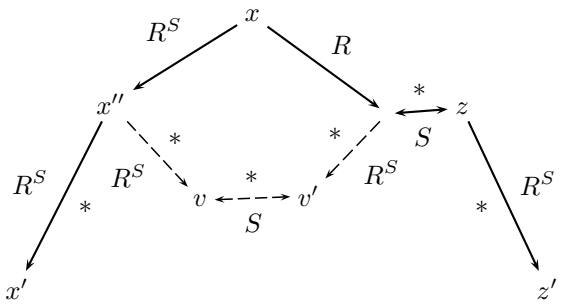
   i. Si $m = 0$, on est dans la situation suivante :



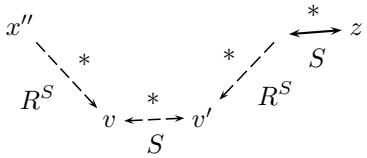
   Soient x' une forme normale de x et z' une forme normale de z, remarquons que comme x se réduit par $\rightarrow_R$, il n'est pas en forme normale pour $\rightarrow_{R^S}$, et qu'il y a donc au moins un pas entre x et x' :



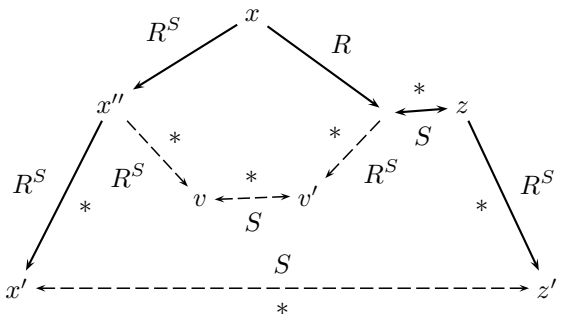
Grâce à la convergence locale, on peut clore en partie le diagramme :



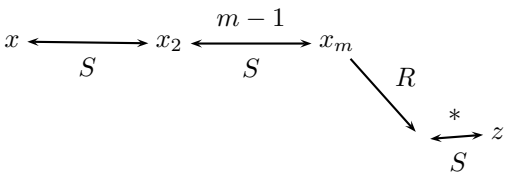
La partie basse du digramme entre x'' et z



peut être expansée en une séquence ne comprenant que des pas $\rightarrow_R$ et $\rightarrow_S$ 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 :



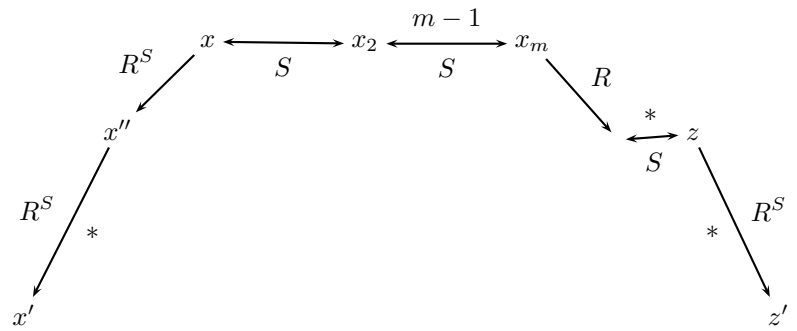
ii. Si $m \geq 1$, on a le digramme suivant :



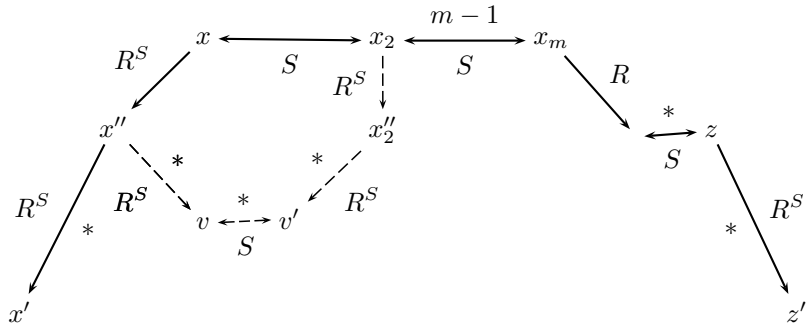
x_2 n'est pas en forme normale, sinon, en appliquant l'hypothèse d'induction $P(m-1)$ avec x_2 et z, on peut conclure que x_2 est S-égal à une forme normale de z, et donc que $\rightarrow_{R/S}$ est cyclique, ce qui contredit la bonne fondation de $\leftarrow_{R/S}$.

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.

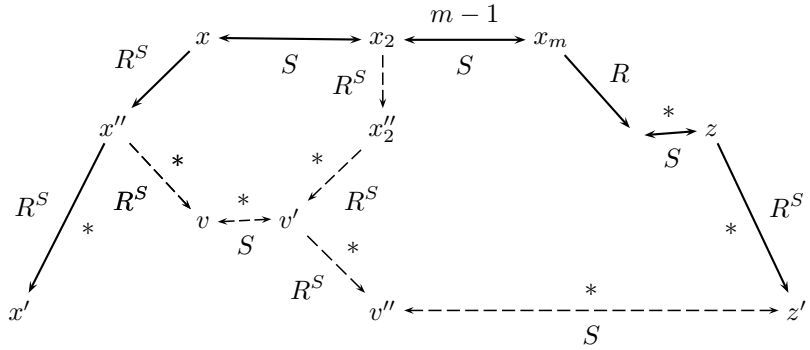
Soient x' une forme normale de x et z' une forme normale de z. D'après les remarques ci-dessus, il y a au moins un pas entre x et x' :



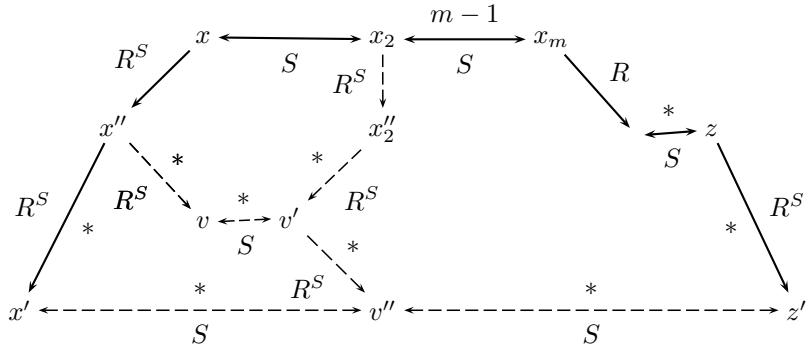
Appliquons la cohérence locale entre x'', x et x_2. Comme indiqué ci-dessus, x_2 n'est pas en forme normale, donc on a le diagramme suivant :



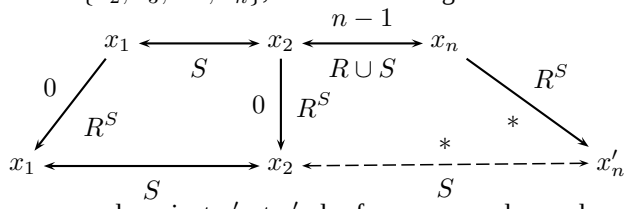
Soit v'' une forme normale de v', en appliquant l'hypothèse d'induction $P(m-1)$ entre x_2 et z, on peut refermer partiellement le diagramme :



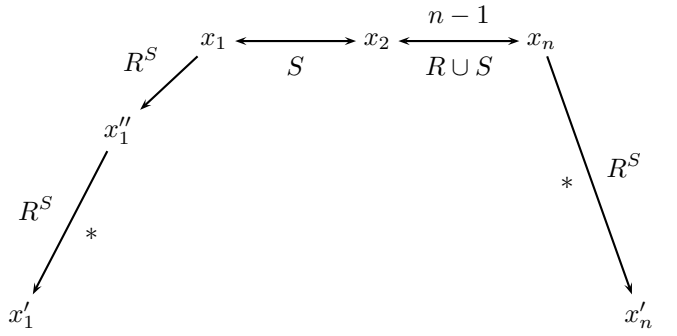
Maintenant, on peut appliquer l'hypothèse d'induction sur la séquence obtenue en expansant la séquence $x'' \ \rightarrow^*_{R^S} \ v \ \longleftrightarrow^*_S \ v''$ en des étapes $\longleftrightarrow_R \cup \longleftrightarrow_S$. En effet tous les éléments qui y apparaissent sont plus petits que x_1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



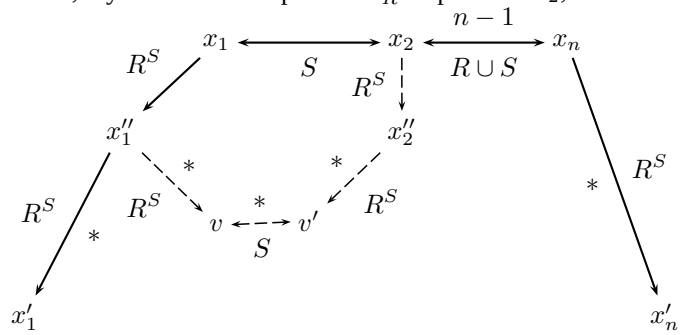
2. Si $n \geq 2$, on va traiter 3 cas, suivant la nature du premier pas entre x_1 et x_2 :
   (a) Si $x_1 \longleftrightarrow_S x_2$, il faut encore distinguer 2 cas :
      i. Si x_1 est en forme normale, comme $\leftarrow R/S$ est bien fondée, et à cause de la cohérence locale, x_2 est également en 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 :



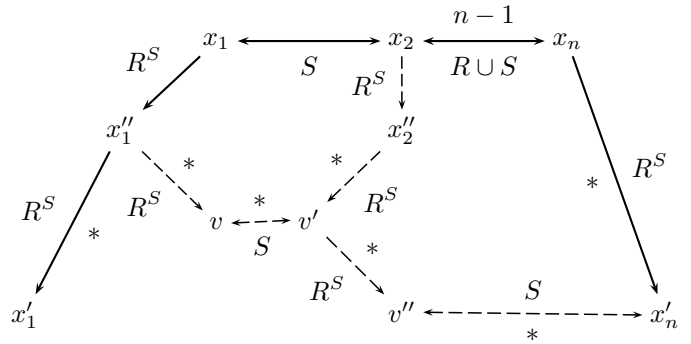
      ii. Si x_1 n'est pas en forme normale, soient x'_1 et x'_n des formes normales quelconques respectivement de x_1 et x_n. En mettant en évidence le premier pas de réduction à partir de x_1, on obtient :



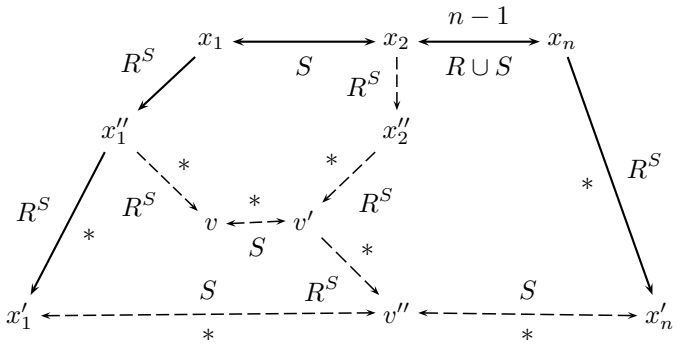
      Par cohérence locale, on referme partiellement le diagramme entre x''_1, x_1 et x_2. Par le même raisonnement que ci-dessus, il y a au moins un pas de $\rightarrow_{R^S}$ à partir de x_2, et on le met en évidence :



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



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



(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 Récriture 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

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

Dans ce cas particulier, la cohérence locale avec la commutativité est toujours acquise. Si l'étape (C) a lieu strictement au dessus de $\to_{\text{AC}\setminus R}$, $\to_{\text{AC}\setminus R}$ 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 récriture par $\to_R$.

La cohérence locale avec l'associativité est assurée dès lors que la récriture $\to_{\text{AC}\setminus R}$ a lieu dans les variables de (A), ou que (A) a lieu en dessous de $\to_{\text{AC}\setminus R}$. 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,

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

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 nécessaire d'étendre les extensions, car l'extension d'une extension est une instance modulo AC de l'extension de niveau 1. Si R est un système de récriture, on note $\mathcal{E}xt(R)$ l'union de l'ensemble R et de ses extensions.

En pratique, on travaille sur des termes aplatis (cf. Définition 4.15) ou même en forme canonique (cf. Définition 4.16) qui sont les structures naturellement manipulées par les algorithmes d'unification et de

filtrage modulo AC. De façon classique, le terme $a + b$ est un sous-terme de $(a + b) + c$. Dans le monde aplati, $a+b$ est un sous-terme de $a+b+c$. De façon moins directe, $a+c$ est aussi un sous-terme de $a+b+c$ (car $a + c$ est un sous-terme de $(a + c) + b$ dont la forme canonique est $a + b + c$).

Il est possible de définir des notions de position $\bullet|_{\bullet}^{\text{AC}}$ et et de sous-terme $\lhd_{\text{AC}}$ sur les termes aplatis qui soit compatibles avec l'égalité modulo AC et cohérentes avec la notion usuelle, c'est-à-dire telles que

$$\begin{array}{l}
\forall s\ t\ p, s = t|_p \Longrightarrow (\exists p', \overline{\overline{s}} = \overline{\overline{t}}|_{p'}^{\text{AC}}) \\
\forall s\ t\ p, s = t|_p \Longrightarrow (\exists p', \overline{s} = \overline{t}|_{p'}^{\text{AC}}) \\
\forall s\ t\ q, \overline{\overline{s}} = \overline{\overline{t}}|_{q}^{\text{AC}} \Longrightarrow (\exists s'\ t'\ q', s =_{\text{AC}} s' \wedge t =_{\text{AC}} t' \wedge s' = t'|_{q'}) \\
\forall s\ t\ q, \overline{s} = \overline{t}|_{q}^{\text{AC}} \Longrightarrow (\exists s'\ t'\ q', s =_{\text{AC}} s' \wedge t =_{\text{AC}} t' \wedge s' = t'|_{q'}) \\
\forall s\ t, s \lhd t \Longleftrightarrow \overline{\overline{s}} \lhd_{\text{AC}} \overline{\overline{t}} \Longleftrightarrow \overline{s} \lhd_{\text{AC}} \overline{t}
\end{array}$$

Heureusement les règles étendues permettent d'éviter cette complication. En conservant *la notion usuelle de position*, mais sur les termes aplatis ou les formes canoniques, on définit

$$\overline{s} \xrightarrow[\text{AC}\backslash R]{} \overline{t} \text{ si et seulement si } \exists l \to r \in R\ \ \exists p \in \mathcal{P}os(\overline{s})\ \ \exists \sigma\ \ \overline{s}|_p =_{\text{AC}} \overline{l\sigma}\ \ \wedge\ \ \overline{t} \equiv \overline{s[r\sigma]_p}$$

Étant donné un système de récriture R, on utilise la relation $\xrightarrow[\text{AC}\backslash\mathcal{E}xt(R)]{}$, par définition cohérente avec AC. De plus sur les systèmes de récriture clos par extension, les deux notions de récriture étendue coïncident :

$$\forall R\ s\ t, s \to_{\text{AC}\backslash\mathcal{E}xt(R)} t \Longleftrightarrow \overline{s} \xrightarrow[\text{AC}\backslash\mathcal{E}xt(R)]{} \overline{t}$$

Chapitre 6

Ordres sur les termes

Définition 6.1 *Un préordre $\preceq$ 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 6.2 *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 6.3 *Soit $\mathcal{T}(\mathcal{F}, \mathcal{X})$ une algèbre de termes, et $\preceq$ un préordre sur cette algèbre. $\preceq$ est un préordre de récriture si :*

- *$\prec$ et $\simeq$ sont monotones,*
- *$\prec$ et $\simeq$ sont stables,*
- *$\prec$ est bien fondé.*

Si on a besoin de distinguer les propriétés de relation strictes et larges, il est commode d'utiliser la notion de *paires d'ordres*. Nous donnons une version simplifiée des *weak reduction pairs* de Kusakari, Nakamura et Toyama [30].

Définition 6.4 *Une* paire d'ordres *est un couple de relations $(\succeq, >)$ tel que*

1. *$\succeq$ est un préordre,*
2. *$>$ est un ordre,*
3. *$\succeq \cdot > = >$.*

Une paire d'ordres est dite :

- Bien fondée *si $>$ est bien fondé ;*
- Stable *si $>$ et $\succeq$ sont stables par application de substitution ;*
- Faiblement monotone *si $\succeq$ est monotone ;*
- Strictement monotone *(ou plus simplement* monotone*) si $>$ et $\succeq$ sont tous deux monotones.*

Parmi les ordres couramment utilisés sur les termes, on peut distinguer deux grandes familles : les ordres *sémantiques* qui reposent sur une interprétation des termes et les ordres *syntaxiques* qui restent dans l'algèbre des termes.

6.1 Interprétations

Considérons un domaine D non vide et muni d'un ordre $\geq_D$ avec $>_D = \geq_D - \leq_D$. Une fonction φ des termes clos vers D induit des relations $\succeq_\varphi$ et $>_\varphi$ définies par $s \succeq_\varphi t$ si et seulement si $\varphi(s) \geq_D \varphi(t)$ et $s >_\varphi t$ si et seulement si $\varphi(s) >_D \varphi(t)$. On obtient facilement que $(\succeq_\varphi, >_\varphi)$ est bien fondée si $>_D$ est bien fondé.

