Logique équationnelle et réécriture

Plan

Plan

Plan

Plan

Plan

Plan

Bibliographie

- Franz Baader & Tobias Nipkow, *Term Rewriting and* All that

Spécifications équationnelles

- Définition d'une structure (mathématique ou informatique) à l'aide d'égalités (équations)
- En déduire d'autres égalités
- Calculer avec des égalités

Exemple : addition des entiers naturels

$$\begin{array}{rcl} x+0 & \approx & x \\ x+S(y) & \approx & S(x+y) \end{array}$$

$$S(0)+S(S(0)) \approx S(S(0)+S(0)) \approx S(S(S(0))+0) \approx S(S(S(0)))$$

Spécifications équationnelles

- Définition d'une structure de données

$$\begin{aligned}
&\mathrm{pop}(\mathrm{empty}) \approx \mathrm{empty}\\
&\mathrm{pop}(\mathrm{add}(x,y)) \approx y\\
&\mathrm{top}(\mathrm{add}(x,y)) \approx x\\
&\mathrm{dup}(\mathrm{add}(x,y)) \approx \mathrm{add}(x,\mathrm{add}(x,y))\\
&\mathrm{pop}(\mathrm{dup}(x)) \approx x\\
&\mathrm{size}(\mathrm{empty}) \approx 0\\
&\mathrm{size}(\mathrm{add}(x,y)) \approx 1+\mathrm{size}(y)\\
&\mathrm{size}(\mathrm{dup}(x)) \approx 1+\mathrm{size}(x)
\end{aligned}$$

- Égalités = règles d'évaluation dans les preuves

$$\mathrm{add}(a,\mathrm{pop}(\mathrm{add}(b,\mathrm{pop}(\mathrm{dup}(\mathrm{add}(a,\mathrm{empty})))))))$$

Spécifications équationnelles

- Définition d'une structure de données

$$\begin{aligned}
&\mathrm{pop}(\mathrm{empty}) \approx \mathrm{empty}\\
&\mathrm{pop}(\mathrm{add}(x,y)) \approx y\\
&\mathrm{top}(\mathrm{add}(x,y)) \approx x\\
&\mathrm{dup}(\mathrm{add}(x,y)) \approx \mathrm{add}(x,\mathrm{add}(x,y))\\
&\mathrm{pop}(\mathrm{dup}(x)) \approx x\\
&\mathrm{size}(\mathrm{empty}) \approx 0\\
&\mathrm{size}(\mathrm{add}(x,y)) \approx 1+\mathrm{size}(y)\\
&\mathrm{size}(\mathrm{dup}(x)) \approx 1+\mathrm{size}(x)
\end{aligned}$$

- Égalités = règles d'évaluation dans les preuves

$$\mathrm{add}(a,\mathrm{pop}(\mathrm{add}(b,\mathrm{pop}(\mathrm{dup}(\mathrm{add}(a,\mathrm{empty}))))))$$

Spécifications équationnelles

- Définition d'une structure de données

$$\begin{aligned}
&\mathrm{pop}(\mathrm{empty}) \approx \mathrm{empty}\\
&\mathrm{pop}(\mathrm{add}(x,y)) \approx y\\
&\mathrm{top}(\mathrm{add}(x,y)) \approx x\\
&\mathrm{dup}(\mathrm{add}(x,y)) \approx \mathrm{add}(x,\mathrm{add}(x,y))\\
&\mathrm{pop}(\mathrm{dup}(x)) \approx x\\
&\mathrm{size}(\mathrm{empty}) \approx 0\\
&\mathrm{size}(\mathrm{add}(x,y)) \approx 1+\mathrm{size}(y)\\
&\mathrm{size}(\mathrm{dup}(x)) \approx 1+\mathrm{size}(x)
\end{aligned}$$

- Égalités = règles d'évaluation dans les preuves

$$\begin{aligned}
& \mathrm{add}(a,\mathrm{pop}(\mathrm{add}(b,\mathrm{pop}(\mathrm{dup}(\mathrm{add}(a,\mathrm{empty})))))) \\
\approx\ & \mathrm{add}(a,\mathrm{pop}(\mathrm{add}(b,\mathrm{add}(a,\mathrm{empty}))))
\end{aligned}$$

