Réécriture

Terminaison des systèmes de réécriture

Indécidabilité de la terminaison

1 Problème de correspondance de Post

Définitions

- Un ensemble $(\alpha_1, \beta_1), ..., (\alpha_n, \beta_n)$
  où $\alpha_i \in A^*$ et $\beta_i \in A^*$ est appelé un problème de correspondance de Post.
- Un match est formé
  - d'un mot $w \in A^+$
  - et d'une suite $(i_1, ..., i_p) \in [1..n]^+$

  tels que $w = \alpha_{i_1}...\alpha_{i_p} = \beta_{i_1}...\beta_{i_p}$.

Exemple

1. (ala, tour)
2. (aman, dela)
3. (dela, rène)
4. (gal, galaman)
5. (magnanime, anime)
6. (rène, ala)
7. (tour, magn)

Une match est galamandelarènealatourmagnanime avec la suite $(4, 2, 3, 6, 1, 7, 5)$.

> Gal, amant de la reine, alla, tour magnanime,
> Gallament de l'arène, à la tour Magne, à Nimes.
>
> Victor Hugo

Mais là c'est super-facile.

2 Des exemples

Les problèmes de correspondance ci-dessus ont-ils un match ?

i	α_i	β_i
1	101	10
2	11	011
3	011	101

1

i	α_i	β_i
1	010	101
2	00	000
3	101	10

2

Pour le premier problème, il n'y a pas de match (preuve en commençant par la fin), le deuxième a un match : 3-1-2.

RÉSULTAT
Le problème de correspondance de Post (ou PCP) est indécidable,
à partir de deux éléments dans A.
Autrement dit, il n'y a pas d'algorithme avec

- entrée : un PCP sur deux lettres
- sortie : le problème a un match ou le problème n'a pas de match .

3 Réduction de la terminaison à PCP

On considère la signature

- $\Sigma_0 = \{\#\}$,
- $\Sigma_1 = A$,
- $\Sigma_2 = \emptyset$,
- $\Sigma_3 = \{f\}$,
- $\Sigma_n = \emptyset$ pour $n \geq 4$.

A chaque mot $a_1a_2...a_n$ de A^* on peut associer un terme
dit monadique

$$(a_1, (a_2(...(a_n(\#))...))) \in T(\Sigma).$$

que l'on note $a_1a_2...a_n$!
Dans la suite $a_1a_2...a_nx$ pour $x \in X$ est

$$(a_1, (a_2(...(a_n(x))...))) \in T(\Sigma).$$

Étant donné un problème de correspondance de Post sur A.

On considère le système de réécriture : $R_1 = \left\{ \begin{array}{ccc} f(\alpha_1 x, \beta_1 y, z) & \longrightarrow & f(x, y, z) \\ \vdots & & \\ f(\alpha_n x, \beta_n y, z) & \longrightarrow & f(x, y, z) \end{array} \right.$

$R_2 = \left\{ \begin{array}{ccc} f(\#, \#, a_1(y)) & \longrightarrow & f(a_1(y), a_1(y), a_1(y)) \\ \vdots & & \\ f(\#, \#, a_m(y)) & \longrightarrow & f(a_m(y), a_m(y), a_m(y)) \end{array} \right.$

$\xrightarrow[R_1 \cup R_2]{}$ ne termine pas si et seulement si PCP a un match.

Démonstration

Si PCP a un match w, alors $f(\#, \#, w) \xrightarrow[R_2]{} f(w, w, w) \xrightarrow[R_1]{+} f(\#, \#, w)$.

Si $\xrightarrow[R_1 \cup R_2]{}$ ne termine pas, alors elle passe une infinité de fois par des règles $\xrightarrow[R_2]{}$. Et cela implique qu'il y a un match.

En effet, les règles $\xrightarrow[R_1]{}$ décroissent la taille des termes et ne peuvent pas contribuer seules à la non terminaison.

Remarque

On a d'ailleurs montré plus fort : $\xrightarrow[R_1 \cup R_2]{}$ est cyclique si et seulement si PCP a un match.

Donc l'acyclicité et la terminaison sont indécidables.