La situation est plus compliquée pour les termes non clos. Les termes non clos vont être interprétés par des fonctions associant toutes D-valuations de ses variables à un élément de D. L'interprétation $\varphi(t)$ est maintenant une fonction $(X \to D) \to D$. On définit sur ces fonctions les relations :

$$f \succeq_{D,X} g \text{ si et seulement si } \forall v \in X \to D, f(v) \geq_D g(v)$$
$$f >_{D,X} g \text{ si et seulement si } \forall v \in X \to D, f(v) >_D g(v)$$

Remarque 6.5 *Il est important de remarquer ici que la relation $>_{D,X}$ n'est* pas *la partie stricte de $\succeq_{D,X}$.*

Contre-exemple 6.6 *Si une constante a est associée à l'élément minimal de D, on a bien pour toute variable x que $x \succeq_{D,X} a$ et $x \not\preceq_{D,X} a$. Ainsi, $(x, a) \in \succeq_{D,X} - \preceq_{D,X}$ mais $x \not\succ_{D,X} a$.*

L'interprétation sur les termes non clos induit les relations $\succeq_\varphi$ et $>_\varphi$ définies par $s \succeq_\varphi t$ si et seulement si $\varphi(s) \succeq_{D,X} \varphi(t)$ et $s >_\varphi t$ si et seulement si $\varphi(s) >_{D,X} \varphi(t)$.

Finalement, $(\succeq_\varphi, >_\varphi)$ est une paire d'ordre bien fondée si $>_D$ est bien fondé.

Définition 6.7 *Une* interprétation homomorphique *φ est une interprétation qui à tout symbole f d'arité n associe une fonction $[\![f]\!]_\varphi : D^n \to D$. On étend cette définition aux termes de la façon suivante : pour toute valuation $v : X \to D$*

$$\begin{aligned} \varphi(x)(v) &= v(x) \quad \textit{pour } x \in X \\ \varphi(f(t_1, \ldots, t_n))(v) &= [\![f]\!]_\varphi(\varphi(t_1)(v), \ldots, \varphi(t_n)(v)) \end{aligned}$$

On pourra n'écrire que $[\![f]\!]$ en l'absence d'ambiguïté sur φ.

Les interprétations homomorphiques se comportent bien vis-à-vis des substitutions.

Lemme 6.8 *Pour une interprétation homomorphique φ, la paire d'ordre $(\succeq_\varphi, >_\varphi)$ est stable.*

Démonstration On définit pour toute substitution σ et toute valuation v la valuation $\varphi(\sigma, v)$ qui à toute variable x associe $\varphi(x\sigma)(v)$. On montre par induction structurelle que pour tout terme t, $\varphi(t\sigma)(v) = \varphi(t)\varphi(\sigma, v)$. Il faut montrer que si $s >_\varphi t$ alors $s\sigma >_\varphi t\sigma$ pour tout σ. Par définition, pour toute valuation v, $\varphi(s)(v) >_D \varphi(t)(v)$ mais $\varphi(s\sigma)(v) = \varphi(s)\varphi(\sigma, v) >_D \varphi(t)\varphi(\sigma, v) = \varphi(t\sigma)(v)$, c'est-à-dire $s\sigma >_\varphi t\sigma$.
□

Enfin on obtient immédiatement :

Lemme 6.9 *Pour tout symbole f d'arité n et $1 \leq i \leq n$, si pour tous $d_1, \ldots, d_{i-1}, d_{i+1}, \ldots, d_n \in D$ on a $[\![f]\!](d_1, \ldots, d_{i-1}, x, d_{i+1}, \ldots, d_n)$ monotone croissante (resp. strictement) en x, alors $\succeq_\varphi$ (resp. $>_\varphi$) est monotone en le i^e argument de f.*

On obtient différents types d'interprétations suivant le domaine choisi. Nous allons nous intéresser ici aux interprétations polynomiales.

Interprétations polynomiales

Définition 6.10 *Soit $D_\mu = \{x \in \mathbb{Z} | x \geq \mu\}$ et soit $>$ l'ordre naturel sur les entiers. Une interprétation* arithmétique *est une interprétation sur D_μ.*

Une interprétation polynomiale *est une interprétation homomorphique arithmétique φ telle que pour tout symbole f, $[\![f]\!]_\varphi$ est une fonction polynomiale.*

Exemple 6.11 *La taille sur les termes est une interprétation polynomiale particulière définie par*

$$[\![f]\!](X_1, \ldots, X_n) = 1 + X_1 + \ldots + X_n$$

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

Exemple 6.12 *Considérons $\mathcal{F}$, une signature monosortée contenant une constante e, un symbole unaire I, et un symbole binaire $\cdot$ (infixe), et le système de récriture 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. 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.*

Dans le cadre de preuves de terminaison, il faudra savoir vérifier que chaque fonction polynomiale associée à un symbole d'arité n est bien de D_μ^n vers D_μ (il suffit par exemple de vérifier pour un polynôme P que $P - \mu$ est positif) et qu'elle vérifie les bonnes propriétés de croissance en chacun de ses arguments. Il faudra aussi être capable de comparer deux interprétations. La comparaison d'équations diophantiennes étant indécidable (c'est le 10^e problème de Hilbert), on devra faire appel à des méthodes incomplètes.

6.2 Comment combiner des ordres

6.2.1 Combinaison lexicographique

Définition 6.13 *Soient $\preceq_1, \ldots, \preceq_n$ n préordres 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 $\prec$:*

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

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

Proposition 6.14 *La composée lexicographique $(\preceq_1, \ldots, \preceq_n)_{lex}$ de n préordres est un préordre ; la relation d'équivalence associée est $\simeq$ et l'ordre strict est $\prec$ donnés par la définition 6.13. 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 préordres, alors $\preceq$ est un préordre :
  - $\preceq$ 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 :

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

  Soit donc H_1 l'hypothèse d'induction

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

  On veut montrer

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

  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

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

  Soit donc H_2 l'hypothèse d'induction

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

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

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

  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.

□

6.2.2 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 6.15 *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 $\forall e \in \mathscr{D}\ \ \emptyset(e) = 0$.*

Définition 6.16 *Soit $\preceq$ un préordre 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 $\preceq^1_{mul}$=$\simeq^1_{mul} \cup \prec^1_{mul}$*

$$M \cup \{x\} \simeq^1_{mul} M \cup \{y\} \text{ si } x \simeq y$$
$$M \cup \{y_1, \ldots, y_n\} \prec^1_{mul} M \cup \{x\} \text{ si } \forall j \in \{1, \ldots, n\}\ \ y_j \prec x$$

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

Proposition 6.17 *$\prec_{mul}$ 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 6.18 *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 construits 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 6.18, 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.

□

6.3 RPO : ordre récursif sur les chemins

Outre les interprétations polynomiales, il existe un type d'ordre sur les termes bien adapté à la mécanisation, il s'agit de l'ordre récursif sur les chemins : RPO (*Recursive Path Ordering*).

Définition 6.19 *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. $\exists t_j \;\; s \preceq_{rpo} t_j$
2. $f \prec g$ *et* $\forall i \in \{1, \ldots, n\} \;\; s_i \prec_{rpo} t$
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* $\forall i \in \{1, \ldots, n\} \;\; s_i \prec_{rpo} t$.

Nous allons montrer que $\preceq_{rpo}$ est un préordre de récriture.

Lemme 6.20 *$\preceq_{rpo}$ 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 6.19.3 ou 6.19.4 suivant le statut du symbole de tête de s.
□

Lemme 6.21 *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 6.19.1, 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 6.19.2 ou 6.19.4, alors $s_i \prec_{rpo} t$ par définition de l'ordre.
- Si $s \preceq t$ par le cas 6.19.3, 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 6.19.1. 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 6.22 *Si $s \preceq_{rpo} t$ par les cas 6.19.1 ou 6.19.2, alors $s \prec_{rpo} t$*

Démonstration La démonstration se fait par contradiction. Supposons que $t \preceq_{rpo} s$. Par le lemme 6.21, $t_j \prec_{rpo} s$ pour tous les sous-termes immédiats t_j de t.

Si $s \preceq_{rpo} t$ par le cas 6.19.1, il existe j_0 tel que $s \preceq_{rpo} t_{j_0}$, d'où une contradiction.

Si $s \preceq_{rpo} t$ par le cas 6.19.2, 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 6.19.1. D'après le cas précédent, $t \prec_{rpo} s$, ce qui contredit $s \preceq_{rpo} t$.
□

Proposition 6.23 *$\preceq_{rpo}$ 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 6.19.1, 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 6.19.1, 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 6.19.1, on conclut $s \preceq_{rpo} u$.
- Si $t \preceq_{rpo} u$ par le cas 6.19.2, $g \prec h$ et pour tous les t_j, $t_j \prec_{rpo} u$.
  - Si $s \preceq_{rpo} t$ par les cas 6.19.2, 6.19.3 ou 6.19.4, $f \preceq g$, et donc $f \prec h$. De plus, par le lemme 6.21, $s_i \prec_{rpo} t$, 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 6.21 ; donc $s_i \prec_{rpo} u$, et $s \preceq_{rpo} u$ par le cas 6.19.2.
- Si $t \preceq_{rpo} u$ par le cas 6.19.3, $g \simeq h$ et $\{t_1, \ldots, t_m\}(\preceq_{rpo})_{mul}\{u_1, \ldots, u_p\}$.
  - Si $s \preceq_{rpo} t$ par le cas 6.19.2, $g \prec h$, et donc $f \prec h$. De plus, par le lemme 6.21, $s_i \prec_{rpo} t$, 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 6.21 ; donc $s_i \prec_{rpo} u$, et $s \preceq_{rpo} u$ par le cas 6.19.2.
  - Si $s \preceq_{rpo} t$ par le cas 6.19.3, $f \simeq g$ 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 6.19.3.
  - Le cas $s \preceq_{rpo} t$ par 6.19.4 est impossible car $f \simeq g \simeq h$ ne peut avoir à la fois le statut lexicographique et multi-ensemble.
- Si $t \preceq_{rpo} u$ par le cas 6.19.4, $g \simeq h$, $(t_1, \ldots, t_m)(\preceq_{rpo})_{lex}(u_1, \ldots, u_p)$ et pour tous les t_j, $t_j \prec_{rpo} u$.
  - Si $s \preceq_{rpo} t$ par le cas 6.19.2, $g \prec h$, et donc $f \prec h$. De plus, par le lemme 6.21, $s_i \prec_{rpo} t$, 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 6.21 ; donc $s_i \prec_{rpo} u$, et $s \preceq_{rpo} u$ par le cas 6.19.2.
  - Le cas $s \preceq_{rpo} t$ par 6.19.3 est impossible car $f \simeq g \simeq h$ ne peut avoir à la fois le statut lexicographique et multi-ensemble.

– Si $s \preceq_{rpo} t$ par le cas 6.19.4, $f \simeq g$, $(s_1, \ldots, s_n)(\preceq_{rpo})_{lex}(t_1, \ldots, t_m)$ 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 6.19.4.

□

Proposition 6.24 *$\prec_{rpo}$ 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 6.19.3 ou 6.19.4 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 6.19.1, 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 6.19.2 est impossible car u_s et u_t ont le même symbole de tête.
– Si $u_t \preceq_{rpo} u_s$ par le cas 6.19.3, $\{u_1, \ldots, u_{i-1}, t, u_{i+1}, \ldots, u_n\}(\preceq_{rpo})_{mul}\{u_1, \ldots, u_{i-1}, s, u_{i+1}, \ldots, u_n\}$, 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 6.19.4, on conclut de même.

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

□

Proposition 6.25 *$\prec_{rpo}$ 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 6.19.1, 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 6.19.1. D'après le corollaire 6.22, $s\sigma \prec_{rpo} t\sigma$.
– si $s \preceq_{rpo} t$ par le cas 6.19.2, $f \prec g$, 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 6.19.2. D'après le corollaire 6.22, $s\sigma \prec_{rpo} t\sigma$.
– si $s \preceq_{rpo} t$ par le cas 6.19.3, $f \simeq g$, 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 6.19.3. 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 6.19.3. Ça n'est pas possible non plus par les cas 6.19.1 et 6.19.2, 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 6.19.4 n'est pas à envisager.
– si $s \preceq_{rpo} t$ par le cas 6.19.4, $f \simeq g$, $(s_1, \ldots, s_n)(\preceq_{rpo})_{lex}(t_1, \ldots, t_m)$, 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 6.19.4. 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 6.19.4. Ça n'est pas possible non plus par les cas 6.19.1 et 6.19.2, 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 6.19.3 n'est pas à envisager.

□

Proposition 6.26 *$\prec_{rpo}$ est bien fondé.*

La démonstration est basée sur le lemme suivant :

Lemme 6.27 *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

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

et $\lessdot_{rpo}$ l'ordre défini sur $\mathscr{A}$ par

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

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

Nous allons montrer

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

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

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

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

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

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

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

- Si $s \prec_{rpo} f(t_1,\ldots,t_n)$ par le cas 6.19.1, 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 6.19.2, 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 $(g,(s_1,\ldots,s_m)) \lessdot_{rpo} (f,(t_1,\ldots,t_n)$, 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 6.19.3, 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 $(g,(s_1,\ldots,s_m)) \lessdot_{rpo} (f,(t_1,\ldots,t_n)$, 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 6.19.4, 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 $(g,(s_1,\ldots,s_m)) \lessdot_{rpo} (f,(t_1,\ldots,t_n)$, donc par H_1, $g(s_1,\ldots,s_m)$ est accessible pour $\prec_{rpo}$.

□

Proposition 6.28 *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 6.27, t est aussi accessible pour $\prec_{rpo}$.
□

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

Théorème 6.29 *$\preceq_{rpo}$ est un préordre de récriture.*

Exemple 6.30 *Considérons la même signature et le même système de récriture qu'à l'exemple 6.12. Soit le RPO défini par $e < I < \cdot, \cdot$ ayant un statut lexicographique de gauche à droite. On a alors*

$$\begin{array}{lll} x & <_{rpo} \; x \cdot e & \textit{par le cas 6.19.1.} \\ e & <_{rpo} \; x \cdot I(x) & \textit{par le cas 6.19.2.} \\ x \cdot (y \cdot z) & <_{rpo} \; (x \cdot y) \cdot z & \textit{par le cas 6.19.4.} \end{array}$$

Détaillons la dernière ligne :
- $(x, y \cdot z)(<_{rpo})_{lex}(x \cdot y, z)$ *car* $x <_{rpo} x \cdot y$ *par le cas 6.19.1.*
- $x <_{rpo} (x \cdot y) \cdot z$ *par le cas 6.19.1 itéré deux fois*
- $y \cdot z <_{rpo} (x \cdot y) \cdot z$ *par le cas 6.19.4. :*
  - $(y, z)(<_{rpo})_{lex}(x \cdot y, z)$ *car* $y <_{rpo} x \cdot y$ *par le cas 6.19.1.*
  - $y <_{rpo} (x \cdot y) \cdot z$ *par le cas 6.19.1 itéré deux fois.*
  - $z <_{rpo} (x \cdot y) \cdot z$ *par le cas 6.19.1.*

6.4 KBO : Knuth-Bendix Ordering

L'ordre KBO [28] 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 récriture, il faut en outre des conditions dites de compatibilité :
- 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 7

Terminaison

7.1 Notion de terminaison

7.1.1 Indécidabilité

Le problème de savoir si un système de récriture quelconque termine est intimement lié à l'arrêt des machines de Turing et, par là, 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 récriture [25] que Dershowitz a ensuite réduit à un nombre fixé (et petit) [15]. Enfin Dauchet a proposé un codage pour une machine de Turing arbitraire avec une seule règle, linéaire gauche et sans superposition sur elle-même [13]. On réduit ainsi à la terminaison des systèmes le problème de l'arrêt des machines de Turing.

Huet & Lankford Nous donnons ici la traduction simple inspirée des travaux de Huet & Lankford [25] en quelques règles de récriture de mots.

Considérons une machine de Turing $(Q, q_0, A = \{0, 1\}, b, \delta)$ où Q est l'ensemble des états, q_0 l'état initial, 0 et 1 les deux lettres de l'alphabet de travail A, b le caractère de blanc et δ la fonction de transition de $Q \times A$ vers $Q \times A \times \{\blacktriangleleft, \blacktriangleright\}$ qui à un état et à un caractère lu associe un nouvel état, un caractère écrit et un déplacement vers la droite $\blacktriangleright$ ou la gauche $\blacktriangleleft$ sur le ruban borné à gauche.

Afin d'exprimer les configurations successives d'une machine de Turing dans le formalisme de la récriture sur les mots, nous considérons les mots sur la signature $\mathcal{F} = Q \cup A \cup \{b\}$.

Dans un mot $q_i\alpha$, q_i représente l'état actuel de la machine et α le caractère sur lequel pointe la tête. Les transitions peuvent donc être codées de la manière suivante :

$$\left.\begin{array}{r} q_i\alpha \to \beta q_j \\ 0q_i\alpha \to q_j 0\beta \\ 1q_i\alpha \to q_j 1\beta \end{array}\right\} \quad \begin{array}{ll} \text{si} & \delta(q_i, \alpha) = (q_j, \beta, \blacktriangleright), \\ \text{si} & \delta(q_i, \alpha) = (q_j, \beta, \blacktriangleleft). \end{array}$$

Il reste à ajouter les cas dégénérés, c'est-à-dire faisant intervenir le caractère blanc, pour obtenir la simulation désirée.

On obtient ainsi une correspondance entre les configurations successives de la machine et les mots obtenus par réduction à l'aide du système.

Correspondance de Post Il existe également une illustration par réduction du problème de la *correspondance de Post* à la terminaison d'un système de récriture.

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 s'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 récriture défini sur la signature $\{f, \epsilon, a_1, \ldots, a_n\}$. Le symbole f est ternaire, le symbole ϵ est une constante et les symboles a_i sont unaires.

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

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

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

En outre pour chaque symbole a_i, on définit

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

Cet ensemble de n règles est appelé R_2.