Spécifications équationnelles

- Définition d'une structure de données

$$\begin{aligned}
&\mathrm{pop}(\mathrm{empty}) \approx \mathrm{empty}\\
&\mathrm{pop}(\mathrm{add}(x,y)) \approx y\\
&\mathrm{top}(\mathrm{add}(x,y)) \approx x\\
&\mathrm{dup}(\mathrm{add}(x,y)) \approx \mathrm{add}(x,\mathrm{add}(x,y))\\
&\mathrm{pop}(\mathrm{dup}(x)) \approx x\\
&\mathrm{size}(\mathrm{empty}) \approx 0\\
&\mathrm{size}(\mathrm{add}(x,y)) \approx 1+\mathrm{size}(y)\\
&\mathrm{size}(\mathrm{dup}(x)) \approx 1+\mathrm{size}(x)
\end{aligned}$$

- Égalités = règles d'évaluation dans les preuves

$$\begin{aligned}
& \mathrm{add}(a,\mathrm{pop}(\mathrm{add}(b,\mathrm{pop}(\mathrm{dup}(\mathrm{add}(a,\mathrm{empty})))))) \\
\approx\ & \mathrm{add}(a,\mathrm{pop}(\mathrm{add}(b,\mathrm{add}(a,\mathrm{empty}))))
\end{aligned}$$

Spécifications équationnelles

- Définition d'une structure de données

$$
\begin{aligned}
&\mathrm{pop}(\mathrm{empty}) \approx \mathrm{empty}\\
&\mathrm{pop}(\mathrm{add}(x,y)) \approx y\\
&\mathrm{top}(\mathrm{add}(x,y)) \approx x\\
&\mathrm{dup}(\mathrm{add}(x,y)) \approx \mathrm{add}(x,\mathrm{add}(x,y))\\
&\mathrm{pop}(\mathrm{dup}(x)) \approx x\\
&\mathrm{size}(\mathrm{empty}) \approx 0\\
&\mathrm{size}(\mathrm{add}(x,y)) \approx 1+\mathrm{size}(y)\\
&\mathrm{size}(\mathrm{dup}(x)) \approx 1+\mathrm{size}(x)
\end{aligned}
$$

- Égalités = règles d'évaluation dans les preuves

$$
\begin{aligned}
& \mathrm{add}(a,\mathrm{pop}(\mathrm{add}(b,\mathrm{pop}(\mathrm{dup}(\mathrm{add}(a,\mathrm{empty})))))) \\
\approx\ & \mathrm{add}(a,\mathrm{pop}(\mathrm{add}(b,\mathrm{add}(a,\mathrm{empty})))) \\
\approx\ & \mathrm{add}(a,\mathrm{add}(a,\mathrm{empty}))
\end{aligned}
$$

Égalité modulo E

$$E = \begin{cases} \mathrm{pop}(\mathrm{empty}) \approx \mathrm{empty} \\ \mathrm{pop}(\mathrm{add}(x,y)) \approx y \\ \mathrm{top}(\mathrm{add}(x,y)) \approx x \\ \mathrm{dup}(\mathrm{add}(x,y)) \approx \mathrm{add}(x,\mathrm{add}(x,y)) \\ \mathrm{pop}(\mathrm{dup}(x)) \approx x \\ \mathrm{size}(\mathrm{empty}) \approx 0 \\ \mathrm{size}(\mathrm{add}(x,y)) \approx 1 + \mathrm{size}(y) \\ \mathrm{size}(\mathrm{dup}(x)) \approx 1 + \mathrm{size}(x) \end{cases}$$

Égalité modulo E :

$$E \models^{?} \mathrm{pop}(\mathrm{dup}(\mathrm{add}(a,\mathrm{empty}))) \approx \mathrm{empty}$$

ou encore

$$\mathrm{pop}(\mathrm{dup}(\mathrm{add}(a,\mathrm{empty}))) \approx^{?}_{E} \mathrm{empty}$$

Plan

Systèmes de réduction

Définition : un système de réduction est un couple $(A, \rightarrow)$ où A est un ensemble et $\rightarrow$ une relation binaire sur A

Notations :

- $\overset{n+1}{\rightarrow} = \overset{n}{\rightarrow} \circ \rightarrow$
- $\overset{+}{\rightarrow} = \bigcup_{i \geqslant 1} \overset{i}{\rightarrow}$ fermeture transitive
- $\overset{*}{\rightarrow} = \bigcup_{i \geqslant 0} \overset{i}{\rightarrow}$ fermeture réflexive transitive
- $\overset{=}{\rightarrow} = \rightarrow \circ \overset{0}{\rightarrow}$ fermeture réflexive
- $\leftarrow = \{(y, x) | (x, y) \in \rightarrow\}$ inverse
- $\leftrightarrow = \rightarrow \cup \leftarrow$ fermeture symétrique
- $\overset{*}{\leftrightarrow} = (\leftrightarrow)^*$ fermeture symétrique réflexive transitive