4 Ordre de réduction

Définition

- Un ordre de réécriture est
  - compatible si $f(t_1, ..., t_{i-1}, s, t_{i+1}, ..., t_n) > f(t_1, ..., t_{i-1}, s', t_{i+1}, ..., t_n)$
  - clos par substitution si $s_1 > s_2 \quad \Rightarrow \quad \sigma(s_1) > \sigma(s_2)$.
- Un ordre de réduction est un ordre de réécriture noethérien.

N.B. La taille du terme n'est pas un ordre de réduction.

Un système de réécriture R termine si et seulement si il existe un ordre de réduction $>$ tel que $l > r$ pour tout $l \longrightarrow r \in R$.

Démonstration

- Si : $>$ est une relation noethérienne qui contient $\xrightarrow{+}$. Donc $\xrightarrow{+}$ elle-même est noethérienne.
- Seulement si : $\xrightarrow{+}$ est une ordre de réduction qui satisfait clairement la condition.

Interprétations

5 Algèbre ordonnée et ordre induit

Une algèbre ordonnée est la donnée d'une algèbre $\mathcal{A}$ et d'un ordre $>$ sur A.

L'ordre $>_{\mathcal{A}}$ sur $T(\Sigma, V)$ (ordre sur les termes) induit par cette algèbre est défini par :

$$(\forall s, t \in T(\Sigma, V))\ s >_{\mathcal{A}} t \quad \text{ssi} \quad \pi(s) > \pi(t) \quad \text{pour tout morphisme } \pi.$$

6 Monotonie

Une fonction $F : A^n \to A$ est monotone (ou croissante en chacune de ses variables si

$b > c \quad \Rightarrow \quad F(a_1, ..., a_{i-1}, b, a_{i+1}, ..., a_n) > F(a_1, ..., a_{i-1}, c, a_{i+1}, ..., a_n)$

Soit $\mathcal{A}$ une algèbre ordonnée, munie d'un ordre noethérien. Si toutes les interprétations $f^{\mathcal{A}}$ sont monotones, alors $>_{\mathcal{A}}$ est un ordre de réduction.

7 Interprétations polynomiales

$A = \mathbb{N}_\star$

Les $f_n^{\mathcal{A}} \in \Sigma_n^{\mathcal{A}}$ sont des polynômes à n variables de $\mathbb{N}[\![X_1, ..., X_n]\!]$.

Un polynôme est monotone s'il dépend de toutes ses indéterminées, c-à-d que pour chaque i tel que $1 \leq i \leq n$, il contient un monôme avec une occurrence de X_i.

Une interprétation polynomiale monotone est une interprétation dans laquelle toute les interprétations sont des polynômes monotones.

L'ordre induit est un ordre de réduction, appelé ordre polynomial.

8 Exercice

Prouver la terminaison des systemes suivants.

$$\begin{array}{rcll} (x * y) * z & \longrightarrow & x * (y * z) & \text{(A)} \\ f(x * y) & \longrightarrow & f(x) * f(y) & \text{(E)} \end{array}$$

$$\begin{array}{rcll} (x * y) * z & \longrightarrow & x * (y * z) & \text{(A)} \\ f(x) * f(y) & \longrightarrow & f(x * y) & \text{(E)} \\ f(x) * (f(y) * z) & \longrightarrow & f(x * y) * z & \text{(EA)} \end{array}$$

9 Les limites des ordres polynômiaux

9.1 Les limites des ordres polynômiaux I

Prouver qu'un ensemble $(F_i(X_1, ..., X_n) > G_i(X_1, ..., X_n))_{1 \leq i \leq m}$ d'inégalités polynomiales est satisfaite revient à prouver qu'un ensemble de polynôme est positif pour toute valeur dans $\mathbb{N}$. En se ramenant au 10^{eme} problème de Hilbert, on montre que c'est indécidable

9.2 10^{eme} problème de Hilbert.

Dans son 10^{eme} problème (1900) Hilbert demandait un algorithme pour déterminer si les systèmes d'équations diophantiennes linéaires ont une solution ou non,

Un système d'équations diophantiennes est une suite $(P_1, ..., P_m)$ où les $P_i \in \mathbf{Z}[\![X_1, ..., X_n]\!]$ sont des polynômes à coefficients entiers relatifs.