Le système de récriture $R_1 \cup R_2$ 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
$$f(\epsilon, \epsilon, w) \underset{R_2}{\to} f(w, w, w) \overset{*}{\underset{R_1}{\to}} f(\epsilon, \epsilon, w)$$
– Si $R_1 \cup R_2$ ne termine pas, il y a forcément un nombre infini d'étapes de R_2, car R_1 seul termine (par décroissance sur les tailles). On a donc un schéma de la forme
$$f(\epsilon, \epsilon, w) \underset{R_2}{\to} f(w, w, w) \overset{*}{\underset{R_1}{\to}} w_1 \underset{R_2}{\to} w_2$$
Comme R_1 ne change pas le symbole de tête des termes, que w ne contient pas de f, que R_1 n'en introduit pas et ne change pas le dernier argument des termes qu'il récrit, w_1 est de la forme $f(\_, \_, w)$ et comme on peut lui appliquer $\to_{R_2}$, il est de la forme $f(\epsilon, \epsilon, w)$, ce qui signifie que w peut s'écrire de deux façons différentes comme
$$u_{j_1} u_{j_2} \ldots u_{j_q} = w = v_{j_1} v_{j_2} \ldots v_{j_q}$$

Le problème étant indécidable, les techniques destinées à prouver la terminaison de systèmes sont, bien sûr, correctes mais surtout forcément incomplètes. Elles aboutissent donc à un argument de terminaison, un argument de non-terminaison ou un constat d'impuissance.

7.1.2 Schéma de preuve

L'approche typique d'une preuve (automatique) de terminaison consiste à passer de problèmes de bonne fondation à d'autres, équivalents mais plus faciles à résoudre, jusqu'à obtenir des problèmes triviaux ou bien qui admettent une solution donnée par un ordre bien fondé.

On aboutit de fait à un arbre de preuve comme celui de la figure 7.1.

Nous appellerons *critères* les méthodes correctes et complètes permettant de passer d'un problème de bonne fondation à un (ensemble d') autre(s). Ils impliquent des contraintes dites *contraintes de terminaison*. La dernière étape de preuve fera éventuellement intervenir des constructions d'ordres bien fondés.

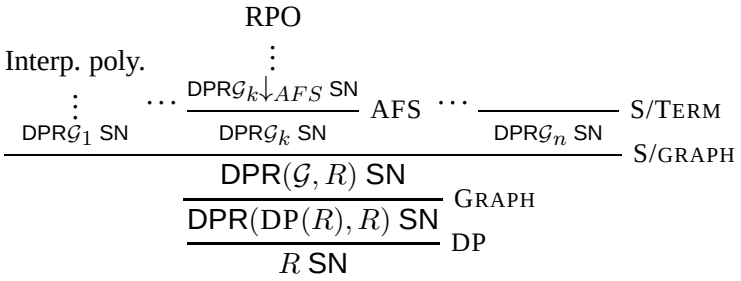


FIG. 7.1 – Un schéma typique de preuve de terminaison. La notation SN signifie « est bien fondée ».

Par exemple, sur le schéma de la figure 7.1, le problème de la terminaison de la relation initiale est transformé en un problème de terminaison de la relation DPR induite par $\mathrm{DP}(R)$ et R à l'aide du critère DP, puis par le critère GRAPH et ainsi de suite jusqu'à obtenir des contraintes d'ordres résolues par interprétations polynomiales (à gauche), d'autres résolues par RPO (au centre) et encore d'autres réduites au problème trivial (à droite). Nous allons passer en revue ces critères.

7.2 Critères

7.2.1 Manna & Ness

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

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

Théorème 7.1 *Soient $R = \{l_i \to r_i\}_{i=1,\dots,n}$ et $\succeq$ un préordre de récriture.*
Si $r_i \prec l_i$ pour $i = 1, \dots, n$, 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 récriture : $t = t[l_i\sigma]_p$ et $s = t[r_i\sigma]_p$. Par hypothèse, $r_i \prec l_i$. Comme $\preceq$ est un préordre de récriture,
- $\prec$ est stable, donc $r_i\sigma \prec l_i\sigma$,
- $\prec$ est monotone, donc $s = t[r_i\sigma]_p \prec t = t[l_i\sigma]_p$.

En conclusion, $\leftarrow_R$ est incluse dans $\prec$. Puisque $\prec$ est bien fondé, $\leftarrow_R$ l'est aussi.
□

La réciproque est évidente : la relation qui termine définit un ordre de récriture bien fondé.

7.2.2 Paires de dépendances (DP)

La méthode précédente est trop restrictive dans le sens où elle impose la monotonie stricte des ordres convenables. Arts & Giesl ont proposé en 1997 une analyse plus fine de la structure des termes non fortement normalisables qui permet de dégager de nouvelles contraintes d'ordres, plus nombreuses mais plus souples [3].

Intuitivement, s'il existe une réduction infinie alors il existe une réduction particulière où les étapes de récriture « cruciales » sont de plus en plus profondes. Il est donc suffisant de constater une décroissance stricte sur ces étapes et une décroissance large sur les autres. Les étapes cruciales sont les réductions de profondeur maximale nécessaire à la réduction infinie. En effet, un argument simple de minimalité suffit pour montrer que si un terme t donne lieu à une réduction infinie, alors il admet un sous-terme non accessible $f(u_1, \ldots, u_n)$ tel que tous les u_i sont accessibles.

On obtient ainsi l'existence d'une réduction particulière suivant les sous-termes minimaux non accessibles. Pour garantir la terminaison il reste à prouver qu'une telle dérivation ne peut se produire. Cette dérivation reposant essentiellement sur l'existence de sous-termes pouvant déclencher une réduction, c'est cette propriété que nous allons chercher à contenir dans de nouvelles contraintes de terminaison.

Nous présentons successivement les versions non marquée puis marquée des paires de dépendance.

Définition 7.2 (Paire de dépendance) *Soit R un système de récriture 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 $\{f \in \mathcal{F} \mid \exists\, l \to r \in R \;\; l(\Lambda) = f\}$,*
- *$\mathcal{C}$ l'ensemble des* constructeurs *est égal à $\mathcal{F} \setminus \mathcal{D}$.*

Une paire de dépendance *$\langle u, v\rangle$ 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 $v(\Lambda) \in \mathcal{D}$.*

L'ensemble des paires de dépendance d'un système R est noté $\mathrm{DP}(R)$.

Exemple 7.3 *Soit $\mathcal{F} = \{0, S, +\}$ et R le système de récriture 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 $\{0, S\}$, $\mathrm{DP}(R)$ est réduit à $\{\langle x + S(y), x + y\rangle\}$.

Définition 7.4 (Chaîne de dépendance) *Une* chaîne de dépendance *est une séquence de paires de dépendance $\langle u_i, v_i\rangle_i$ telle qu'il existe une substitution σ et*

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

Une chaîne de dépendance est dite minimale *si les sous-termes stricts des instances de paires sont tous fortement normalisables.*

Remarquons que les substitutions sont considérées ici comme étant à support *infini*.

Théorème 7.5 (Arts et Giesl) *$\leftarrow_R$ n'est pas bien fondée si et seulement s'il existe une chaîne de dépendance infinie.*

Démonstration S'il existe une chaîne de dépendance infinie $\langle u_i, v_i\rangle_i$, montrons que $\leftarrow_R$ n'est pas bien fondée : soit σ la substitution correspondante. À partir de la chaîne de dépendance, on reconstruit une séquence de récriture dans R en accumulant les contextes perdus :

- Au début le contexte $C_0[\_]$ est égal à $[\_]$,
- La paire $\langle u_i, v_i\rangle$ provenant de $l \to r$ avec $v_1 = r|_p$ donne lieu à l'étape de récriture $C_{i-1}[l\sigma] \to C_{i-1}[r\sigma]$, le nouveau contexte $C_i[\_]$ est égal à $C_{i-1}[r[\_]_p]$ et les étapes $v_i\sigma \to_R^* u_{i+1}\sigma$ deviennent $C_i[v_i\sigma] \to_R^* C_i[u_{i+1}\sigma]$.

On a donc construit une séquence de récriture infinie, $\leftarrow_R$ n'est pas bien fondée.

Réciproquement, supposons que $\leftarrow_R$ n'est pas bien fondée. Soit t un terme non accessible minimal pour l'ordre sous-terme (c.-à-d. 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 :

$$t \xrightarrow[R]{\neq\Lambda\ *} t' \xrightarrow[l\to r]{\Lambda} t''$$

$t'' = r\sigma$ 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, puisqu'aucune récriture 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, $\langle l, v\rangle$ est donc une paire de dépendance de R ; le début de la séquence infinie de pas de récriture est transformé en un début de chaîne de dépendance par

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

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.

□

Remarque 7.6 *Dershowitz utilise le fait qu'on montre l'absence de chaîne* minimale *pour se passer de certaines paires qui* ne peuvent pas *y figurer : typiquement, les paires $\langle l, r\rangle$ provenant de $l \to C[r]$ où r est un sous-terme de l. C'est ce qu'on fera dans la pratique (les preuves de ce chapitre restent inchangées).*

Le contrôle est effectué sur les réductions en tête des sous-termes minimaux non fortement normalisants. On peut donc distinguer les symboles en tête de ces sous-termes.

Définition 7.7 (Paires de dépendance marquées) *Soit R un système de récriture 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*

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

On note $\widehat{t}$ le terme t où le symbole de tête de t a été remplacé par sa version marquée. Une paire de dépendance marquée *$\langle u, v\rangle$ de R est formée à partir d'une règle $l \to r$ de R et telle que :*

- *$u = \widehat{l}$,*
- *$v = \widehat{w}$ où w est un sous-terme de r tel que $w(\Lambda) \in \mathcal{D}$.*

Une chaîne de dépendance marquée *est une séquence de paires de dépendance marquées $\langle u_i, v_i\rangle_i$ telle qu'il existe une substitution σ et*

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

Théorème 7.8 (Arts et Giesl) *$\leftarrow_R$ n'est pas bien fondée si et seulement s'il existe une chaîne de dépendance* marquée *infinie.*

Démonstration La preuve du théorème 7.5 s'adapte sans problème car les étapes de récriture $\rightarrow_R$ ont lieu uniquement dans des sous-termes stricts et ne sont pas gênées par les marques ; les étapes de paires de dépendance deviennent des étapes marquées.

□

Les chaînes minimales suivent un certain schéma et peuvent être généralisées en une relation paramétrée par un ensemble de « paires » et un système de règles.

Définition 7.9 *On définit la relation $\leftarrow_{\mathsf{DPR}(\mathscr{D},R)}$ par par $s \rightarrow_{\mathsf{DPR}(\mathscr{D},R)} t$ si :*

- *Tous les sous-termes stricts de s et t sont fortement normalisables,*
- *Il existe σ telle que $s \xrightarrow[R]{\neq\Lambda\ *} s' \equiv u\sigma \xrightarrow[\langle u,v\rangle\in\mathscr{D}]{} t$.*

La conclusion de ce paragraphe est donc que la bonne fondation de $\leftarrow_R$ est équivalente à la bonne fondation de la relation $\leftarrow_{\mathsf{DPR}(\mathrm{DP}(R),R)}$.

Nous obtenons le critère :

$$\frac{\mathsf{DPR}(\mathrm{DP}(R),R)\ \mathsf{SN}}{R\ \mathsf{SN}}\ \mathrm{DP}$$

7.2.3 Graphes de dépendance (GRAPH, S/GRAPH)

On peut remarquer que certaines paires de dépendance ne peuvent apparaître qu'un nombre fini de fois dans une réduction. Pour les détecter, on considère un graphe dont les nœuds sont les paires et tel que les paires qui peuvent apparaître consécutivement dans une chaîne de dépendance sont reliées. Une chaîne de dépendance infinie fait donc intervenir des paires d'une partie fortement connexe de ce graphe.

Dans la suite de cette section les systèmes sont tous finis.

Définition 7.10 *Soit R un système de récriture. On appelle* graphe de dépendance de R *le graphe $\mathcal{G}$ dont les nœuds sont les paires de dépendance de R et tel que $(\langle s,t\rangle, \langle s',t'\rangle) \in \mathcal{G}$ si et seulement s'il existe une substitution σ vérifiant $t\sigma \xrightarrow{\neq\Lambda\ *} s'\sigma$.*

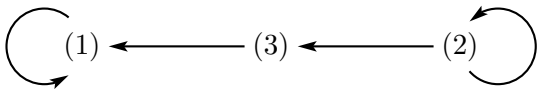
Exemple 7.11 (Arts & Giesl [2]) *Considérons un système de division des entiers de Peano :*

$$\begin{array}{llll} x-0 & \rightarrow & x & 0 \div s(y) \rightarrow 0 \\ s(x)-s(y) & \rightarrow & x-y & s(x) \div s(y) \rightarrow s((x-y)\div s(y)) \end{array}$$

On extrait trois paires de dépendance :

$$(1)\ \langle s(x)\widehat{-}s(y), x\widehat{-}y\rangle \qquad (2)\ \langle s(x)\widehat{\div}s(y), (x-y)\widehat{\div}s(y)\rangle \qquad (3)\ \langle s(x)\widehat{\div}s(y), x\widehat{-}y\rangle$$

On peut les agencer en :



Puisque (3) n'est pas sur une partie fortement connexe du graphe, elle n'est pas à considérer.

Théorème 7.12 (Critère par graphe) *Soit R un système de récriture sur $\mathcal{T}(\mathcal{F},\mathcal{X})$, soit $\mathcal{G}$ son graphe de dépendance. Soient $\mathcal{G}_1,\ldots,\mathcal{G}_k$ les k parties fortement connexes de $\mathcal{G}$ (et donc $\mathcal{G}_i \subseteq \mathrm{DP}(R)$ pour tout $i \leq k$). Toutes les $\leftarrow_{\mathsf{DPR}(\mathcal{G}_i,R)}$ sont bien fondées si et seulement si $\leftarrow_{\mathsf{DPR}(\mathrm{DP}(R),R)}$ est bien fondée.*

Démonstration Supposons que $\mathsf{DPR}(\mathrm{DP}(R),R)$ ne termine pas et considérons une réduction infinie de $\mathsf{DPR}(\mathrm{DP}(R),R)$. Puisque le graphe est fini, une chaîne infinie implique une infinité d'occurrences d'instances de mêmes paires qui correspondent à des paires joignables (par définition). Ces paires appartiennent donc à une partie fortement connexe, disons $\mathcal{G}_i$. Ainsi à partir d'un certain nombre de pas, cette réduction infinie de $\mathsf{DPR}(\mathrm{DP}(R),R)$ est incluse dans $\leftarrow_{\mathsf{DPR}(\mathcal{G}_i,R)}$ ce qui est impossible puisque cette dernière est bien fondée.

La réciproque est évidente puisque $\leftarrow_{\mathsf{DPR}(\mathcal{G}_i,R)} \subseteq \leftarrow_{\mathsf{DPR}(\mathcal{G},R)} \subseteq \leftarrow_{\mathsf{DPR}(\mathrm{DP}(R),R)}$.

□

Ce même argument permet en fait de montrer les deux critères :

$$\frac{\mathsf{DPR}(\mathcal{G}, R)\ \mathsf{SN}}{\mathsf{DPR}(\mathrm{DP}(R), R)\ \mathsf{SN}}\ \textsc{Graph} \qquad \frac{\forall \mathcal{G}_i \text{ fortement connexe} \subseteq \mathcal{G}, \mathsf{DPR}(\mathcal{G}_i, R)\ \mathsf{SN}}{\mathsf{DPR}(\mathcal{G}, R)\ \mathsf{SN}}\ \textsc{S/Graph}$$

Exemple 7.13 *Soit R le système réduit à une seule règle :*

$$f(f(x)) \to f(s(f(x)))$$

Ses paires de dépendances sont $\langle \widehat{f}(f(x)), \widehat{f}(x)\rangle$ et $\langle \widehat{f}(f(x)), \widehat{f}(s(f(x)))\rangle$. La première est écartée d'emblée grâce à la remarque 7.6. Comme il n'existe aucune substitution telle que $\widehat{f}(f(x))\sigma \xrightarrow{\neq\Lambda\ *} \widehat{f}(s(f(x)))\sigma$ la deuxième paire n'est pas sur un cycle du graphe. On en déduit que $\mathsf{DPR}(\mathrm{DP}(R), R)$ et donc R terminent par les critères GRAPH *puis* DP.

Approximation simple du graphe Le graphe de dépendance n'est pas calculable en toute généralité : l'existence de σ telle que $s\sigma \to t\sigma$ n'est pas décidable. On doit donc l'approcher par un graphe qui le contient.

Nous décrivons ici la première méthode introduite par Arts & Giesl. Elle est peu coûteuse et souvent suffisamment puissante.

On veut déterminer un ensemble d'arcs qui contient le graphe de dépendance. Les réductions éventuelles des sous-termes stricts vont forcément laisser intacts les symboles constructeurs situés entre leur position et la racine des instances de paires. En revanche, les sous-termes dont la racine est un symbole défini peuvent être récrits (de façon quelconque). On va donc sélectionner les paires suivant la possibilité d'unifier des membres dont les sous-termes ayant des symboles définis à la racine ont été remplacés par des variables distinctes *quelles que soient les occurrences*.

Définition 7.14 *Soit R un système de récriture sur $\mathcal{T}(\mathcal{F}, \mathcal{X})$. Pour tout terme t,*

- $\mathrm{CAP}(t)$ *désigne le terme obtenu en remplaçant par des variables les sous-termes de t dont le symbole à la racine est défini dans R ;*
- $\mathrm{REN}(t)$ *est le terme obtenu en remplaçant les variables de t par de nouvelles variables distinctes (pour toutes occurrences).*

Le terme t est connectable *à un terme t' si $\mathrm{REN}(\mathrm{CAP}(t))$ et t' sont unifiables.*

Par unification syntaxique sur les images des termes par REN et CAP on détermine des arcs entre les paires de dépendance du système. Le graphe obtenu contient bien le graphe de dépendance.