Induction bien fondée

Une relation $\prec \subseteq A \times A$ est dite *bien fondée* s'il n'existe pas de suite infinie $a_1, \ldots, a_n, \ldots$ d'éléments de A telle que

$$\cdots \prec a_n \prec \cdots \prec a_1$$

Principe d'induction bien fondée

Soit Q un prédicat sur A

$$\frac{\forall x \in A, (\forall y \in A, y \prec x \implies Q(y)) \implies Q(x)}{\forall x \in A, Q(x)}$$

Théorème

Le principe d'induction bien fondée est valide ssi la relation $\prec$ est bien fondée.

Confluence, terminaison et normalisation

Forme normale :

- x est en forme normale (irréductible) s'il n'existe pas de y tel que $x \to y$
- x et y sont joignables s'il existe z tel que $x \xrightarrow{*} z \xleftarrow{*} y$, noté $x \downarrow y$

Un système de réduction est

- « Church-Rosser » ssi $x \overset{*}{\leftrightarrow} y \Rightarrow x \downarrow y$
- confluent ssi $y_1 \xleftarrow{*} x \xrightarrow{*} y_2 \Rightarrow y_1 \downarrow y_2$
- terminant ss'il n'existe pas de chaîne infinie $a_0 \to a_1 \to \dots$ (la relation $\leftarrow$ est bien fondée)
- normalisant ssi tout élément a une forme normale
- convergent ss'il est à la fois confluent et terminant

Propriétés générales

Théorème

Un système est Church-Rosser ssi il est confluent.

Preuve : Exercice □

Corollaire

Si $\rightarrow$ est confluent, tout élément a au plus une forme normale.

Corollaire

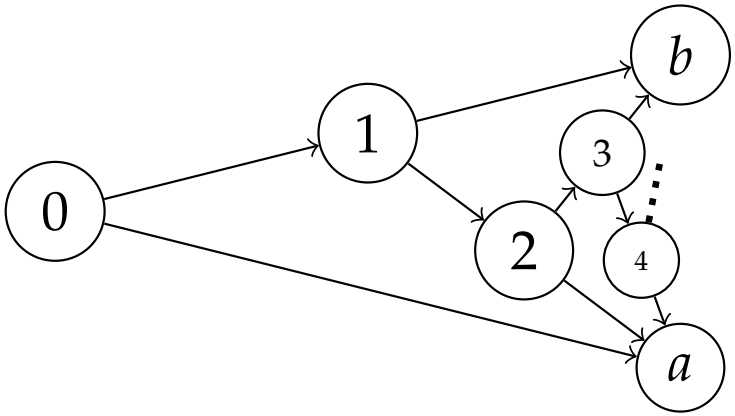
Si $\rightarrow$ est normalisant et confluent, tout élément a une forme normale unique.

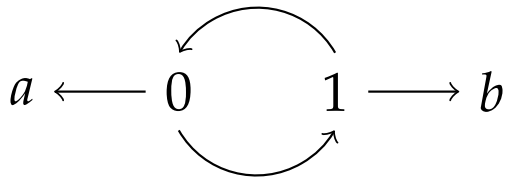
Théorème

Si $\rightarrow$ est normalisant et confluent, alors $x \overset{*}{\leftrightarrow} y \Leftrightarrow x\downarrow = y\downarrow$.

Confluence locale

Une relation $\rightarrow$ est localement confluente si $y_1 \leftarrow x \rightarrow y_2 \Rightarrow y_1 \downarrow y_2$





Lemme (Newman)