Une solution est un n-uplet $(a_1, ...a_n) \in \mathbf{Z}^n$ tel pour chaque $1 \leq i \leq n$, on a $P_i(a_1, ..., a_n) = 0$.

Youri Matijasevič (1970) a démontré que ce problème est indécidable.

9.3 Indécidabilité de la positivité

Le problème de la positivité des systèmes de polynômes est de savoir si un système de $(Q_1, ..., Q_m)$ où les $Q_i \in \mathbf{Z}[\![X_1, ..., X_n]\!]$ est strictement positif pour tout $(a_1, ..., a_n) \in \mathbf{Z}^n$.

Si l'on a un algorithme pour la positivité des systèmes de polynômes, alors pour chaque système d'équations diophantiennes linéaires $(P_1, ..., P_m)$ où les $P_i \in \mathbf{Z}[\![X_1, ..., X_n]\!]$, on est capable de répondre si le système $(P_1^2, ..., P_m^2)$ est strictement positif ou non sur $\mathbf{Z}^n$, donc de décider s'il a une solution ou non $\mathbf{Z}^n$. On aurait donc un algorithme pour résoudre le dixième problème de Hilbert.

Question : Comment passe-t-on de la positivité sur $\mathbf{Z}^n$ à la positivité sur $\mathbb{N}^n$?

9.4 Prouver la positivité par ordinateur

Puisque c'est indécidable on peut se ramener à la décidabilité sur les réels qui est décidable, mais inefficaces (méthode d'élimination des quantificateurs de Tarski).

On peut utiliser des heuristiques. De toute façon la terminaison est indécidable !

10 Les limites des ordres polynômiaux II

La longueur maximale des réductions d'un système de réécriture dont la terminaison est prouvée par un ordre polynomial est doublement exponentielle.

Théorème

Si R est un système de réécriture dont la preuve de terminaison est faite par une interprétation polynomial monotone, alors il existe c telle que la longueur de toute dérivation à partir de t soit majorée par $2^{2^{c\|t\|}}$.

Remarques

- la longueur de t est en fait son nombre de connecteurs.
- On fait l'hypothèse que R n'a qu'un nombre fini de règles (on peut généraliser à un nombre infini de règles).
- Soit t un terme et a un entier quelconque pour interpréter le variable de t. Soit π_a une interprétation qui prend la valeur a sur chaque variable.

Soit $t \longrightarrow t_1 \cdots \longrightarrow t_m$ une dérivation partant de t et de longueur m. On a $\pi_a(t) > \pi_a(t_1) > \cdots > \pi_a(t_m)$ donc $\pi_a(t) \geq m$.
Il suffit donc de majorer $\pi_a(t)$.

Démonstration
Soit $c = k + \log_2(d)$ où pour chaque opérateur f apparaissant dans le système de réécriture

$$P_f(a_1, ..., a_n) \leq d \prod_{1 \leq i \leq n} a_i^k.$$

On procède par récurrence sur la structure de t.
Si t est une constante alors $\pi_a(t) \leq d \leq 2^{2^c}$.
Si $t = f(t_1, ..., t_n)$ alors

$$\begin{aligned}
\pi_a(f(t_1, ..., t_n)) &= P_f(\pi(t_1), ..., \pi(t_n)) \\
&\leq d \prod_{1 \leq i \leq n} \pi(t_i)^k \\
&\leq d \prod_{1 \leq i \leq n} 2^{2^{c.\|t_i\|}} = d.2^{\Sigma_{1 \leq i \leq n} 2^{c.\|t_i\|}} \\
&\leq d.2^{2^{c.\Sigma_{1 \leq i} \|t_i\|}} \\
&\leq 2^c.2^{2^{c.\Sigma_{1 \leq i \leq n} \|t_i\|}} = 2^{2^{c.(1+\Sigma_{1 \leq i \leq n} \|t_i\|)}} \\
&= 2^{2^{c\|t\|}}.
\end{aligned}$$

On a aussi le résultat : La borne est atteinte

Exercice : Trouver un système de réécriture tel que la longueur de la plus longue chaîne de dérivation soit doublement exponentielle.
Indication : Un systeme de réécriture

- qui "calcule" quelque chose comme une exponentielle (une puissance d'un entier par exemple),
- dont la preuve de terminaison est faite par une interprétation polynomiale,
- dans lequel il y un terme court qui se normalise en temps doublement exponentiel.