Proposition 7.15 *Soit R un système de récriture. S'il existe une substitution σ telle que*

$$t\sigma \xrightarrow[R]{\neq\Lambda\ *} t'\sigma,$$

alors t est connectable à t'.

Démonstration Par induction sur la structure de t, supposons que $t\sigma$ se récrive en un terme t'' par le système R.

- Si t est une variable ou si son symbole de tête est défini alors $\mathrm{REN}(\mathrm{CAP}(t))$ est une variable et donc unifiable à t''.

– Si t a pour symbole de tête un constructeur c, on peut alors poser $t = c(\ldots, t_i, \ldots)$. Ainsi $\text{REN}(\text{CAP}(t))$ s'écrit $c(\ldots, \text{REN}(\text{CAP}(t_i)), \ldots)$. Comme un terme dont la racine est un constructeur ne peut se réduire qu'en un (autre) terme dont la racine est le même constructeur, t'' s'écrit $c(\ldots, {t''}_i, \ldots)$ avec les $t_i\sigma$ se récrivant en ${t''}_i$. Par hypothèse et par distinction des variables après application de REN, $\text{REN}(\text{CAP}(t))$ est unifiable à t''.

On a le résultat pour $t\sigma \to^* t''$, donc en particulier pour $t\sigma \to^* t'\sigma$, il vient ainsi que $\text{REN}(\text{CAP}(t))$ est unifiable à $t'\sigma$ et finalement, comme il ne contient que de nouvelles variables, à t'.

□

Remarque 7.16 *Il s'agit bien d'unification et pas de filtrage : prenons $s = f(x, s(y))$ et $t = f(s(u), v)$ où f est le seul symbole défini, s et t sont unifiables mais s ne filtre pas t. Pourtant avec $R : \{f(x, y) \to s(x)\}$ et $\sigma = \{x \mapsto f(u, u),\ v \mapsto s(y)\}$ on peut avoir la réduction*

$$s\sigma = f(f(u, u), s(y)) \xrightarrow[R]{\neq\Lambda\ *} f(s(u), s(y)) = t\sigma.$$

Enfin, puisque plusieurs règles peuvent s'appliquer à un même terme, il est important de renommer chacune des occurrences des variables pour ne pas obtenir de conclusions erronées. En effet : un terme peut, dans le cas général, se réduire en deux termes différents.

7.2.4 Subterm-criterion (S/TERM)

On peut encore affiner l'analyse du graphe en généralisant la remarque 7.6 et en ne considérant que les parties fortement connexes convenant à une chaîne *minimale*. Nous présentons l'analyse de Hirokawa et Middeldorp [23].

Définition 7.17 *Une* projection simple *P est une application de $\widehat{\mathcal{F}} \to \mathbb{N}$ qui associe à chaque $\widehat{f}$ d'arité n un indice $i_{\widehat{f}} \leq n$. Par abus $P(\widehat{f}(t_1, \ldots, t_n))$ désignera le sous-terme $t_{P(\widehat{f})}$.*

Théorème 7.18 (Hirokawa & Middeldorp [23]) *Soit $\mathscr{D}$ l'ensemble des paires de dépendance d'un système R formant une partie fortement connexe du graphe de dépendance de R. S'il existe une projection simple P telle que :*

1. *$\forall \langle u, v\rangle \in \mathscr{D}$, il existe une position p telle que $P(u)|_p = P(v)$ (noté $P(u) \trianglerighteq P(v)$),*
2. *$p \neq \Lambda$ pour au moins une $\langle u, v\rangle \in \mathscr{D}$ (noté $P(u) \triangleright P(v)$),*

alors une chaîne de dépendance de $\mathscr{D}$ sur R n'est pas minimale.

Il s'agit bien d'une généralisation de la remarque de Dershowitz : on construit grâce à la projection simple une réduction plus petite (profonde) au sens du sous-terme. En conclusion il n'y a pas besoin de contrôler la décroissance sur cette partie fortement connexe dans la preuve de terminaison.

Démonstration Par contradiction : supposons l'existence d'une chaîne *minimale* infinie

$$t_1 \xrightarrow[R]{\neq\Lambda\ *} s_2 \xrightarrow[\mathscr{D}]{\Lambda} t_2 \xrightarrow[R]{\neq\Lambda\ *} s_3 \xrightarrow[\mathscr{D}]{\Lambda} t_3 \cdots$$

et d'une projection simple P vérifiant les hypothèses.

1. Pour chaque étape $s_i = u\sigma \xrightarrow[\langle u,v\rangle\in\mathscr{D}]{\Lambda} v\sigma = t_i$, on a par hypothèse que $P(s_i) = P(t_i)$ ou bien $P(s_i) \triangleright P(t_i)$, ce dernier cas arrivant une infinité de fois.
2. Pour chaque étape $t_i \xrightarrow[R]{\neq\Lambda\ *} s_{i+1}$, comme $t_i(\Lambda) = s_{i+1}(\Lambda)$ on a que $P(t_i) \xrightarrow[R]{*} P(s_{i+1})$, réduction qui diffère éventuellement de la réduction originale car on perd les étapes ayant pu se produire dans les sous-termes écartés par P.

On obtient donc une réduction infinie

$$P(t_1) \xrightarrow[R]{*} P(s_2) \trianglerighteq P(t_2) \xrightarrow[R]{*} P(s_3) \trianglerighteq P(t_3) \cdots$$

dont une infinité d'étapes $\trianglerighteq$ sont en fait des étapes $\triangleright$ et comme la relation sous-terme strict est bien fondée, on a une infinité d'étapes de R. Par applications de $\triangleright \cdot \to_R \subseteq \to_R \cdot \triangleright$ on construit une réduction infinie par R partant de $P(t_1)$ ce qui contredit la minimalité de t_1.
□

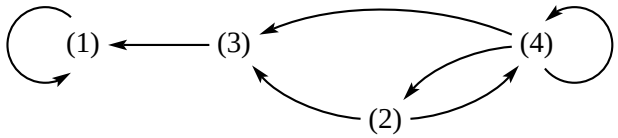
Exemple 7.19 *Hirokawa et Middeldorp [23] sur un système de Steinbach et Külher :*

$\text{intlist}([\,]) \to [\,]$	$\text{int}(0,0) \to 0 : [\,]$	$\text{int}(s(x),0) \to [\,]$
$\text{intlist}(x : y) \to s(x) : \text{intlist}(y)$	$\text{int}(0,s(y)) \to 0 : \text{int}(s(0),s(y))$	$\text{int}(s(x),s(y)) \to \text{intlist}(\text{int}(x,y))$

Les quatre paires de dépendances sont :

(1)	$\langle \widehat{\text{intlist}}(x : y), \widehat{\text{intlist}}(y) \rangle$	(2)	$\langle \widehat{\text{int}}(0,s(y)), \widehat{\text{int}}(s(0),s(y)) \rangle$
(3)	$\langle \widehat{\text{int}}(s(x),s(y)), \widehat{\text{intlist}}(\text{int}(x,y)) \rangle$	(4)	$\langle \widehat{\text{int}}(s(x),s(y)), \widehat{\text{int}}(x,y) \rangle$

Elles forment le graphe :



- *Le cycle réduit à la paire* (1) *est prouvé non minimal par le critère de sous-terme en utilisant la projection simple de* $\widehat{\text{intlist}}$ *sur son argument.*
- *Les cycles contenus dans la composante fortement connexe formée par* (2) *et* (4) *sont prouvés non minimaux par le critère de sous-terme avec la projection simple de* $\widehat{\text{int}}$ *sur son deuxième argument.*

7.2.5 Filtrage d'arguments (AFS)

Il existe des méthodes de transformation d'une relation en une autre dont la bonne fondation implique celle de la relation originale. L'une d'elles met en œuvre des AFS ou *Argument Filtering Systems*, des systèmes de récriture de forme particulière par lesquels on va transformer les contraintes de terminaison.

Pour un système R sur $\mathcal{T}(\mathcal{F},\mathcal{X})$ un AFS A est un système sur $\mathcal{T}(\mathcal{F}',X)$ avec $\mathcal{F} \subseteq \mathcal{F}'$ tel que pour tout $l \to r \in A$:

- l est de la forme $f(x_1,\ldots,x_n)$ où $f \in \mathcal{F}$ d'arité n et où les x_i sont des variables deux à deux distinctes,
- l apparaît au plus une fois comme membre gauche,
- r est soit :
  - $x_i \in \{x_1,\ldots,x_n\}$, soit
  - $f'(x_{i_1},\ldots,x_{i_k})$ où $f' \in (\mathcal{F}' \setminus \mathcal{F})$ et $[x_{i_1},\ldots,x_{i_k}]$ est une sous-liste (év. vide) de $[x_1,\ldots,x_n]$.

Les AFS sont confluents et terminent par définition.

Intuitivement, un AFS est un système convergent qui va permettre de retirer certains arguments de fonction et donc de profiter de la perte de monotonie. On note $t{\downarrow_A}$ la forme normale de t par l'AFS A (qu'on omettra en l'absence d'ambiguïté) et, de manière générale, on désigne par $R{\downarrow}$ l'ensemble de règles constitué des $l{\downarrow} \to r{\downarrow}$ pour chaque $l \to r \in R$. On peut remarquer que, puisque l'AFS est convergent, si $s \xrightarrow[R]{} t$ alors $s{\downarrow} \xrightarrow[R{\downarrow}]{} t{\downarrow}$ (ou $s{\downarrow} = t{\downarrow}$).

Théorème 7.20 *Soit R un système sur $\mathcal{T}(\mathcal{F},\mathcal{X})$ et soit $\mathscr{D} \subseteq \mathrm{DP}(R)$. Soit A un AFS, $\leftarrow_{\mathsf{DPR}(\mathscr{D},R)}$ est bien fondée si $\leftarrow_{\mathsf{DPR}(\mathscr{D}\downarrow_A,R\downarrow_A)}$ est bien fondée.*

Exemple 7.21 *Reprenons l'exemple 7.11. L'AFS $\{x - y \to m(x)\}$ permet d'obtenir un ensemble $\mathscr{D} = \{\langle s(x)\mathbin{\widehat{-}}s(y), x\mathbin{\widehat{-}}y\rangle, \langle s(x)\mathbin{\widehat{\div}}s(y), x\mathbin{\widehat{-}}y\rangle, \langle s(x)\mathbin{\widehat{\div}}s(y), m(x)\mathbin{\widehat{\div}}y\rangle\}$ et des règles R'*

$$m(x) \to x \qquad 0 \div s(y) \to 0 \qquad m(s(x)) \to m(x) \qquad s(x) \div s(y) \to s(m(x) \div s(y))$$

Pour lesquels $\mathsf{DPR}(\mathscr{D},R')$ termine et entraîne que R termine (voir l'exemple 7.23).

7.2.6 Prouver que DPR($\mathscr{D}$,R) termine à l'aide d'ordres

Puisque la relation $\to_{\mathsf{DPR}(\mathscr{D},R)}$ ne conserve pas les contextes, il n'est pas nécessaire que l'ordre pour lequel elle décroît strictement soit stable par contexte. On peut donc utiliser des paires d'ordres (cf. déf. 6.4).

Proposition 7.22 *Soit R un système de récriture défini sur $\mathcal{T}(\mathcal{F},\mathcal{X})$et soit $\mathscr{D} \subseteq \mathrm{DP}(R)$. S'il existe une paire d'ordres $(\succeq, >)$ bien fondée, faiblement monotone et telle que*

1. *$l \succeq r$ pour toute règle $l \to r \in R$,*
2. *$u > v$ pour toute paire $\langle u, v\rangle \in \mathscr{D}$*

alors $\leftarrow_{\mathsf{DPR}(\mathscr{D},R)}$ est bien fondée.

Exemple 7.23 *La relation $\mathsf{DPR}(\mathscr{D},R')$ termine par* RPO *avec la précédence $\mathbin{\widehat{\div}} \succ \mathbin{\widehat{-}}$ et $\div \succ s \succ m$.*

Corollaire 7.24 *En particulier, en reprenant les hypothèses de la proposition 7.22 :*
R est bien fondée si $\mathscr{D} = \mathrm{DP}(R)$.

7.3 Modularité de la terminaison

La complexité des systèmes rencontrés dans la pratique incite à l'utilisation d'une approche « Divide and Conquer » de la preuve de terminaison, c'est-à-dire de prouver la terminaison d'un ensemble de règle à partir de la preuve de terminaison de (certaines de) ses parties. Il n'est cependant pas possible de le faire en toute généralité : la terminaison n'est pas une propriété *modulaire* des systèmes de récriture, même lorsque les symboles ne sont pas partagés.

Contre-exemple 7.25 *Considérons les deux systèmes de récriture qui portent sur des symboles de fonction disjoints :*

$$R_1 = \{f(a,b,x) \to f(x,x,x)\}$$

$$\Pi = \{\pi(x,y) \to x, \pi(x,y) \to y\}$$

Le système Π termine trivialement. On peut montrer que R_1 termine : le graphe de dépendance ne contient pas de partie fortement connexe.

Toutefois, si on combine R_1 et Π, l'ensemble ne termine plus puisqu'on a la réduction infinie :

$$\cdots \to f(a,b,\pi(a,b)) \underset{R_1}{\longrightarrow} f(\pi(a,b),\pi(a,b),\pi(a,b)) \underset{\Pi}{\to} f(a,\pi(a,b),\pi(a,b)) \underset{\Pi}{\to} f(a,b,\pi(a,b)) \to \cdots$$

On considère donc en général des notions plus fortes de terminaison (terminaison *simple*, terminaison CE) ou des restrictions sur les relations (par exemple commutation, stratégie de réduction, etc.) qui se comportent mieux vis-à-vis de la modularité.

La plupart des outils utilisent la notion de terminaison CE (pour « Collapse Exended »), introduite par Gramlich [20] et étudiée par Ohlebusch [37] et Urbain [42]. Elle présente l'avantage d'être modulaire pour les unions sans symboles partagés. C'est également le cas pour les unions où les seuls symboles partagés sont des *constructeurs* (au sens de la déf. 7.2) *pour des systèmes à branchement fini*[1].

Définition 7.26 *Soient R un système sur $\mathcal{T}(\mathcal{F},\mathcal{X})$ et $\Pi = \{\pi(x,y) \to x, \pi(x,y) \to y\}$ où $\pi \notin \mathcal{F}$. On dit que R* termine CE *si $R \cup \Pi$ termine.*

Remarque 7.27 *Le graphe de dépendance approché par la méthode présentée section 7.2.3 est suffisamment grossier pour permettre de prouver la terminaison* CE.

C'est en fait la structure hiérarchique des systèmes (et donc des programmes) qui nous intéresse.

Définition 7.28 *On appelle* union hiérarchique *de deux systèmes R_1 sur $\mathcal{T}(\mathcal{F}_1,X)$ et R_2 sur $\mathcal{T}(\mathcal{F}_2,X)$ une union $R_1 \cup R_2$ qui, en notant*
- *D_1 et D_2 les ensembles des symboles définis de R_1 et R_2,*
- *C_1 et C_2 leurs constructeurs respectifs,*

vérifie les deux propriétés :
- *$(D_2 \setminus D_1) \cap \mathcal{F}_1 = \emptyset$ et*
- *$\{l \to r \mid l(\Lambda) \in D_1 \cap D_2\} = R_1 \cap R_2$.*

La structure hiérarchique peut être définie à l'aide de *modules*.

Définition 7.29 (Module de récriture)
Soit R_1 un système de récriture défini sur $\mathcal{T}(\mathcal{F},\mathcal{X})$. On note $\mathcal{F}_1$ l'ensemble des symboles de $\mathcal{F}$ qui apparaissent dans les règles de R_1. Soit R_2 un second système de récriture 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 apparaissent 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 portant sur l'arithmétique [42].

Exemple 7.30 *Le système de récriture 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 $R = R_0 \cup R_+ \cup R_*$:*

$$
\begin{array}{rcl}
R_0 & = & \{\#0 \to \# \\
R_+ & = & \left\{ \begin{array}{llll} \#+x & \to & x & x0+y1 \to (x+y)1 \\ x+\# & \to & x & x1+y0 \to (x+y)1 \\ x0+y0 & \to & (x+y)0 & x1+y1 \to ((x+y)+\#1)0 \end{array} \right. \\
R_\times & = & \left\{ \begin{array}{llll} \#\times x & \to & \# & x0\times y \to (x\times y)0 \\ x\times \# & \to & \# & x1\times y \to (x\times y)0+y \end{array} \right.
\end{array}
$$

[1] C'est-à-dire lorsque tout terme n'admet qu'un nombre fini de réduits en un pas.