Une relation terminante est confluente ssi elle est localement confluente

Preuve du lemme de Newman

Induction bien fondée sur la propriété

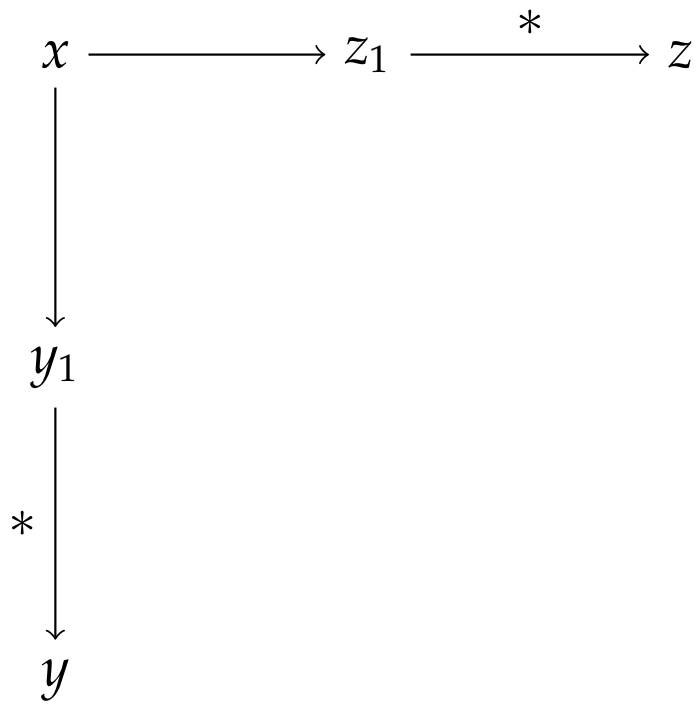
$$P(x) = \forall y, \forall z, y \overset{*}{\leftarrow} x \overset{*}{\rightarrow} z \Rightarrow y \downarrow z$$

Relation bien fondée ?

Preuve du lemme de Newman

Induction bien fondée sur la propriété

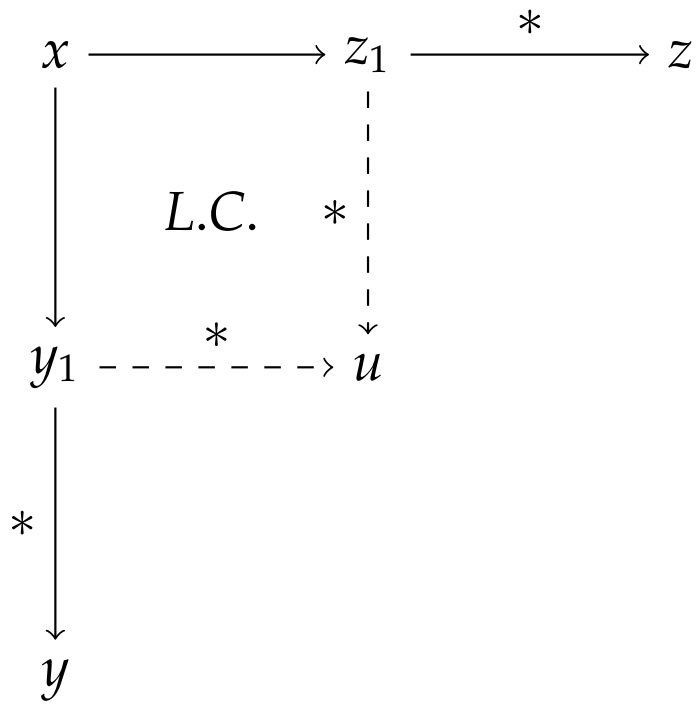
$$P(x) = \forall y, \forall z, y \overset{*}{\leftarrow} x \overset{*}{\rightarrow} z \Rightarrow y \downarrow z$$



Preuve du lemme de Newman

Induction bien fondée sur la propriété

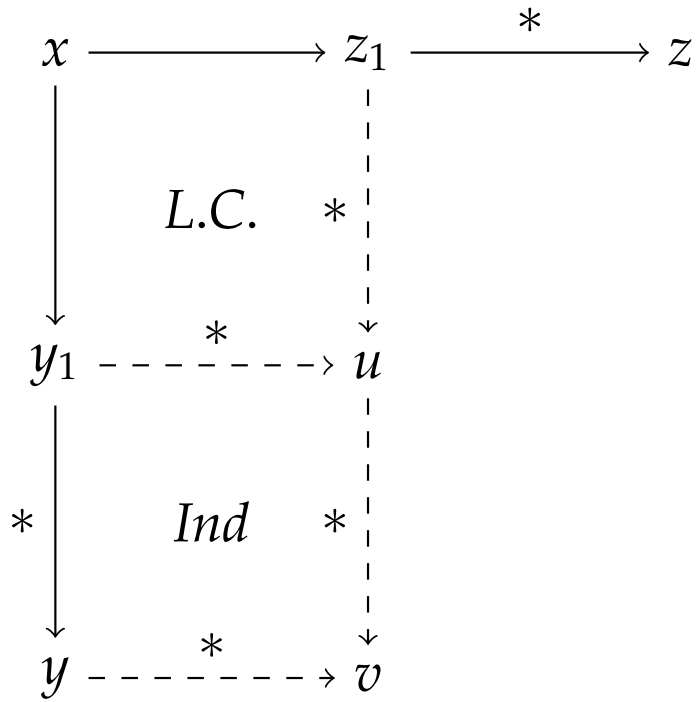
$$P(x) = \forall y, \forall z, y \overset{*}{\leftarrow} x \overset{*}{\rightarrow} z \Rightarrow y \downarrow z$$