10.1 Les limites des ordres polynômiaux III

Théorème : :

Si un système de réécriture calcule une fonction entière (avec les deux constructeurs 0 et s) alors cette fonction est à croissance polynomiale.

Exercice : Le système

$$\begin{aligned}
add(0, m) &\longrightarrow m \\
add(s(n), m) &\longrightarrow s(add(n, m)) \\
binom(n, 0) &\longrightarrow 1 \\
binom(0, s(p)) &\longrightarrow 0 \\
binom(s(n), s(p)) &\longrightarrow add(binom(n, s(p)), binom(n, p))
\end{aligned}$$

ne peut être prouvé terminer par une interprétation polynomiale.

Remarque : Comment faut-il faire, si on veut le faire par une interprétation ?

Les beaux ordres

11 Les bons ordres

Définitions :

- Les bons ordres sont les ordres noetheriens totaux.
- Les bons ordres sont aussi appelés des ordinaux.
- En réécriture, nous sommes intéressés par des ordres partiels.

12 Ordre noethérien et incrémentalité

12.1 Pourquoi ?

Jusqu'à maintenant, on a montré qu'un ordre était noethérien parce qu'il était contenu dans un ordre qui était noethérien. Mais un informaticien veut construire les ordres noethériens incrémentalement, c-à-d en ajoutant de nouvelles paires à un ordre déjà connu. C'est exactement ce qui se passe dans une procédure de complétion. On ajoute des paires nouvelles que l'ordre que l'on a ne sait pas orienter.

Il faut donc agrandir l'ordre à la volée.

12.2 Ordre Incrémental, sous-suites

Définitions :

- Un ordre est incrémental si :
  1. il est noethérien,
  2. tout ordre qui le contient est noethérien.
- Une antichaîne est un ensemble tel que $x \leq y \quad \Rightarrow \quad x = y$.
- Un ordre incrémental doit être "beau" : il ne doit être :
  1. Ni trop grand : noethérien,

2. Ni trop gros : il n'y a pas d'antichaîne infinie. C-à-d que dans une antichaîne, tous les éléments sont tous deux à deux incomparables.

– Une suite $(x_i)_{i\in\mathbb{N}}$ dans laquelle il existe i et j tels que $i < j$ et $x_i \leq x_j$ est dite bonne. Si une suite n'est pas bonne, elle est mauvaise. (!!)

– Une sous-suite de $(x_i)_{i\in\mathbb{N}}$ est donnée par une application $\varphi : \mathbb{N} \to$ croissante, autrement dit la sous-suite est celle des $(x_{\varphi(i)})_{i\in\mathbb{N}}$

12.3 Résultats

PROPOSITION :
Les définitions suivantes sont équivalentes :

– l'ordre est incrémental

– l'ordre est noethérien et sans antichaîne infinie.

– toute suite est bonne,

– de toute suite $(x_i)_{i\in\mathbb{N}}$ on peut extraire une sous-suite croissante.

Démonstration
Elle se fait suivant le shéma suivant :

$$(1) \iff (2) \qquad (2) \implies (3)$$
$$(4) \implies (3) \qquad (3) \implies (4)$$
$$(3) \implies (2)$$

$(2) \implies (3)$
Supposons qu'il n'y a pas de suite finie strictement décroissante.
Soit une suite mauvaise.
On considère la suite extraite $(x_{\varphi(i)})_{i\in\mathbb{N}}$ ainsi

– $x_{\varphi(0)}$ est tel que pour $j > \varphi(0)$ on a $x_{\varphi(0)} \# x_j$. C'est possible, car la suite est mauvaise et il n'y a pas de suite infinie strictement décroissante.

– $x_{\varphi(k+1)}$ est tel que $\varphi(k) < \varphi(k+1)$ et pour $j > \varphi(k+1)$ on a $x_{\varphi(k+1)} \# x_j$.

Cette suite extraite $(x_{\varphi(i)})_{i\in\mathbb{N}}$ est une antichaîne infinie. Contradiction !