Si on partitionne l'ensemble des symboles de fonctions en

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

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 récriture, on est amené à étendre la notion de paires de dépendance :

Définition 7.31 (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 $\langle \widehat{l}, \widehat{v} \rangle$ telle que*

- *$\langle \widehat{l}, \widehat{v} \rangle$ 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 7.32 *Dans l'arithmétique binaire, selon l'approche classique, il y a 5 paires de dépendance pour la multiplication :*

$$\begin{array}{lll} \langle x0 \mathbin{\widehat{\times}} y, x \mathbin{\widehat{\times}} y \rangle & \langle x1 \mathbin{\widehat{\times}} y, x \mathbin{\widehat{\times}} y \rangle & \langle x1 \mathbin{\widehat{\times}} y, (x\ \times\ y)0 \mathbin{\widehat{+}} y \rangle \\ \langle x0 \mathbin{\widehat{\times}} y, (x\ \times\ y)\widehat{0} \rangle & \langle x1 \mathbin{\widehat{\times}} y, (x\ \times\ y)\widehat{0} \rangle & \end{array}$$

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

$$\begin{array}{l} \langle x0 \mathbin{\widehat{\times}} y, x \mathbin{\widehat{\times}} y \rangle \\ \langle x1 \mathbin{\widehat{\times}} y, x \mathbin{\widehat{\times}} y \rangle \end{array}$$

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

Définition 7.33 (Chaîne de dépendance relative) *Soit $(\mathcal{F}_2, R_2)$ un module étendant $(\mathcal{F}_1, R_1)$ et S un système de récriture quelconque. Un chaîne de dépendance de $(\mathcal{F}_2, R_2)$ relative à S est une séquence $\langle u_i, v_i \rangle_i$ 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 terminaison CE pour l'union de systèmes et pour l'extension de modules :

Théorème 7.34 (Urbain [42]) *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* CE*,*
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 CE*.*

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

Lemme 7.35 *Soit $(\mathcal{F}_2, R_2)$ un module étendant $(\mathcal{F}_1, R_1)$. Soient 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 récriture 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 $\mathcal{T}(\widehat{\mathcal{F}_1} \cup \{\pi, \bot\}, \mathcal{X})$, on définit un interprétation I des termes par :

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

$$\begin{array}{l} List\_of(\emptyset) = \bot \\ List\_of(\{a\} \cup E) = \pi(a, List\_of(E)) \quad \text{si } \forall e \in E,\ a < e \\ Red(t) = \{I(t') \mid t \to_{R_1 \cup R_2} t'\} \end{array}$$

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 récriture modulo $R_1 \cup R_2$, alors qu'il n'y a aucune garantie que $R_1 \cup R_2$ termine.

Pour une substitution σ on notera $I(\sigma)$ la substitution qui à tout x associe $I(x\sigma)$.

Quelques remarques préliminaires :

Remarque 7.36 *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 multi-ensemble des sous-termes maximaux à symbole de tête dans $\mathcal{F}_2$ de t et de la taille de t, avec l'ordre $((\leftarrow^+_{R_1 \cup R_2})_{mul}, \lhd)_{lex}$. Le fait que $R_1 \cup R_2$ est à branchement fini est crucial ici, car sinon le multi-ensemble des réduits d'un terme n'est pas nécessairement fini.*

Remarque 7.37 *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 $s(\Lambda) \in \widehat{\mathcal{F}_1}$ 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 $l \to r$ 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 $\widehat{\mathcal{F}_1}$ et des variables on a $I(l\sigma) = lI(\sigma)$ et $I(r\sigma) = rI(\sigma)$. Donc*
$$I(s) \equiv lI(\sigma) \xrightarrow[l \to r]{\Lambda} I(t) \equiv rI(\sigma)$$
2. *Si $p = i \cdot q$, il faut distinguer deux cas :*
   (a) *Si $s(\Lambda) \in \widehat{\mathcal{F}_1}$, $s = f(s_1, \ldots, s_i, \ldots, s_n)$, $I(s) = f(I(s_1), \ldots, I(s_i), \ldots, I(s_n))$, 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 $I(s) = List\_of(Red(s))$:*
$$I(s) \xrightarrow[\Pi]{+} I(t)$$

[2] c'est-à-dire construite comme dans la preuve du théorème 7.5.

Remarque 7.38 *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 $s(\Lambda) \in \widehat{\mathcal{F}_1}$ 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 $s(\Lambda) \in \widehat{\mathcal{F}_1}$, $s = f(s_1, \ldots, s_n)$, $I(s) = f(I(s_1, \ldots, I(s_n))$. La règle de R_2 ne peut pas s'appliquer en tête de s, elle s'applique 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 7.35 elle-même. Soit donc une chaîne de dépendance infinie $\langle u_i, v_i\rangle_i$ 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 $\langle u_i, v_i\rangle_i$ 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 $\widehat{\mathcal{F}_1}$. Donc par la remarque 7.36, tous les termes intermédiaires s sont tels que $I(s)$ est bien définie car leur symbole de tête est dans $\widehat{\mathcal{F}_1}$ 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 7.37 et 7.38, ce qui conclut la preuve.

□

Voici maintenant la preuve du théorème principal :

Démonstration[Théorème 7.34] Par contradiction. Supposons que $R_1 \cup R_2 \cup R_3$ ne termine pas CE, 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 $\langle u_i, v_i\rangle_i$ 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 $\langle \widehat{u}, \widehat{v}\rangle$ 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 récritures intermédiaires 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

1. Uniquement de paires de type $\{1,2\} \times \{1,2\}$
2. Uniquement de paires de type $(3,3)$
3. D'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 7.35 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 terminaison CE 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 7.35 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.

□

7.4 Terminaison avec théorie AC

L'introduction d'une théorie équationnelle comme AC complique assez nettement les preuves de terminaison. La relation d'ordre bien fondée qui sert à la preuve doit être en particulier *compatible* avec la théorie équationnelle et cela impose des contraintes très fortes sur les ordres convenables.

Dans la suite de cette section et sauf mention contraire on considérera uniquement la récriture AC sur les termes aplatis. En particulier, c'est cette relation (pour un système R) à laquelle se rapportera la notation $\rightarrow_R$.

Ne considérer que la récriture étendue AC sur termes aplatis n'est pas une restriction du point de vue de la terminaison.

Théorème 7.39 *Un système termine modulo AC si et seulement s'il termine pour la récriture étendue AC sur termes aplatis.*

Démonstration On va construire pour toute réduction infinie modulo AC une réduction infinie pour la récriture étendue. On suppose l'existence d'une réduction infinie pour la récriture modulo AC issue d'un terme s et débutant par : $s \xrightarrow[l \to r/\mathrm{AC}]{p} \ldots$. On a donc une substitution σ telle que $s =_{\mathrm{AC}} s'$ où $s'|_p = l\sigma$ et le terme aplati $\overline{s}$ contient une instance de l ou de son extension. On peut donc récrire $\overline{s}$ en un terme plat égal modulo AC à t. En itérant cette construction, on construit bien une réduction infinie pour la récriture étendue AC sur termes aplatis. La réciproque est évidente.

□

7.4.1 Compatibilité AC

La contrainte ajoutée aux ordres pour tenir compte de la théorie équationnelle est d'évaluer de la même façon deux termes équivalents pour cette théorie. On restreint les ordres utilisables à l'ensemble des ordres

$(>, \geq)$ *compatibles* AC :

$$\text{Si} \quad \left.\begin{array}{lll} s & > & t \\ s & =_{\text{AC}} & s' \\ t & =_{\text{AC}} & t' \end{array}\right\} \quad \text{alors} \quad s' > t' \quad \text{et si} \quad \left.\begin{array}{lll} s & \geq & t \\ s & =_{\text{AC}} & s' \\ t & =_{\text{AC}} & t' \end{array}\right\} \quad \text{alors} \quad s' \geq t'.$$

Interprétation polynomiales compatibles

On appelle polynômes AC les polynômes convenables dans les interprétations de symboles AC.

Définition 7.40 *Le polynôme P est un* polynôme AC *si :*

1. $P(P(X,Y),Z) = P(X,P(Y,Z))$,
2. $P(X,Y) = P(Y,Z)$.

Une interprétation polynomiale associant à tout symbole AC un polynôme compatible AC définit un ordre compatible AC.

Il est aisé de vérifier qu'un polynôme (sur un anneau commutatif) est compatible.

Lemme 7.41 (Ben Cherifa & Lescanne [6]) *Le polynôme $P(X,Y)$ est AC si et seulement si c'est un polynôme symétrique du premier degré en chacune des indéterminées $aXY + b(X+Y) + c$ tel que $b^2 = b + ac$.*

Ordres sur les chemins compatibles AC

Le RPO n'est pas compatible AC.

Exemple 7.42 (Delor & Puel [14]) *Sur la signature à deux symboles f et g, où f est AC. Avec la précédence $f > g$ nous obtenons l'inégalité $f(x,y) >_{\text{RPO}} g(x,y)$. Plongeons ces deux termes dans le contexte $f(\Box, z)$: $f(x,y,z)$ et $f(g(x,y),z)$ sont orientés dans le « mauvais » sens.*

Différents adaptations des ordres sur les chemins ont été définies, notamment en posant des contraintes sur la précédence ainsi qu'à l'aide de systèmes qui redistribuent les symboles avant comparaison. On citera l'*Associative Path Ordering* (APO) de Bachmair et Plaisted [5] modifié ensuite par Bachmair et Dershowitz [4] ainsi que l'*Extended Associative Path Ordering* et le *Modified Associative Path Ordering* de Delor et Puel [14].

Un RPO compatible AC sans système de normalisation Pour se passer des systèmes de normalisations, Rubio définit [39] un ordre qui utilise directement le principe du RPO sans interprétation préliminaire.

Nous donnons ici la version de cet ordre pouvant utiliser des précédences partielles sur les symboles de la signature. Cette possibilité est importante dès qu'on veut ajouter des symboles à la signature.

Pour cela, il faut redéfinir la notion d'extension multi-ensemble d'un ordre compatible AC en particulier en la rendant dépendante de certains symboles.

Définition 7.43 *Soient $>$ un ordre compatible AC sur $\mathcal{T}(\mathcal{F},\mathcal{X})$ et $\succ$ une précédence partielle (sur $\mathcal{F}$). Soit f un symbole de $\mathcal{F}_{AC}$.*

*L'*extension multi-ensemble *de* $>$ par rapport à f*, notée* $>_{mul}^{f}$*, est définie comme la plus petite relation transitive contenant :*

$$M \cup \{s\} >_{mul}^{f} N \cup \{t_1 ; \ldots ; t_n\} \text{ si } \begin{cases} M \text{ et } N \text{ sont équivalents modulo AC,} \\ \text{Pour tout } i,\ 1 \leq i \leq n, \\ \quad - \ s > t_i \text{ et} \\ \quad - \ \text{Si } s(\Lambda) \not\succ f \text{ alors } s(\Lambda) \succeq t_i(\Lambda). \end{cases}$$

On doit savoir traiter les ensembles de sous-termes de s dont les symboles de tête sont éventuellement comparables pour la précédence à $s(\Lambda)$.

Définition 7.44 *Soient* $s = f(s_1, \ldots, s_n)$ *un terme avec* $f \in \mathcal{F}_{AC}$ *et* $\succ$ *une précédence sur* $\mathcal{F}$*. On définit les ensembles :*

$$\mathit{BigHead}(s) = \{s_i,\ 1 \leq i \leq n \mid s_i(\Lambda) \succ f\} ;$$

$$\mathit{NoSmall}(s) = \{s_i,\ 1 \leq i \leq n \mid f \not\succ s_i(\Lambda)\}.$$

Afin de permettre le plongement dans un terme s de termes à travers des symboles qui ne sont pas plus grands pour la précédence $\succ$ que $s(\Lambda)$ on définit l'ensemble $\mathit{EmbNoBig}(s)$.

Définition 7.45 *Soit* s *un terme de la forme* $f(s_1, \ldots, s_n)$ *avec* $f \in \mathcal{F}_{AC}$*,* $\succ$ *une précédence sur* $\mathcal{F}$*. On désigne par* $\overline{t}^{f}$ *le terme* t *après aplatissement du symbole* f *à la racine uniquement.*

$$\mathit{EmbNoBig}(s) = \\ \{f(s_1, \ldots, s_{i-1}, \overline{v_j}^{f}, s_{i+1}, \ldots, s_n) \mid s_i = h(v_1, \ldots, v_k) \text{ avec } h \not\succ f \text{ et pour } 1 \leq j \leq k\}.$$

Enfin pour comparer des termes contenant des variables on introduit une interprétation $\#$ des termes vers des expressions diophantiennes par $\#(f(t_1, \ldots, t_n)) = \#'(t_1) + \ldots + \#'(t_n)$ où $\#'(x) = x$ pour $x \in X$ et $\#'(t) = 1$ sinon.

On peut alors définir l'ordre ACRPO.

Définition 7.46 *Soient* s *et* t *deux termes de* $\mathcal{T}(\mathcal{F}, \mathcal{X})$*,* $\succ$ *une précédence sur* $\mathcal{F}$ *et* $>_e$ *l'ordre d'évaluation sur les fonctions sur* $\mathbb{N}$*. L'ordre* ACRPO *est défini par* $s = f(s_1, \ldots, s_n) >_{\text{ACRPO}} g(t_1, \ldots, t_m) = t$ *si et seulement si l'une des conditions suivantes est vérifiée :*

1. $s_i \geq_{\text{ACRPO}} t$ *pour un* i*,* $1 \leq i \leq n$ *;*
2. $f \succ g$ *et* $s >_{\text{ACRPO}} t_j$ *pour tout* j*,* $1 \leq j \leq m$ *;*
3. $f = g \notin \mathcal{F}_{AC}$*,* $[s_1, \ldots, s_n] >_{\text{ACRPO}}^{lex} [t_1, \ldots, t_n]$ *et* $s >_{\text{ACRPO}} t_j$ *pour tout* j*,* $1 \leq j \leq m$ *;*
4. $f = g \in \mathcal{F}_{AC}$ *et il existe un* $s' \in \mathit{EmbNoBig}(s)$ *tel que* $s' \geq_{\text{ACRPO}} t$ *;*
5. $f = g \in \mathcal{F}_{AC}$*,* $s >_{\text{ACRPO}} t'$ *pour tout* $t' \in \mathit{EmbNoBig}(t)$*,* $\mathit{NoSmall}(s)(\geq_{\text{ACRPO}})_{mul}^{f}\mathit{NoSmall}(t)$ *et ou bien :*
   (a) $\mathit{BigHead}(s)(>_{\text{ACRPO}})_{mul}\mathit{BigHead}(t)$ *ou*
   (b) $\#(s) >_e \#(t)$ *ou*
   (c) $\#(s) \geq_e \#(t)$ *et* $\{s_1 ; \ldots ; s_n\}(>_{\text{ACRPO}})_{mul}\{t_1 ; \ldots ; t_m\}$.

Théorème 7.47 (Rubio [39]) ACRPO *est un ordre de simplification compatible avec AC.*

7.4.2 Paires de dépendance AC

L'une des difficultés de la définition de paires de dépendance pour la récriture AC réside dans le traitement des marques. Celles-ci interfèrent en effet avec les pas AC : on ne peut pas en toute généralité permuter un symbole avec sa version marquée. Marché et Urbain [35] introduisent une notion abstraite de paires de dépendance autorisant plusieurs façon de gérer les marques.

Définition 7.48 *Une* définition abstraite de paires de dépendance AC *est une fonction qui, à un système de récriture R sur $\mathcal{T}(\mathcal{F},\mathcal{X})$ où $\mathcal{F} = C \cup D$ (C constructeurs, D symboles définis) associe un couple constitué :*

1. *D'un ensemble E de paires de termes $\langle u, v\rangle$ sur la signature $\widehat{D} \cup D \cup C$,*
2. *D'une relation $\rightarrow_{mh}$ telle que :*
   (a) *Pour tout $\langle u, v\rangle \in E$ on a $u(\Lambda) \in \widehat{D}$, $v(\Lambda) \in \widehat{D}$ et il existe $l \to r \in R$ tel que*
   – *$u(\Lambda) = \widehat{l(\Lambda)}$ (simplement $l(\Lambda)$ dans le cas non marqué) et*
   – *$v(\Lambda) = \widehat{f}$ (simplement f dans la cas non marqué) pour un symbole f de r.*
   (b) *$\rightarrow_{mh}$ est induite par un système fini R_{mh} tel que pour tout $l \to r \in R_{mh}$ il y a un symbole AC f vérifiant $l(\Lambda) = r(\Lambda) = \widehat{f}$ et les sous-termes stricts de l et r ne peuvent contenir que $\widehat{f}$, f et des variables.*
   (c) *R termine si et seulement s'il n'y a aucune séquence infinie $\langle u_i, v_i\rangle_i$ telle que pour tout i :*

$$v_i\sigma \ (\xrightarrow[AC\setminus R]{\neq\Lambda} \cup \rightarrow_{mh})^* \ u_{i+1}\sigma$$

Le système R_{mh} n'est pas supposé terminant. Il est vide dans le cas non marqué.

Ces conditions abstraites définies, on peut les instancier avec différentes approches concrètes des paires de dépendance AC :

– Kusakari et Toyama [31] ne considèrent que des symboles d'arité fixe et dans leur variante avec marques le système R_{mh} est constitué des règles : $\widehat{f}(\widehat{f}(x,y),z) \leftrightarrow \widehat{f}(f(x,y),z)$ et $\widehat{f}(\widehat{f}(x,y),z) \to \widehat{f}(x,y)$ pour tout symbole f AC.
– Giesl et Kapur [19] ne considèrent que des symboles d'arité fixe et le système R_{mh} est constitué des règles : $\widehat{f}(f(x,y),z) \leftrightarrow \widehat{f}(x,f(y,z))$ et $\widehat{f}(x,y) \leftrightarrow \widehat{f}(y,x)$ pour tout symbole f AC.
– Kusakari, Marché et Urbain [29] considèrent des termes aplatis et le système R_{mh} est constitué des règles $\widehat{f}(x,y,z) \to \widehat{f}(f(x,y),z)$ pour tout symbole f AC.

Une instance concrète non marquée Nous présentons les paires de dépendance AC dans leur forme non marquée proposées par Kusakari, Marché et Urbain [29].