Preuve du lemme de Newman

Induction bien fondée sur la propriété

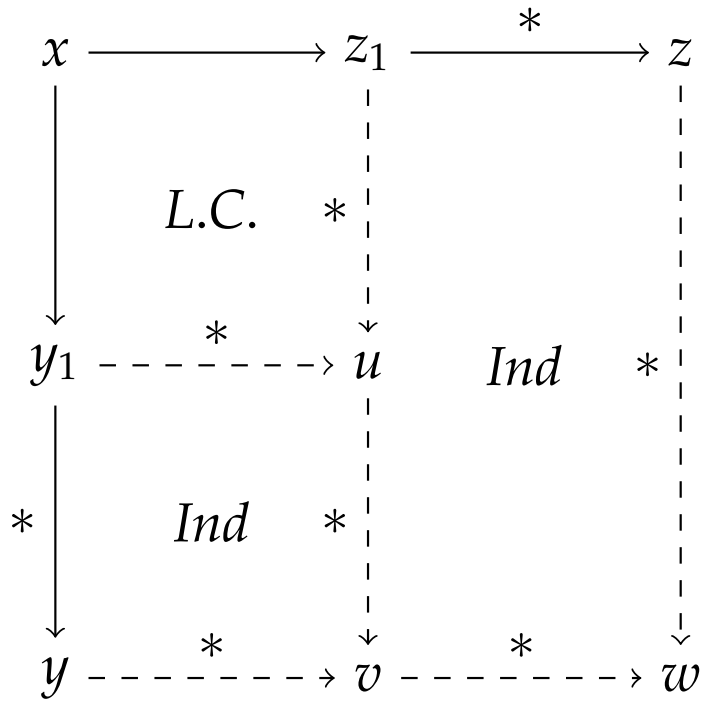
$$P(x) = \forall y, \forall z, y \overset{*}{\leftarrow} x \overset{*}{\rightarrow} z \Rightarrow y \downarrow z$$



Preuve du lemme de Newman

Induction bien fondée sur la propriété

$$P(x) = \forall y, \forall z, y \overset{*}{\leftarrow} x \overset{*}{\rightarrow} z \Rightarrow y \downarrow z$$



Plan

Termes

Rappel : soit $\mathcal{F}$ un ensemble de symboles munis d'une arité, et $\mathcal{X}$ un ensemble dénombrable de variables, l'ensemble $\mathcal{T}(\mathcal{F},\mathcal{X})$ des termes construits sur $\mathcal{F}$ et $\mathcal{X}$ est défini inductivement par

$$t \quad ::= \quad x \quad | \quad f(t_1,\ldots,t_n)$$

avec $x \in \mathcal{X}, f \in \mathcal{F}$ un symbole de fonction d'arité n et $t_1,\ldots,t_n$ des termes de $\mathcal{T}(\mathcal{F},\mathcal{X})$.
L'ensemble $\mathcal{T}(\mathcal{F})$ des termes clos est défini par

$$t \quad ::= \quad f(t_1,\ldots,t_n)$$

Exemples : soit $\mathcal{F} = \{f : 3, \text{cons} : 2, \text{car} : 1, a : 0, nil : 0\}$, et $x \in \mathcal{X}$, et soient

$$\begin{array}{rcl} s & = & f(a,x,\text{nil}) \\ t & = & \text{car}(\text{cons}(a,\text{cons}(a,\text{nil}))) \end{array}$$

On a $s,t \in \mathcal{T}(\mathcal{F},\mathcal{X})$, et $t \in \mathcal{T}(\mathcal{F})$ et $Var(s) = \{x\}$.

Notation : on notera $t|_p$ le sous-terme de t à la position p.

Substitutions

Définition : Une substitution $\sigma = [t_1/x_1, \ldots, t_n/x_n]$ est une fonction dont l'application à un terme t est le terme σt dans lequel toutes les occurrences des variables $x_1, ..., x_n$ ont été substituées par $t_1, ..., t_n$. La substitution σ est prolongée de façon classique sur l'ensemble des termes.