$(3) \implies (4)$
Soit une suite qui n'admet pas de sous-suite croissante.
Les sous-ensembles qui sont ordonnés par indices croissants et sont faiblement croissants pour $\leq$ sont tous finis.

– Ils sont en nombre infinis.

– Ils ont tous un dernier élément.

La suite (infinie) $(x_{\varphi(i)})_{i\in\mathbb{N}}$ de ces derniers éléments est bonne, donc il existe dans cette suite $i < j$ avec $x_{\varphi(i)} \leq x_{\varphi(j)}$.
En contradiction avec la supposition que $x_{\varphi(i)}$ est un dernier élément !

13 Beaux ordres

Les ordres qui satisfont les conditions du lemme précédent sont appelés aussi des beaux ordres.

FAIT : Le produit (composante par composante) de beaux ordres est un bel ordre.

Le produit d'ordre $(A_1, \leq_1) \times \cdots \times (A_n, \leq_n)$ est défini par

$$(a_1, ..., a_n) \leq_1 \times ... \times \leq_n (b_1, ..., b_n) \iff \bigwedge_{i=1}^{n} a_i \leq_i b_i$$

14 Le plongement

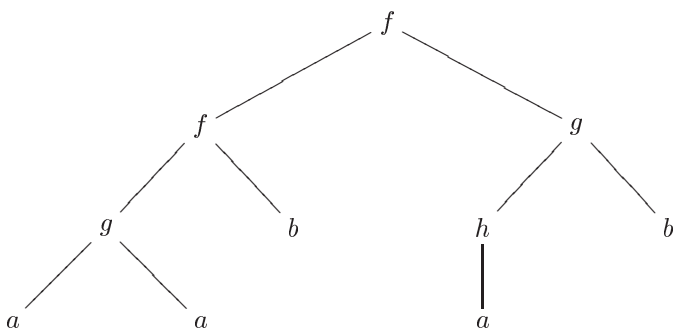
Considérons le système de réécriture $\mathcal{EMB}$qui contient pour chaque $f \in \Sigma_n$ et chaque $1 \leq i \leq n$ une règle,

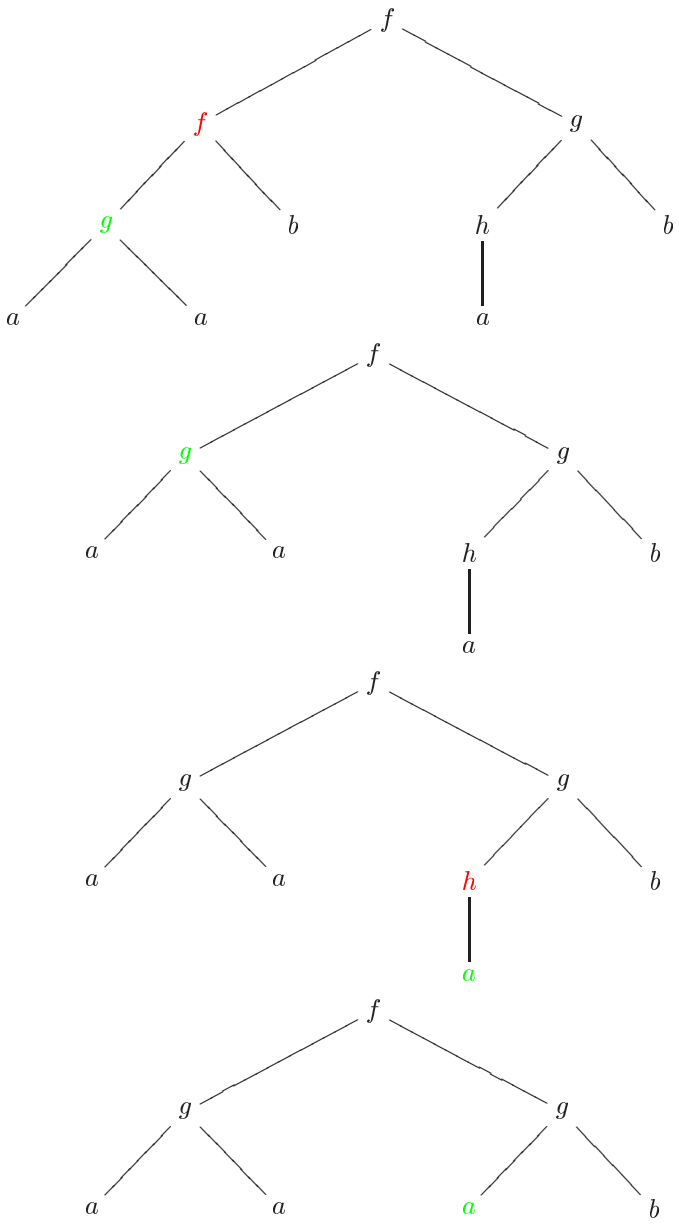
$$f(x_1, ..., x_n) \longrightarrow x_i$$