Définition 7.49 *Soit R un système sur $\mathcal{T}(\mathcal{F} \cup \mathcal{F}_{AC}, X)$. À l'ensemble des paires de dépendance « classiques » d'une règle $f(t_1,\ldots,t_n) \to r$ on ajoute, si $f \in \mathcal{F}_{AC}$, celles de $f(t_1,\ldots,t_n,x) \to \overline{f(r,x)}$*
*où x est une nouvelle variable n'apparaissant pas dans la règle (*paires étendues*).*

L'union de ces paires forme alors l'ensemble des paires de dépendance AC*.*

Afin d'obtenir un représentant canonique de la classe d'équivalence AC de $f(r,x)$ l'aplatissement est requis si $r(\Lambda) = f$.

Remarque 7.50 *Si la règle ne demande pas de filtrage étendu alors il n'est pas nécessaire de considérer ses paires étendues.*

Exemple 7.51 *Pour un système de calcul dans l'arithmétique de Peano avec $+$ et $\times$ AC :*

$$\begin{array}{lcllcl} x+0 & \to & x & x\times 0 & \to & 0 \\ x+s(y) & \to & s(x+y) & x\times s(y) & \to & (x\times y)+x \end{array}$$

On a 6 paires étendues (à droite) qui donnent au total 9 paires AC.

$$\begin{array}{ll} \langle x+s(y), x+y\rangle & \langle x+s(y)+z, x+y\rangle \\ \langle x\times s(y), x\times y\rangle & \langle x+s(y)+z, s(x+y)+z\rangle \\ \langle x\times s(y), (x\times y)+x\rangle & \langle x\times 0\times z, 0\times z\rangle \\ & \langle x\times s(y)\times z, x\times y\rangle \\ & \langle x\times s(y)\times z, ((x\times y)+x)\rangle \\ & \langle x\times s(y)\times z, ((x\times y)+x)\times z\rangle \end{array}$$

Il existe bien une notion de minimalité des contre-exemples à la terminaison AC :

Lemme 7.52 *Soit R un système AC sur $\mathcal{T}(\mathcal{F},\mathcal{X})$ qui ne termine pas. Soit t un terme non fortement normalisable, alors t contient un sous-terme $u = f(u_1,\ldots,u_n)$ tel que*[3] *:*

1. *u est non fortement normalisable ;*
2. *Tous les u_i, $1 \le i \le n$, sont fortement normalisables ;*
3. *f est un symbole défini ;*
4. *Si $f \in \mathcal{F}_{AC}$, alors il existe un k, $2 \le k \le n$, tel que $f(u_1,\ldots,u_k)$ est non fortement normalisable mais $f(u_1,\ldots,u_{k-1})$ est fortement normalisable.*

Théorème 7.53 *La définition 7.49 concrétise la notion abstraite de paires de dépendances AC (déf. 7.48). En particulier R termine AC s'il n'existe pas de séquence infinie $\langle u_i, v_i\rangle_i$ telle que pour tout i :*

$$v_i\sigma \ (\xrightarrow[AC\setminus R]{\neq\Lambda} \cup \to_{mh})^* \ u_{i+1}\sigma$$

Si on souhaite obtenir des paires de dépendance marquées il faudra définir un système R_{mh} convenable. En particulier, l'approche naïve qui consiste à simplement marquer les paires de dépendance AC sans marques échoue.

Contre-exemple 7.54 *Considérons le système R sur $\mathcal{T}(\mathcal{F},\mathcal{X})$:*

$$f(x) \to x+a \qquad c+a \to f(b+c)$$

Ce système ne termine pas, on a la réduction infinie

$$f(b+c) \to b+c+a \to b+f(b+c) \to \cdots$$

Les paires de dépendance AC de R sont :

$$\begin{array}{ll} \langle f(x), x+a\rangle & \langle c+a+x, f(b+c)+x\rangle \\ \langle c+a, f(b+c)\rangle & \langle c+a+x, f(b+c)\rangle \\ \langle c+a, b+c\rangle & \langle c+a+x, b+c\rangle \end{array}$$

[3] On omet ici le cas u constante.

Elles permettent la chaîne infinie :

$$\langle f(b+c), b+c+a\rangle, \langle c+a+b, f(b+c)\rangle, \langle f(b+c), b+c+a\rangle, \ldots$$

Si on se contente de marquer les symboles de tête des paires on obtient

$$\begin{array}{ll} \langle \widehat{f}(x), x\widehat{+}a\rangle & \langle c\widehat{+}a\widehat{+}x, f(b+c)\widehat{+}x\rangle \\ \langle c\widehat{+}a, \widehat{f}(b+c)\rangle & \langle c\widehat{+}a\widehat{+}x, \widehat{f}(b+c)\rangle \\ \langle c\widehat{+}a, b\widehat{+}c\rangle & \langle c\widehat{+}a\widehat{+}x, b\widehat{+}c\rangle \end{array}$$

La chaîne correspondante devient

$$\langle \widehat{f}(b+c), (b+c)\widehat{+}a\rangle$$

forcément finie car aucune paire ne peut suivre $(b+c)\widehat{+}a$.

7.4.3 Prouver la terminaison AC à l'aide d'ordres

Théorème 7.55 *Soit R un système sur $\mathcal{T}(\mathcal{F},\mathcal{X})$ comportant des symboles AC. Soit $\mathscr{D}$ l'ensemble des paires de dépendance AC de R. S'il existe une paire d'ordre $(\succeq, >)$ compatible AC (sur les symboles AC non marqués), bien fondée, faiblement monotone telle que :*

1. $l \succeq r$ *pour toute règle* $l \to r \in R$,
2. $u > v$ *pour toute paire* $\langle u, v\rangle \in \mathscr{D}$
3. $l \succeq r$ *pour toute règle* $l \to r \in R_{mh}$,

alors R termine modulo AC.

Exemple 7.56 *Considérons le système R suivant, où $\cup$ et $\cap$ sont AC et eq commutatif, qui calcule l'intersection multi-ensembliste. Les multi-ensembles d'entiers de Peano sont représentés par la constante $\emptyset$, le symbole $\{\_\}$ qui construit un singleton et l'union.*

$$\begin{array}{lcllcl} \mathit{if}(\mathit{true}, x, y) & \to & x & \emptyset \cup x & \to & x \\ \mathit{if}(\mathit{false}, x, y) & \to & y & x \cap \emptyset & \to & \emptyset \\ \mathit{eq}(0,0) & \to & \mathit{true} & x \cap (y \cup z) & \to & (x \cap y) \cup (x \cap z) \\ \mathit{eq}(0, s(x)) & \to & \mathit{false} & \{x\} \cap \{y\} & \to & \mathit{if}(\mathit{eq}(x,y), \{x\}, \emptyset) \\ \mathit{eq}(s(x), s(y)) & \to & \mathit{eq}(x,y) & & & \end{array}$$

On a les paires AC :

$$\begin{array}{lll} \langle \mathit{eq}(s(x), s(y)), \mathit{eq}(x,y)\rangle & \langle x \cap (y \cup z), x \cap y\rangle & \langle z \cap \{x\} \cap \{y\}, \mathit{if}(\mathit{eq}(x,y), \{x\}, \emptyset)\rangle \\ \langle \{x\} \cap \{y\}, \mathit{if}(\mathit{eq}(x,y), \{x\}, \emptyset)\rangle & \langle x \cap (y \cup z), x \cap z\rangle & \langle z \cap \{x\} \cap \{y\}, z \cap \mathit{if}(\mathit{eq}(x,y), \{x\}, \emptyset)\rangle \\ \langle \{x\} \cap \{y\}, \mathit{eq}(x,y)\rangle & \langle y \cap x \cap \emptyset, y \cap \emptyset\rangle & \langle t \cap x \cap (y \cup z), (x \cap y) \cup (x \cap z)\rangle \\ \langle x \cap (y \cup z), (x \cap y) \cup (x \cap z)\rangle & \langle z \cap \{x\} \cap \{y\}, \mathit{eq}(x,y)\rangle & \langle t \cap x \cap (y \cup z), t \cap ((x \cap y) \cup (x \cap z))\rangle \\ \langle t \cap x \cap (y \cup z), x \cap z\rangle & \langle t \cap x \cap (y \cup z), x \cap y\rangle & \end{array}$$

L'ordre induit par l'interprétation polynomiale :

$$\begin{array}{ll} [\![\mathit{true}]\!] = [\![\mathit{false}]\!] = [\![0]\!] = [\![\emptyset]\!] = 0 & [\![\{\_\}]\!](x) = x \\ [\![\mathit{if}]\!](x_0, x_1, x_2) = x_2 + x_1 & [\![\cup]\!](x_0, x_1) = x_1 + x_0 + 2 \\ [\![s]\!](x) = x + 1 & [\![\cap]\!](x_0, x_1) = 2x_0x_1 + 2x_1 + 2x_0 + 1 \\ [\![\mathit{eq}]\!](x_0, x_1) = x_0x_1 & \end{array}$$

est compatible AC et suffit à prouver la terminaison AC.

7.5 Terminaison sous stratégies de réduction

La notion de terminaison n'est évidemment pas la même lorsqu'on décide qu'une certaine stratégie de récriture sera appliquée. Cela peut être pour certaines raisons d'efficacité, pour mimer le comportement d'une machine abstraite, d'un langage, etc. Dans ce cas on s'intéresse à la terminaison selon cette stratégie.

7.5.1 Innermost

La stratégie *innermost* correspond à une stratégie d'appel par valeurs.

Définition 7.57 *Soit R un système de récriture sur $\mathcal{T}(\mathcal{F},\mathcal{X})$. La relation $\underset{R}{\rightarrow}_{\mathrm{i}}$ est définie par $s \underset{R}{\rightarrow}_{\mathrm{i}} t$ si et seulement s'il existe une règle $l \rightarrow r \in R$ une position $p \in \mathcal{P}os(s)$ et une substitution σ telles que $s \xrightarrow[l\rightarrow r,\sigma]{p} t$ et $\forall q >_{pref} p$, $s|_p$ est normalisé innermost. On dit alors que s se réduit selon la stratégie innermost en t.*

Un système R *termine innermost* si $\underset{R}{\leftarrow}_{\mathrm{i}}$ est bien fondée.

La relation innermost est bien sûr contenue dans la relation de récriture sans stratégie.

Théorème 7.58 *Si un système R termine alors R termine innermost.*

La réciproque est fausse.

Contre-exemple 7.59 *Considérons le système comportant les deux règles :*

$$f(a) \rightarrow f(a) \qquad a \rightarrow b$$

Ce système termine innermost mais pas sans contrainte de stratégie puisqu'on a $f(a) \rightarrow f(a) \rightarrow \cdots$

Paires de dépendances et stratégie innermost

Savoir que les sous-termes stricts des termes réduits sont en forme normale nous donne beaucoup d'information sur leur structure. Puisqu'on s'intéresse à la stratégie innermost on sait déjà que toutes les règles du système ne sont pas applicables entre deux instances de paires : on va pouvoir ne considérer que certaines d'entre elles dans la preuve de terminaison. Ensuite, les différentes occurrences d'une même variable dans le même droit d'une paire sont instanciées en forme normale : l'approximation du graphe n'a plus à les différencier.

En suivant la stratégie innermost, on ne peut pas appliquer une règle dont le membre gauche contient strictement un redexe. Arts et Giesl [3] utilisent cette constatation pour se restreindre à un ensemble de règles utilisables entre deux paires de dépendance.

Définition 7.60 *Soit R un système sur $\mathcal{T}(\mathcal{F},\mathcal{X})$. On note*

$$NR(R,f) = \{l \rightarrow r \in R \mid l(\Lambda) = f \text{ et } l \text{ n'a pas de redexe en sous-terme strict}\}$$

L'ensemble des règles utilisables *de R pour un terme t est défini par :*

$$\begin{cases} U(R,x) = \emptyset \quad \text{pour toute variable } x \\ U(R,f(u_1,\ldots,u_n)) = \quad NR(R,f) \cup \bigcup_{j=1}^{n} U(R',u_j) \cup \bigcup_{l\rightarrow r\in NR(R,f)} U(R',r) \\ \qquad\qquad\qquad\qquad \text{où } R' = R \setminus NR(R,f) \end{cases}$$

On adapte la relation DPR à la stratégie innermost.

Définition 7.61 *On définit maintenant la relation $\leftarrow_{\mathsf{DPR}_i(\mathscr{D},R)}$ par par $s \rightarrow_{\mathsf{DPR}_i(\mathscr{D},R)} t$ si :*

- *Tous les sous-termes stricts de s et t sont fortement normalisables,*
- *Il existe σ (normale[4]) telle que $s \xrightarrow[U(R,s)]{\neq\Lambda\ *}_{\mathrm{i}} s' \equiv u\sigma \xrightarrow[\langle u,v\rangle\in\mathscr{D}]{} t$,*
- *Tous les sous-termes stricts de s' sont en forme normale.*

De façon similaire au cas de la récriture standard la bonne fondation de $\leftarrow_{\mathsf{DPR}_i(\mathrm{DP}(R),R)}$ est équivalente à la bonne fondation de $\leftarrow_R$.

Remarque 7.62 *Puisqu'un membre droit de paire est instancié par une substitution* normale*, ses seuls sous-termes intéressants du point de vue de la terminaison sont ceux ayant en tête un symbole défini.*

Cas des graphes La notion de graphe de dépendance innermost est similaire au cas standard mais en considérant cette fois la relation innermost pour joindre les nœuds.

Définition 7.63 *Soit R un système de récriture. On appelle* graphe de dépendance innermost de R *le graphe $\mathcal{G}$ dont les nœuds sont les paires de dépendance de R et tel que $(\langle s,t\rangle, \langle s',t'\rangle) \in \mathcal{G}$ si et seulement s'il existe une substitution σ normale vérifiant $t\sigma \xrightarrow{\neq\Lambda\ *}_{\mathrm{i}} s'\sigma$ et tout sous-terme strict de s' est en forme normale.*

Ce graphe innermost n'est pas plus calculable que le graphe standard.

Théorème 7.64 (Critère par graphe innermost) *Soit R un système de récriture sur $\mathcal{T}(\mathcal{F},\mathcal{X})$, soit $\mathcal{G}$ son graphe de dépendance innermost. Soient $\mathcal{G}_1,\ldots,\mathcal{G}_k$ les k parties fortement connexes de $\mathcal{G}$ (et donc $\mathcal{G}_j \in \mathrm{DP}(R)$ pour tout $j \leq k$). Toutes les $\leftarrow_{\mathsf{DPR}_i(\mathcal{G}_j,R)}$ sont bien fondées si et seulement si $\leftarrow_{\mathsf{DPR}_i(\mathrm{DP}(R),R)}$ est bien fondée.*

Définition 7.65 *Soit R un système de récriture sur $\mathcal{T}(\mathcal{F},\mathcal{X})$. Pour tout $\langle s,t\rangle \in \mathrm{DP}(R)$,*

$\mathrm{CAP}_s(t)$ désigne le terme obtenu en remplaçant par des variables les sous-termes de t dont le symbole à la racine est défini dans R et qui ne sont pas des sous-termes de s.

Un terme t d'une paire $\langle s,t\rangle$ est connectable *à un terme s' si $\mathrm{CAP}_s(t)$ et s' sont unifiables par un mgu σ tel que $s\sigma$ et $s'\sigma$ sont en forme normale.*

Le graphe obtenu contient bien le graphe de dépendance innermost.

Prouver que $\mathsf{DPR}_i(\mathscr{D},R)$ termine à l'aide d'ordres

Proposition 7.66 *Soit R un système de récriture défini sur $\mathscr{T}(\mathcal{D}\cup\mathcal{C},X)$ et soit $\mathscr{D} \subseteq \mathrm{DP}(R)$. S'il existe une paire d'ordres $(\succeq,>)$ bien fondée, et telle que pour toute paire $\langle u,v\rangle \in \mathscr{D}$:*

1. *$l \succeq r$ pour toute règle $l \to r \in U(R,v)$,*
2. *$u > v$,*
3. *$\succeq$ est faiblement monotone sur $\mathcal{F}$ et tel que pour $v = C[v_1,\ldots,v_n]$ avec $v_i(\Lambda) \in \mathcal{D}$ on a que $x_1 \succeq y_1,\ldots,x_n \succeq y_n$ implique $C[x_1,\ldots,x_n] \succeq C[y_1,\ldots,y_n]$,*

alors $\leftarrow_{\mathsf{DPR}(\mathscr{D},R)}$ est bien fondée.

Le point 3 signifie en fait que la monotonie faible n'est nécessaire qu'aux positions définies des paires marquées conformément à la remarque 7.62. Si les paires ne sont pas marquées on ne peut pas différencier ces étapes des pas de récriture et la monotonie faible doit être vérifiée pour toutes les positions.

[4] C'est-à-dire que pour toute variable x, $x\sigma$ est en forme normale.

Applications de la terminaison innermost

On a vu au paragraphe précédent que prouver la terminaison innermost pouvait être plus facile que prouver la terminaison en toute généralité, notamment en tirant parti de la structure des termes au cours d'une réduction innermost. On peut en profiter sur des classes de systèmes pour lesquelles la terminaison innermost implique la terminaison [20, 21].

Théorème 7.67 (Gramlich [20]) *Soit R un système sur $\mathcal{T}(\mathcal{F}, \mathcal{X})$ sans paire critique. Si R termine pour la stratégie innermost alors R termine.*

Chapitre 8

Compilation du fitrage

Dans tout ce chapitre, on suppose fixée une algèbre de termes $\mathcal{T}(\mathcal{F}, \mathcal{X})$.