$\xrightarrow[\mathcal{EMB}]{+}$ est sans cycle.

La relation $\xrightarrow[\mathcal{EMB}]{+}$ est un ordre strict noetherien qui s'appelle le plongement.

Intuitivement, un terme t est plongé dans un terme t', si $t' \xrightarrow[\mathcal{EMB}]{+} t$, c-à-d si on passe de t' à t en effaçant des nœuds et en recollant un terme fils du nœud manquant à la place laissée libre.




f
f
g
g
b
h
b
a
a
a
f
g
g
a
a
h
b
a
f
g
g
a
a
h
b
a
f
g
g
a
a
a
b

14.1 Le plongement dans A^*

Les termes de A^* sont les termes monadiques.
Le plongement est défini par

- $(\forall \alpha \in A^+) \quad \alpha \xrightarrow[\mathcal{EMB}]{+} \epsilon,$
- $(\forall \alpha \in A^*)(\forall \beta \in A^*)(\forall a \in A)\ \alpha \xrightarrow[\mathcal{EMB}]{+} \beta \Rightarrow a\alpha \xrightarrow[\mathcal{EMB}]{+} a\beta$
- $(\forall \alpha \in A^*)(\forall \beta \in A^*)(\forall a \in A)\ \alpha \xrightarrow[\mathcal{EMB}]{+} \beta \Rightarrow a\alpha \xrightarrow[\mathcal{EMB}]{+} \beta$

14.2 Théorème de Higman

Théorème :

Le plongement sur A^* est un bel ordre.

Remarque : Une suite qui contient ϵ est bonne car ϵ est plongé dans tous les termes.

Démonstration

Supposons que $(A^*, \xrightarrow[\mathcal{EMB}]{+})$ n'est pas un bel ordre.

Soit un mauvaise suite $(\alpha_i)_{i\in\mathbb{N}}$ "minimale" au sens suivant :

1. α_0 est un des plus petits éléments qui commence une mauvaise suite.
2. α_i est le plus petit élément en $i$ème position d'une mauvaise suite qui commence par $\alpha_0, ...\alpha_{i-1}$.

Cette suite ne contient aucun ϵ. Donc tous les termes sont de la forme $a_i\alpha'_i$.
On peut extraire une sous-suite $a\alpha'_{\varphi(i)}$, pour le même a (car l'alphabet de travail est fini).

La suite $\alpha_0, ...\alpha_{\varphi(0)-1}, \alpha'_{\varphi(0)}, ..., \alpha'_{\varphi(j)}...$ est bonne pour l'ordre $\xrightarrow[\mathcal{EMB}]{+}$.

Donc

1. soit $\alpha_k \xleftarrow[\mathcal{EMB}]{*} \alpha'_{\varphi(j)}$ pour $0 \leq k < \varphi(0)$ et $0 \leq j$,

   donc $\alpha_k \xleftarrow[\mathcal{EMB}]{*} a\alpha'_{\varphi(j)}$, contradiction !

2. soit $\alpha'_{\varphi(j)} \xleftarrow[\mathcal{EMB}]{*} \alpha'_{\varphi(j')}$ pour $0 \leq j < j'$,

   donc $a\alpha'_{\varphi(j)} \xleftarrow[\mathcal{EMB}]{*} a\alpha'_{\varphi(j')}$, contradiction !

14.3 Théorème de Higman généralisé

On a besoin du théorème de Higman sur un alphabet infini dénombrable.
Comment le généraliser ?