8.1 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 8.1 (Squelette, frange) *Un squelette S est un terme (clos) bien formé de $\mathcal{T}(\mathcal{F} \cup \{\Box\})$, où $\Box$ 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*

$$F(S) = \{p \in \mathcal{P}os(S) \mid S(p) = \Box\}$$

Un terme t est compatible avec un squelette S si et seulement si

$$\forall p \in \mathcal{P}os(S),\ p \notin F(S) \Longrightarrow \begin{cases} p \in \mathcal{P}os(t) \wedge S(p) = t(p) \\ \textit{ou} \\ \exists q \in \mathcal{P}os(t),\ q \leq p \wedge t(q) \in \mathcal{X} \end{cases}$$

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 8.2 (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 $\Box$,*
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 $u \to^f v$ 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.

8.2 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 récriture varie.

Chapitre 9

Complétion

Nous avons vu que le problème du mot est décidable dans une théorie équationnelle E lorsque l'on dispose d'un système de récriture 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 récriture 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

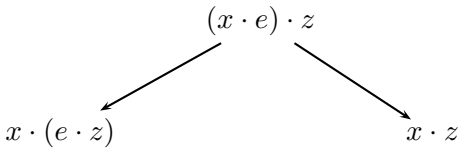
$$E = \{x \cdot e = x; x \cdot I(x) = e; (x \cdot y) \cdot z = x \cdot (y \cdot z)\}$$

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 récriture 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



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.

9.1 Règles de complétion

Le processus de complétion est décrit comme un ensemble de règles d'inférence. Les entrées 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 récriture $\geq$. Les règles manipulent une paire (E, R) composée d'un ensemble d'équations E (qui restent à transformer en règles de récriture) et un ensemble de règles de récriture R.

Initial $\quad \dfrac{}{E_0 \ ; \ \emptyset}$

Trivial $\quad \dfrac{E \cup \{s = s\} \ , \ R}{E \ , \ R}$

Paire Critique $\quad \dfrac{E \ , \ R}{E \cup \{e\} \ , \ R}$ si e est une paire critique de R

Oriente $\quad \dfrac{E \cup \{s = t\} \ , \ R}{E \ , \ R \cup \{s \to t\}}$ si $s \succ t$

Simplifie $\quad \dfrac{E \cup \{s = t\} \ , \ R}{E \cup \{s' = t'\} \ , \ R}$ si $s \to_R s'$ et/ou $t \to_R t'$

Compose $\quad \dfrac{E \ , \ R \cup \{l \to r\}}{E \ , \ R \cup \{l \to r'\}}$ si $r \to_R r'$

Collapse $\quad \dfrac{E \ , \ R \cup \{l \to r\}}{E \cup \{l' = r\} \ , \ R}$ si $l \to_{g \to d} l', g \to d \in R$, et (($l|_p = g\sigma$ avec $p \neq \Lambda$ ou σ n'est pas un renommage) ou ($l = g\sigma$ et $r \succ d\sigma$ avec σ qui est un renommage)).

Lemme 9.1 *Chacune des règles d'inférences de l'ensemble ci-dessus :*

$$\frac{E \ , \ R}{E' \ , \ R'}$$

conserve la théorie équationnelle : $=_{E \cup E_R} \equiv =_{E' \cup E_{R'}}$.

La complétion peut être entièrement automatisée si l'on dispose d'un algorithme pour décider le préordre de récriture 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.

9.2 Stratégies et contrôle équitable

Les règles de la complétion sont hautement non-deterministes, et même s'il existe un système de récriture convergent compatible avec l'ordre de la complétion et dont la théorie équationnelle est équivalente à celle de l'ensemble d'équations de départ, uns mauvaise statégie ne le trouvera pas et ne terminera pas en temps fini.

On note une excécution de la complétion comme suit :

$$(E_0, \emptyset) \vdash (E_1, R_1) \vdash \ldots (E_n, R_n) \vdash (E_{n+1}, R_{n+1}) \ldots$$

Définition 9.2 (Équation/Règle persistante) *Une équation e (resp. une règle r) est dite persistante si elle appartient à tous les E_n (resp. R_n) à partir d'un certain rang n_0, c'est-à-dire :*

$$e \in \bigcup_{n_0} \bigcap_{n \geq n_0} E_n \qquad r \in \bigcup_{n_0} \bigcap_{n \geq n_0} R_n$$

L'ensemble des équations persistantes $\bigcup_{n_0} \bigcap_{n \geq n_0} E_n$ est noté E_ω et l'ensemble des règles persistantes $\bigcup_{n_0} \bigcap_{n \geq n_0} R_n$ est noté R_ω.

Supposons qu'il existe un système de récriture équivalent à E_0, confluent et compatible avec l'ordre de la complétion. On peut en prendre une version interréduite, qui est alors canonique et unique. On note ce système R_{E_0}. Dans ce cas, le but souhaité de la complétion est bien évidemment de réussir en produisant en sortie R_{E_0}. On dit que la complétion est équitable si toute preuve par récriture dans R_{E_0} peut se faire dans l'un des R_n, ce qui revient précisément à

$$R_{E_0} = R_\omega$$

Définition 9.3 *Une séquence de complétion est un succès si $E_\omega = \emptyset$.*

Une paire critique est persistante si c'est une paire critique de R_ω.

Exemple 9.4

$$R = \begin{cases} f(g(h(x))) & \to & g(x) \\ g(h(g(x))) & \to & g(g(h(x))) \\ f(g(g(x))) & \to & f(g(x)) \\ h(h(x)) & \to & h(x) \end{cases}$$

Si le contrôle évite de traiter la troisième règle, on n'obtient pas le système canonique équivalent :

$$R' = \begin{cases} f(g(x)) & \to & g(x) \\ h(h(x)) & \to & h(x) \\ g(h(x)) & \to & g(x) \\ g(g(x)) & \to & g(x) \end{cases}$$

On se restreindra dans la suite à des contrôles qui traitent les paires critiques persistantes :

Définition 9.5 *Une séquence de complétion est équitable si $CP(R_\omega) \subset \bigcup_n E_n$.*

9.3 Algèbres de preuves, la complétion vue comme récriture de preuves

Soit $(E_0, \emptyset) \vdash (E_1, R_1) \vdash \ldots (E_n, R_n) \vdash (E_{n+1}, R_{n+1}) \ldots$ une exécution de la complétion. Le but de la complétion est de transformer toutes les preuves équationnelles qui utilisent des équations de E_0 en des preuves par récriture dans R_ω. Nous allons formaliser ces transformations en nous plaçant dans un cadre plus large qui peut accueillir toutes ces preuves à la fois, y compris les preuves « intermédiaires ».

On note donc $E_\infty = \bigcup_n E_n$ et $R_\infty = \bigcup_n R_n$.

Définition 9.6 *Une preuve (non trivale) de $s = t$ dans $E_\infty \cup R_\infty$ est une séquence mixte (non vide) qui combine des étapes équationnelles de E_∞ et des récritures de R_∞, $s = s_0, s_1, \ldots, s_{n-1}, s_n = t$ telle que :*

- $s_{i-1} \leftrightarrow_{E_\infty} s_i$,
- *ou bien* $s_{i-1} \to_{R_\infty} s_i$,
- *ou bien* $s_{i-1} \leftarrow_{R_\infty} s_i$.

Deux preuves sont équivalentes si elles ont mêmes premiers et derniers termes.

Le processus de complétion transforme peu à peu une preuve équationelle de E_0 en une preuve par récriture dans R_ω équivalente. Cette transformation s'accompagne de la décroissance d'une mesure, le *coût de la preuve*.

Définition 9.7 (Coût d'une preuve) *Le coût d'une étape de preuve est un triplet défini comme suit :*
- *Le coût de* $s_{i-1} \leftrightarrow_{E_\infty} s_i$ *est* $(\{s_{i-1}, s_i\}, \_, \_)$.
- *Le coût de* $s_{i-1} \rightarrow_{R_\infty} s_i$*, si la règle utilisée est* $l \rightarrow r$*, est* $(\{s_{i-1}\}, l, s_i)$.
- *Le coût de* $s_{i-1} \leftarrow_{R_\infty} s_i$*, si la règle utilisée est* $l \rightarrow r$*, est* $(\{s_i\}, l, s_{i-1})$.

Le coût total d'une preuve est le multi-ensemble de ses coûts élémentaires.

Les coûts élementaires sont comparés lexicographiquement :
- La première composante est comparée avec l'extension multi-ensemble de l'ordre utilisé pour la complétion.
- La deuxième composante est comparée avec la partie stricte de l'ordre de plongement.
- La troisième composante est comparée avec l'ordre de la complétion.

Les coûts totaux sont comparés avec l'extension multi-ensemble de l'ordre utilisé pour les coûts élementaires.

Lemme 9.8 *Si l'ordre utilisé par la complétion est un pré-ordre de récriture, l'ordre sur les preuves est bien fondé.*

Lemme 9.9 *Soit* $(E_0, \emptyset) \vdash (E_1, R_1) \vdash \ldots (E_n, R_n) \vdash (E_{n+1}, R_{n+1}) \ldots$ *une exécution de la complétion équitable (*$CP(R_\omega) \subset E_\infty$*) et qui n'échoue pas (*$E_\omega = \emptyset$*). Soit* P *une preuve dans* $E_\infty \cup R_\infty$ *qui n'est pas une preuve par récriture dans* R_ω*. Il existe une preuve* P' *de* $E_\infty \cup R_\infty$*, équivalente à* P *et de coût strictement inférieur.*

Démonstration Examinons les raisons pour lesquelles P n'est pas une preuve par récriture de R_ω :
- P contient une étape $s_{i-1} \leftrightarrow_{E_\infty} s_i$ qui utilise l'égalité $u = v$ de E_∞. Comme la complétion n'échoue pas, l'égalité $u = v$ n'est pas persistante. Elle disparaît de l'ensemble des équations à un moment donné de la complétion.
  - L'équation $u = v$ peut disparaître par la règle **Trivial**. Dans ce cas, il suffit de supprimer l'étape élementaire $s_{i-1} \leftrightarrow_{E_\infty} s_i$ de la preuve P pour obtenir une preuve strictement plus petite.
  $$s_0 \ldots s_{i-1} \leftrightarrow_{E_\infty} s_i s_{i+1} \ldots s_n \quad \rightsquigarrow \quad s_0 \ldots s_{i-1} s_{i+1} \ldots s_n$$
  - L'équation $u = v$ peut être orientée et passer dans l'ensemble des règles. Supposons sans perte de généralité que l'équation se transforme en la règle $u \rightarrow v$ (le cas $v \rightarrow u$ est symétrique). L'étape $\leftrightarrow_{E_\infty}$ de coût $(\{s_{i-1}, s_i\}, \_, \_)$ entre s_{i-1} et s_i se transforme en $s_{i-1} \rightarrow_{R_\infty} s_i$ de coût $(\{s_{i-1}\}, u, s_{i-1})$ strictement inférieur.
  $$s_0 \ldots s_{i-1} \leftrightarrow_{E_\infty} s_i \ldots s_n \quad \rightsquigarrow \quad s_0 \ldots s_{i-1} \rightarrow_{R_\infty} s_i \ldots s_n$$
  - L'équation $u = v$ peut être récrite par une règle $l \rightarrow r$ de R_∞. Supposons sans perte de généralité que c'est le terme u qui est récrit en u'. E_∞ contient donc l'équation $u' = v$. L'étape élementaire
  $$s_{i-1}[u\sigma] \leftrightarrow_{E_\infty} s_i[v\sigma]$$
  de coût $(\{s_{i-1}[u\sigma], s_i[v\sigma]\}, \_, \_)$ peut être remplacée par les deux étapes
  $$s_{i-1}[u\sigma] \rightarrow_{R_\infty} s_{i-1}[u'\sigma] \quad \text{et} \quad s_{i-1}[u'\sigma] \leftrightarrow_{E_\infty} s_i[v\sigma]$$
  de coûts respectifs $(\{s_{i-1}[u\sigma]\}, l, s_{i-1}[u'\sigma])$ et $(\{s_{i-1}[u'\sigma], s_i[v\sigma]\}, \_, \_)$ tous deux strictement inférieurs ($l \rightarrow r$, donc $l \succ r$, donc $u' \succ u$, donc $s_{i-1}[u\sigma] \succ s_{i-1}[u'\sigma]$).
  $$s_0 \ldots s_{i-1}[u\sigma] \leftrightarrow_{E_\infty} s_i[v\sigma] \ldots s_n \quad \rightsquigarrow \quad s_0 \ldots s_{i-1}[u\sigma] \rightarrow_{R_\infty} s_{i-1}[u'\sigma] \leftrightarrow_{E_\infty} s_i[v\sigma] \ldots s_n$$
- P contient une étape $s_{i-1} \rightarrow_{R_\infty} s_i$ qui utilise une règle $l \rightarrow r$ de $R_\infty \setminus R_\omega$. La règle $l \rightarrow r$ ne peut disparaître par **Compose** ou par **Collapse**.

– La règle $l \to r$ a été remplacée par la règle $l \to r'$ où r s'est récrit en r' avec $g \to d$. La preuve élémentaire

$$s_{i-1}[l\sigma] \to_{R_\infty} s_i[r\sigma]$$

de coût $(\{s_{i-1}[l\sigma]\}, l, s_i[r\sigma])$ peut être remplacée par les deux étapes

$$s_{i-1}[l\sigma] \to_{R_\infty} s_{i-1}[r'\sigma] \quad \text{et} \quad s_{i-1}[r'\sigma] \leftarrow_{R_\infty} s_i[r\sigma]$$

de coûts respectifs $(\{s_{i-1}[l\sigma]\}, l, s_{i-1}[r'\sigma])$ et $(\{s_i[r\sigma]\}, g, s_{i-1}[r'\sigma])$ tous deux strictement inférieurs.

$$s_0 \dots s_{i-1}[l\sigma] \to_{R_\infty} s_i[r\sigma] \dots s_n \qquad \rightsquigarrow \qquad s_0 \dots s_{i-1}[l\sigma] \to_{R_\infty} s_{i-1}[r'\sigma] \leftarrow_{R_\infty} s_i[r\sigma] \dots s_n$$

– La règle $l \to r$ a été remplacée par l'équation $l' = r$ où l s'est récrit en l' avec $g \to d$. La preuve élémentaire

$$s_{i-1}[l\sigma] \to_{R_\infty} s_i[r\sigma]$$

de coût $(\{s_{i-1}[l\sigma]\}, l, s_i[r\sigma])$ peut être remplacée par les deux étapes

$$s_{i-1}[l\sigma] \to_{R_\infty} s_{i-1}[l'\sigma] \quad \text{et} \quad s_{i-1}[l'\sigma] \leftrightarrow_{E_\infty} s_i[r\sigma]$$

de coûts respectifs $(\{s_{i-1}[l\sigma]\}, g, s_{i-1}[l'\sigma])$ et $(\{s_{i-1}[l'\sigma], s_i[r\sigma]\}, \_, \_)$ tous deux strictement inférieurs (les conditions d'application de la règle **Collapse** assurent que $l > g$ pour l'ordre de plongement, ou que l et g sont renommages l'un de l'autre, mais que dans ce cas, $r\sigma \succ l'\sigma$).

$$s_0 \dots s_{i-1}[l\sigma] \to_{R_\infty} s_i[r\sigma] \dots s_n \qquad \rightsquigarrow \qquad s_0 \dots s_{i-1}[l\sigma] \to_{R_\infty} s_{i-1}[l'\sigma] \leftrightarrow_{E_\infty} s_i[r\sigma] \dots s_n$$

– P ne contient que des étapes $\to_{R_\omega}$ et $\leftarrow_{R_\omega}$, mais n'est pas une preuve par récriture car elle contient un pic $s_{i-1} \leftarrow_{R_\omega} s_i \to_{R_\omega} s_{i+1}$. Comme dans la preuve du lemme des paires critiques 5.11, on peut distinguer s'il s'agit d'un pic critique, ou d'un pic non critique que l'on peut joindre.
  – S'il s'agit d'un pic non critique, on le remplace par des étapes $\to_{R_\omega}$ et $\leftarrow_{R_\omega}$ entre des termes qui sont tous strictement inférieurs au sommet du pic s_i. La sous-séquence $s_{i-1} \leftarrow_{R_\omega} s_i \to_{R_\omega} s_{i+1}$ est donc remplacée par une autre séquence de coût strictement plus petit.

$$s_0 \dots s_{i-1} \leftarrow_{R_\omega} s_i \to_{R_\omega} s_{i+1} \dots s_n \qquad \rightsquigarrow \qquad s_0 \dots s_{i-1} \to^*_{R_\omega} \leftarrow^*_{R_\omega} s_{i+1} \dots s_n$$

  – S'il s'agit d'un pic critique entre deux règles $l \to r$ et $g \to d$, comme le contrôle est équitable, la paire critique associée $r\phi = l\phi[d\phi]_p$ va être calculée et ajoutée à l'ensemble des équations, elle appartient donc à E_∞. Elle permet en outre une étape équationnelle entre s_{i-1} et s_{i+1}. Le pic

$$s_{i-1} \leftarrow_{R_\omega} s_i \to_{R_\omega} s_{i+1}$$

de coût $(\{s_i[l\sigma]\}, l, s_{i-1}[r\sigma]); \{s_i[g\tau]\}, g, s_{i+1}[d\tau])$ est donc remplacé par une étape

$$s_{i-1} \leftrightarrow_{E_\infty} s_{i+1}$$

de coût $(\{s_{i-1}[r\sigma], s_{i+1}[d\tau]\}, \_, \_)$.

$$s_0 \dots s_{i-1} \leftarrow_{R_\omega} s_i \to_{R_\omega} s_{i+1} \dots s_n \qquad \rightsquigarrow \qquad s_0 \dots s_{i-1} \leftrightarrow_{E_\infty} s_{i+1} \dots s_n$$

□

Théorème 9.10 *Soit $(E_0, \emptyset) \vdash (E_1, R_1) \vdash \dots (E_n, R_n) \vdash (E_{n+1}, R_{n+1}) \dots$ une exécution de la complétion, équitable et qui n'échoue pas.*

– *Toute preuve de $E_\infty \cup R_\infty$ est équivalente à une preuve par récriture dans R_ω.*
– *R_ω est équivalent à E_0.*

– *R_ω est convergent.*
– *Si R_ω est fini, le problème du mot pour E_0 est décidable. Sinon, cette exécution fournit une procédure de semi-décision pour l'égalité modulo E_0.*

La complétion échoue dans le cas où l'ensemble des équations persistantes est non-vide. Pour résoudre ce problème, deux solutions sont envisageables. Elles sont basées sur la même approche, une modification de la notion de récriture, et les modifications subséquentes dans les règles d'inférence de la complétion de façon à pouvoir démontrer un théorème similaire au théorème 9.10.

9.4 Complétion ordonnée

La récriture utilisée ici est la récriture *ordonnée* définie par rapport à un pré-ordre de récriture $\succeq$ que l'on supposera fixé dans le reste de cette section. L'idée est que les équations ne sont pas orientées une fois pour toutes, car en général suivant les instances, l'orientation par rapport à $\succeq$ peut varier. Les équations seront utilisées pour récrire si l'instance utilisée décroît.

Définition 9.11 (Récriture ordonnée) *Soit s et t deux termes, et E une théorie équationnelle. $s \to_{E\succ} r$ si $s \leftrightarrow_E$ et $s \succ t$.*

La complétion travaille sur un ensemble d'équations, il n'y a plus de règles. En fait les règles sont simplement des équations qui s'orientent toujours dans le même sens, quelles que soient les instances utilisées. On définit donc une notion de paire critique adaptée à la récriture ordonnée :

Définition 9.12 (Paire critique ordonnée) *Soient $s = t$ et $u = v$ deux équations, telles qu'il existe un renommage ρ de s, une position p non variable de s, avec $s\rho|_p$ et u unifiables. Soit σ un unificateur plus général de $s\rho|_p$ et u. S'il existe une substitution close τ telle que*

$$s\rho\sigma\tau \succ t\rho\sigma\tau \quad \textit{et} \quad s\rho\sigma\tau[u\sigma\tau]_p \succ s\rho\sigma\tau[v\sigma\tau]_p$$

alors $(t\rho\sigma, s\rho\sigma[v\sigma]_p)$ est une paire critique ordonnée *de $s = t$ et $u = v$. On note $CP^\succ(E)$ l'ensemble des paires critiques ordonnées de $E \cup E^{-1}$.*

On a encore un lemme des paires critiques adapté :

Théorème 9.13 *La relation $\to_{E\succ}$ est localement confluente* sur les termes clos si et seulement si toutes les paires de $CP^\succ(E)$ sont localement confluentes.

La démonstration est une adaptation sans difficultés de la preuve usuelle.

9.4.1 Les règles de la complétion ordonnée

Les règles sont adaptées de la complétion usuelle, avec le problème suivant : il n'est pas toujours possible de décider l'existence d'une substitution close qui satisfasse les conditions de l'ordre. La solution consiste à employer une sur-approximation correcte $ACP^\succ(E)$ (*i.e.* telle que toutes les paires sont bien égales dans $\leftrightarrow_E^*$) de $CP^\succ(E)$), par exemple $CP(E \cup E^{-1})$, ou

$$\{(t\rho\sigma, s\rho\sigma[v\sigma]_p) \mid s = t, u = v \in E \cup E^{-1} \wedge s\rho\sigma \not\succ t\rho\sigma \wedge s\rho\sigma[u\sigma]_p \not\succ s\rho\sigma[v\sigma]_p\}$$

Trivial $\dfrac{E \cup \{s = s\}}{E}$

Paire Critique $\dfrac{E}{E \cup \{s = t\}}$ si $s = t$ est une paire critique de $ACP^{\succ}(E)$

Définition 9.14 (Coût d'une preuve ordonnée) *Le coût d'une étape de preuve est un triplet défini comme suit :*

- *Le coût de $s \leftrightarrow_{u=v} t$ est $(\{s\}, u, t)$ si $s \succ t$.*
- *Le coût de $s \leftrightarrow_{u=v} t$ est $(\{t\}, v, s)$ si $t \succ s$.*
- *Le coût de $s \leftrightarrow_{u=v} t$ est $(\{s, t\}, \_, \_)$ sinon.*

Le coût total d'une preuve est le multi-ensemble de ses coûts élémentaires.

Les coûts sont comparés comme dans le cas standard. En se basant sur la preuve de complétude de la complétion usuelle, on peut ajouter toutes les règles d'inférences qui détruisent ou modifient des équations pourvu que les preuves utilisant ces équations puissent encore se faire avec les nouvelles équations pour un coût moindre. Par exemple :

Trivial $\dfrac{E \cup \{s =_n s\}}{E}$

Simplifie $\dfrac{E \cup \{s = t\}}{E \cup \{s' = t'\}}$ si $s \rightarrow_{E\succ} s'$ et/ou $t \rightarrow_{E\succ} t'$

Lemme 9.15 *Soit $E_0 \vdash E_1 \vdash \ldots E_n \vdash E_{n+1} \ldots$ une exécution de la complétion équitable ($CP^{\succ}(E_\omega) \subset E_\infty$). Soit P une preuve close dans E_∞ qui n'est pas une preuve par récriture dans $\rightarrow_{E_\omega\succ}$. Si $\succ$ est un ordre total sur les termes clos, il existe une preuve close P' de E_∞, équivalente à P et de coût strictement inférieur.*

Théorème 9.16 *Soit E_ω l'ensemble des équations persistantes d'une dérivation de complétion ordonnée équitable. Si $\succ$ est un ordre total sur les termes clos, E_ω est convergent sur les termes clos.*

9.5 Complétion modulo

Il est possible d'adapter la complétion au cas de la récriture étendue, en modifiant encore une fois le calcul des paires critiques.

Chapitre 10

Clôture par congruence

Pour semi-décider le problème du mot, on peut utiliser la complétion ordonnée, mais des procédures de décision ont été développées, comme la clôture par congruence, et ses extensions modulo, utilisées essentiellement dans les prouveurs SMT (satisfiability modulo theories). Évidement, il y a un prix à payer pour passer de la semi-décision à la décision. Ici, la théorie modulo laquelle on raisonne est close, ou dans certaines des extensions modulo des théories prédéfinies possède des propriétés qui les contraignent fortement.

10.1 Clôture par congruence standard

Le problème qui se pose est, étant donnée une algèbre de termes, un ensemble fini d'équations entre termes clos E_0 et deux termes clos s et t, de savoir si

$$E_0 \vdash s = t$$

La clôture par congruence standard travaille sur un quadruplet $(\Theta, \Gamma, \Delta, \Phi)$ où

- Θ est l'ensemble des termes déjà rencontrés,
- Γ est un dictionnaire qui à un terme associe l'ensemble de ses sur-termes directs qui apparaîssent dans Θ,
- Δ est une structure d'« union-find » qui à chaque terme de Θ associe son représentant modulo l'ensemble d'équations déjà traitées (éventuellement lui-même)
- Φ est l'ensemble des équations qui restent à traiter, et la requête $s \stackrel{?}{=} t$, supposée traitée en dernier.

Elle est définie par un ensemble de règles d'inférences :

$$\textsc{Congr}\ \frac{\langle\ \Theta\ \mid\ \Gamma\ \mid\ \Delta\ \mid\ a=b\,;\,\Phi\ \rangle}{\langle\ \Theta\ \mid\ \Gamma\uplus\Gamma'\ \mid\ \Delta'\ \mid\ \Phi';\,\Phi\ \rangle}\ a,b\in\Theta,\ \ \Delta[a]\not\equiv\Delta[b]$$

où,

$(p,P)=(\Delta[a],\Delta[b])$

$\Gamma'(P)=\Gamma(P)\cup\Gamma(p)$

$\Delta'(r):=P$ si $\Delta(r)=p$

$\Delta'(r):=\Delta(r)$ si $\Delta(r)\neq p$

$$\Phi'=\left\{f(\vec{u})=f(\vec{v})\ \middle|\ \begin{array}{l}\Delta'[\vec{u}]\equiv\Delta'[\vec{v}],\ \ f(\vec{u})\in\Gamma(p)\\ f(\vec{v})\in\Gamma(p)\cup\Gamma(P)\end{array}\right\}$$

$$\textsc{Remove}\ \frac{\langle\ \Theta\ \mid\ \Gamma\ \mid\ \Delta\ \mid\ a=b\,;\,\Phi\ \rangle}{\langle\ \Theta\ \mid\ \Gamma\ \mid\ \Delta\ \mid\ \Phi\ \rangle}\ a,b\in\Theta,\ \ \Delta[a]\equiv\Delta[b]$$

$$\textsc{Add}\ \frac{\langle\ \Theta\ \mid\ \Gamma\ \mid\ \Delta\ \mid\ C[f(\vec{a})]\,;\,\Phi\ \rangle}{\langle\ \Theta\cup\{f(\vec{a})\}\ \mid\ \Gamma\uplus\Gamma'\ \mid\ \Delta\ \mid\ \Phi';\,C[f(\vec{a})]\,;\,\Phi\ \rangle}\ \left\{\begin{array}{l}f(\vec{a})\notin\Theta\\ \forall v\in\vec{a},v\in\Theta\end{array}\right.$$

où $C[f(\vec{a})]$ est une équation ou une requête contenant $f(\vec{a})$

$$\text{avec}\ \left\{\begin{array}{l}\Gamma'=\ \displaystyle\bigcup_{l\in\Delta[\vec{a}]}\ l\mapsto\Gamma(l)\cup\{f(\vec{a})\}\\ \Phi'=\ \left\{f(\vec{a})=f(\vec{b})\ \middle|\ \Delta[\vec{a}]\equiv\Delta[\vec{b}],\ \ f(\vec{b})\in\displaystyle\bigcap_{l\in\Delta[\vec{a}]}\Gamma(l)\right\}\end{array}\right.$$

$$\textsc{Query}\ \frac{\langle\ \Theta\ \mid\ \Gamma\ \mid\ \Delta\ \mid\ a\stackrel{?}{=}b\ \rangle}{\top}\ a,b\in\Theta,\ \ \Delta[a]\equiv\Delta[b]$$

$$\textsc{Fail}\ \frac{\langle\ \Theta\ \mid\ \Gamma\ \mid\ \Delta\ \mid\ a\stackrel{?}{=}b\ \rangle}{\bot}\ a,b\in\Theta,\ \ \Delta[a]\not\equiv\Delta[b]$$

La configuration initiale est égale à $\langle\ \emptyset\ \mid\ \emptyset\ \mid\ \mathbf{i}d\ \mid\ E_0;s\stackrel{?}{=}t\ \rangle$.

Ces règles sont évidemment correctes car toutes les équations ajoutées à la dernière composante sont valides (modulo E_0) car $=_{E_0}$ est une congruence. La terminaison est évidente car aucun nouveau terme n'est créé, et on ne peut engendrer qu'un nombre d'équations quadratique en la taille de l'entrée.

La complétude, c'est-à-dire que si FAIL s'applique, alors s et t ne sont pas égaux modulo E_0 se démontre en contruisant un modèle de E_0 où s et t sont distincts.

10.2 Clôture par congruence modulo X : $CC(X)$

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. Automatically proving termination where simplification orderings fail. In Michel Bidoit and Max Dauchet, editors, *Theory and Practice of Software Development*, volume 1214 of *Lecture Notes in Computer Science*, Lille, France, April 1997. Springer.

[3] Thomas Arts and Jürgen Giesl. Termination of term rewriting using dependency pairs. *Theoretical Computer Science*, 236 :133–178, 2000.

[4] Leo Bachmair and Nachum Dershowitz. Commutation, transformation, and termination. In Jörg H. Siekmann, editor, *Proc. 8th Int. Conf. on Automated Deduction, Oxford, England, LNCS 230*, pages 5–20, July 1986.

[5] Leo Bachmair and David A. Plaisted. Termination orderings for associative-commutative rewriting systems. *Journal of Symbolic Computation*, 1(4) :329–349, December 1985.

[6] Ahlem Ben Cherifa and Pierre Lescanne. An actual implementation of a procedure that mechanically proves termination of rewriting systems based on inequalities between polynomial interpretations. In Jörg H. Siekmann, editor, *8th International Conference on Automated Deduction*, volume 230 of *Lecture Notes in Computer Science*, pages 42–51, Oxford, England, 1986. Springer.

[7] Alexandre Boudet. Combining unification algorithms. *Journal of Symbolic Computation*, 16 :597–626, 1993.

[8] 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.

[9] Jim Christian and Patrick Lincoln. Adventures in associative-commutative unification. Technical Report ACA-ST-275-87, MCC, AI Program Austin, Austin, October 1987.

[10] M. Clausen and A. Fortenbacher. Efficient solution of linear diophantine equations. *Journal of Symbolic Computation*, 8(1&2) :201–216, 1989.

[11] 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.

[12] Evelyne Contejean and Hervé Devie. An efficient algorithm for solving systems of diophantine equations. *Information and Computation*, 113(1) :143–172, August 1994.

[13] 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.

[14] Catherine Delor and Laurence Puel. Extension of the associative path ordering to a chain of associative-commutative symbols. In Claude Kirchner, editor, *5th International Conference on Rewriting Techniques and Applications*, volume 690 of *Lecture Notes in Computer Science*, pages 389–404, Montreal, Canada, June 1993. Springer.

[15] Nachum Dershowitz. Termination of rewriting. *Journal of Symbolic Computation*, 3(1) :69–115, February 1987.

[16] Eric Domenjoud. AC unification through order-sorted AC1 unification. *Journal of Symbolic Computation*, 14(6) :537–556, December 1992.

[17] François Fages. Associative-commutative unification. *Journal of Symbolic Computation*, 3(3), June 1987.

[18] A. Fortenbacher. An algebraic approach to unification under associativity and commutativity. In *Proc. Rewriting Techniques and Applications 85, Dijon, LNCS 202*. Springer, May 1985.

[19] Jürgen Giesl and Deepak Kapur. Dependency pairs for equational rewriting. In Aart Middeldorp, editor, *12th International Conference on Rewriting Techniques and Applications*, volume 2051 of *Lecture Notes in Computer Science*, pages 93–108, Utrecht, The Netherlands, May 2001. Springer.

[20] Bernhard Gramlich. Abstract relations between restricted termination and confluence properties of rewrite systems. *Fundamenta Informaticæ*, 24 :3–23, 1995.

[21] Bernhard Gramlich. On proving termination by innermost termination. In Harald Ganzinger, editor, *7th International Conference on Rewriting Techniques and Applications*, volume 1103 of *Lecture Notes in Computer Science*, pages 93–107, New Brunswick, NJ, USA, July 1996. Springer.

[22] Alexander Herold and Jörg H. Siekmann. Unification in abelian semi-groups. *Journal of Automated Reasoning*, 3(3) :247–283, 1987.

[23] Nao Hirokawa and Aart Middeldorp. Dependency Pairs Revisited. In Vincent van Oostrom, editor, *15th International Conference on Rewriting Techniques and Applications*, volume 3091 of *Lecture Notes in Computer Science*, pages 249–268, Aachen, Germany, June 2004. Springer.

[24] Gérard Huet. An algorithm to generate the basis of solutions to homogeneous linear diophantine equations. *Information Processing Letters*, 7(3), April 1978.

[25] Gérard Huet and Dallas S. Lankford. On the uniform halting problem for term rewriting systems. Research Report 283, INRIA, March 1978.

[26] 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.

[27] Claude Kirchner. Méthodes et outils de conception systématique d'algorithmes d'unification dans les théories equationnelles. Thèse d'Etat, Univ. Nancy, France, 1985.

[28] 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.

[29] Keiichirou Kusakari, Claude Marché, and Xavier Urbain. Termination of associative-commutative rewriting using dependency pairs criteria. Research Report 1304, LRI, 2002. `http://www.lri.fr/~urbain/textes/rr1304.ps.gz`.

[30] Keiichirou Kusakari, Masaki Nakamura, and Yoshihito Toyama. Argument filtering transformation. In Gopalan Nadathur, editor, *Principles and Practice of Declarative Programming, International Conference PPDP'99*, volume 1702 of *Lecture Notes in Computer Science*, pages 47–61, Paris, 1999. Springer.

[31] Keiichirou Kusakari and Yoshihito Toyama. On proving AC-termination by AC-dependency pairs. *IEICE Transactions on Information and Systems*, E84-D(5) :604–612, 2001.

[32] 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.

[33] 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.

[34] M. Livesey and Jörg H. Siekmann. Unification of bags and sets. Research report, Institut fur Informatik I, Universität Karlsruhe, West Germany, 1976.

[35] Claude Marché and Xavier Urbain. Modular and incremental proofs of AC-termination. *Journal of Symbolic Computation*, 38 :873–897, 2004.

[36] M. H. A. Newman. On theories with a combinatorial definition of 'equivalence'. *Ann. Math.*, 43(2) :223–243, 1942.

[37] Enno Ohlebusch. On the modularity of termination of term rewriting systems. *Theoretical Computer Science*, 136 :333–360, 1994.

[38] 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.

[39] Albert Rubio. A fully syntactic AC-RPO. In Paliath Narendran and Michael Rusinowitch, editors, *10th International Conference on Rewriting Techniques and Applications*, volume 1631 of *Lecture Notes in Computer Science*, Trento, Italy, July 1999. Springer.

[40] Mark 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.

[41] Mark E. Stickel. A unification algorithm for associative-commutative functions. *Journal of the ACM*, 28(3) :423–434, 1981.

[42] Xavier Urbain. Modular and incremental automated termination proofs. *Journal of Automated Reasoning*, 32 :315–355, 2004.

Index