- Il faut ordonner A par un bel ordre,
- Il faut généraliser l'énoncé.

15 Plongement avec restriction

15.1 Définitions

Un ordre sur Σ est appelé une précédence.

On se donne une précédence $\geq$ et un système de réécriture appelé restriction tel que pour chaque f et chaque g avec $f \geq g$ et chaque permutation $\pi : [1..n] \longrightarrow [1..n]$

$$f(x_1, ..., x_n) \longrightarrow g(x_{\pi(1)}, ..., x_{\pi(p)})$$

15.2 Plongement avec restriction sur A^*

- $(\forall \alpha \in A^+) \quad \alpha \xrightarrow[\mathcal{EMB}]{+} \epsilon,$
- $(\forall \alpha \in A^*)(\forall \beta \in A^*)(\forall a \in A)$
  $a \geq b \ \& \ \alpha \xrightarrow[\mathcal{EMB}]{+} \beta \quad \Rightarrow \quad a\alpha \xrightarrow[\mathcal{EMB}]{+} b\beta$
- $(\forall \alpha \in A^*)(\forall \beta \in A^*)(\forall a \in A)\ \alpha \xrightarrow[\mathcal{EMB}]{+} \beta \quad \Rightarrow \quad a\alpha \xrightarrow[\mathcal{EMB}]{+} \beta$

16 Théorème de Higman généralisé

Théorème :

Si $\geq$ est un bel ordre sur A le plongement est un bel ordre sur A^*.

Démonstration

Là où on a dit

"On peut extraire une sous-suite $a\alpha'_{\varphi(i)}$, pour le même a.", on dit

"On peut extraire une sous-suite $a_{\varphi(i)}\alpha'_{\varphi(i)}$ telle que la suite $a_{\varphi(i)}$ soit croissante."

Le reste de la démonstration est identique.

17 Théorème de Kruskal

Théorème :

Si A est fini, le plongement est un bel ordre sur $T(\Sigma, X)$.

Démonstration Supposons que le plongement n'est pas un bel ordre.

Considérons une mauvaise suite minimum, construite comme dans le théorème de Higman. (Exercice : refaire la construction.)

De cette suite on peut extraire une sous-suite de la forme $(f(s_1^i, ..., s_n^i))_{i\geq 0}$.

Les n suites $(s_j^i)_{i\geq 0}$ ainsi que leurs sous-suites sont bonnes.

De $(s_1^i)_{i\geq 0}$ on peut extraire une sous-suite croissante $(s_1^{\varphi_1(i)})_{i\geq 0}$.

De $(s_2^{\varphi_1(i)})_{i\geq 0}$ on peut extraire une sous-suite croissante $(s_2^{\varphi_2\varphi_1(i)})_{i\geq 0}$.

$\vdots$

De $(s_n^{\varphi_{n-1}\ldots\varphi_1(i)})_{i\geq 0}$ on peut extraire une sous-suite croissante $(s_n^{\varphi_n\ldots\varphi_1(i)})_{i\geq 0}$.

Clairement la suite $(f(s_1^{\varphi_{n-1}\ldots\varphi_1(i)},\ldots,s_n^{\varphi_{n-1}\ldots\varphi_1(i)}))_{i\geq 0}$ est croissante. Contradiction !

Les ordres de simplification

18 Ordre de simplification

Un ordre de simplification est un ordre de réécriture $>$ qui satisfait la propriété de sous-terme, c-à-d :

$$(\forall p \in Pos(t)) \qquad t > t_{|p}$$

Combiné avec la compatibilité, les ordres de simplification contiennent le plongement. Donc

- les ordres de simplification sont des beaux ordres,
- les ordres de simplification terminent,
- les ordres de simplification sont des ordres de réduction.

19 L'ordre lexicographique sur les chemins

Supposons donné un préordre $\leq$ sur Σ appelé précédence.

$s <_{lpo} t$ ssi

(A) s est une variable x, t n'est pas une variable et $x \in Var(t)$

ou

(B1) $s \equiv f(s_1,\ldots,s_m)$ et $t \equiv g(t_1,\ldots,t_n)$ et

(B2) $\forall i \in [1..m]\ s_i <_{lpo} t$

(B21) $f < g$ ou

(B22) $f \sim g$ et $(s_1,\ldots,s_m) <_{lpo}^{lex} (t_1,\ldots,t_n)$ ou

(B23) $\exists j \in [1..n] s \leq_{lpo} t_j$

20 Exercices

20.1 Ackermann

Prouver la terminaison de la fonction d'Ackermann

$$\begin{aligned} Ack(0,n) &\longrightarrow s(n) \\ Ack(S(m),0) &\longrightarrow Ack(m,s(0)) \\ Ack(s(m),s(n)) &\longrightarrow Ack(m,Ack(s(m),n)) \end{aligned}$$

20.2 (E) + (A)

Premier cas :

$$\begin{array}{rcll} (x*y)*z & \longrightarrow & x*(y*z) & (\mathrm{A}) \\ f(x*y) & \longrightarrow & f(x)*f(y) & (\mathrm{E}) \end{array}$$

Deuxième cas :

$$\begin{array}{rcll} (x*y)*z & \longrightarrow & x*(y*z) & (\mathrm{A}) \\ f(x)*f(y) & \longrightarrow & f(x*y) & (\mathrm{E}) \\ f(x*y)*z & \longrightarrow & f(x)*(f(y)*z) & (\mathrm{EA}) \end{array}$$

Troisième cas :

$$\begin{array}{rcll} (x*y)*z & \longrightarrow & x*(y*z) & (\mathrm{A}) \\ f(x)*f(y) & \longrightarrow & f(x*y) & (\mathrm{E}) \\ f(x)*(f(y)*z) & \longrightarrow & f(x*y)*z & (\mathrm{EA}) \end{array}$$

20.3 Les groupes

$$\begin{array}{rcl} x*e & \longrightarrow & x \\ e*x & \longrightarrow & x \\ x*i(x) & \longrightarrow & e \\ i(x)*x & \longrightarrow & e \\ i(e) & \longrightarrow & e \\ i(i(x)) & \longrightarrow & x \\ i(x*y) & \longrightarrow & i(y)*i(x) \\ (x*y)*z & \longrightarrow & x*(y*z) \\ x*(i(x)*y) & \longrightarrow & y \\ i(x)*(x*y) & \longrightarrow & y \end{array}$$

21 Le statut des operateurs

On peut choisir de prendre l'ordre lexicographique de gauche à droite, ou de droite à gauche.

22 Exercices

22.1 Les groupes (variante avec division)

$$
\begin{array}{rcl}
i(e) & \longrightarrow & e \\
x/e & \longrightarrow & x \\
e/x & \longrightarrow & i(x) \\
x/x & \longrightarrow & e \\
i(x/y) & \longrightarrow & y/x \\
i(i(x)) & \longrightarrow & x \\
x/(y/z) & \longrightarrow & (x/i(z))/y \\
(x/i(y))/y & \longrightarrow & x \\
(x/y)/i(y) & \longrightarrow & x
\end{array}
$$

22.2 Exercice d'école

$$f(f(x)) \longrightarrow f(g(f(x)))$$

22.3 Factorielle

$$
\begin{array}{rcl}
fact(s(x)) & \longrightarrow & fact(p(s(x))) \\
p(s(0)) & \longrightarrow & 0 \\
p(s(s(x))) & \longrightarrow & s(x)
\end{array}
$$

22.4 Pgcd

$$
\begin{array}{rcl}
if(s(m), s(n), p, q) & \rightarrow & if(m, n, p, q) \\
if(s(m), 0, p, q) & \rightarrow & p \\
if(0, n, p, q) & \rightarrow & q \\
 & & \\
s(m) - s(n) & \rightarrow & m - n \\
s(m) - 0 & \rightarrow & m \\
0 - s(n) & \rightarrow & 0 \\
 & & \\
pgcd(m, 0) & \rightarrow & m \\
pgcd(0, m) & \rightarrow & m \\
pgcd(s(m), s(n)) & \rightarrow & if(m, n, pgcd(m - n, s(n)), pgcd(s(m), n - m))
\end{array}
$$

23 L'ordre lexicographique sur les chemins

Résultat

$<_{lpo}$ est un ordre.

1. Antisymétrie ;
2. Transitivité.