Exemple : soit $\mathcal{F} = \{f : 3, h : 1, g : 1, a : 0\}$, soit $\sigma = [x \mapsto g(a), y \mapsto h(z)]$ et $t = f(h(x), x, g(y))$.
On a $\sigma t = f(h(g(a)), g(a), g(h(z)))$.

Définition (substitution plus générale) :
Soient deux substitutions σ et θ. On dit que σ est plus générale que θ s'il existe une substitution β telle que $\beta \circ \sigma = \theta$.

Exemple :
$\sigma = [g(y, z)/x]$ est plus générale que $\theta = [g(a, z)/x]$ avec $\beta = [a/y]$.

Notation : on notera $t[u]_p$ le terme t dans lequel le sous-terme u est greffé à la position p.

Théorie équationnelle

Définition : une théorie équationnelle E est un ensemble d'égalités de la forme $l \approx r$ où $l, r \in \mathcal{T}(\mathcal{F}, \mathcal{X})$.

Définition : la relation $\rightarrow_E$ est définie par $s \rightarrow_E t$ ssi il existe une égalité $l \approx r$ de E, une position p de s et une substitution σ telles que

$$s|_p = \sigma l \text{ et } t = s[\sigma r]_p$$

Proposition

la relation $\overset{*}{\leftrightarrow}_E$ est la plus petite relation d'équivalence $\sim$ vérifiant

1. pour tout f dans $\mathcal{F}$ d'arité n,
   $(\forall i \in \{1, \ldots, n\}, t_i \sim u_i) \implies f(t_1, ..., t_n) \sim f(u_1, ..., u_n)$ (compatibilité)
2. pour toute substitution σ et toute égalité $l \approx r$ de E, $\sigma l \sim \sigma r$ (substitution)

Exemple : entiers de Peano

$$E = \begin{cases} 0 + x \approx x \\ S(x) + y \approx S(x + y) \end{cases}$$

A-t-on

$$S(0) + S(0) \overset{*}{\leftrightarrow}_E S(S(0)) \quad ?$$

Exemple : entiers de Peano

$$E = \begin{cases} 0 + x \approx x \\ S(x) + y \approx S(x + y) \end{cases}$$

A-t-on

$$S(0) + S(0) \overset{*}{\leftrightarrow}_E S(S(0)) \quad ?$$

Oui :

$$S(0) + S(0) \overset{*}{\leftrightarrow}_E S(0 + S(0))$$

Exemple : entiers de Peano

$$E = \left\{ \begin{array}{l} 0 + x \approx x \\ S(x) + y \approx S(x + y) \end{array} \right.$$

A-t-on

$$S(0) + S(0) \overset{*}{\leftrightarrow}_E S(S(0)) \quad ?$$

Oui :

$$S(0) + S(0) \overset{*}{\leftrightarrow}_E S(0 + S(0)) \overset{*}{\leftrightarrow}_E S(S(0))$$

Exemple : ensembles

Soit $\mathcal{F} = \{\cap : 2, \cup : 2, \emptyset : 0, \{\cdot\} : 1\}$, $\mathcal{X} = \{A, B, C, x, y\}$, et E contenant les six égalités suivantes :

1. $A \cap \emptyset \approx \emptyset$
2. $A \cup \emptyset \approx A$
3. $A \cup (B \cap C) \approx (A \cup B) \cap (A \cup C)$
4. $A \cup A \approx A$
5. $A \cap B \approx B \cap A$
6. $A \cup B \approx B \cup A$

Prouvez les équivalences suivantes

- $(\{x\} \cup \{y\}) \cap \{y\} \overset{*}{\leftrightarrow}_E \{y\}$
- $A \cap (A \cup B) \overset{*}{\leftrightarrow}_E A$
- $A \cap A \overset{*}{\leftrightarrow}_E A$

Exemple : ensembles

Soit $\mathcal{F} = \{\cap : 2, \cup : 2, \emptyset : 0, \{\cdot\} : 1\}$, $\mathcal{X} = \{A, B, C, x, y\}$, et E contenant les six égalités suivantes :

1. $A \cap \emptyset \approx \emptyset$
2. $A \cup \emptyset \approx A$
3. $A \cup (B \cap C) \approx (A \cup B) \cap (A \cup C)$
4. $A \cup A \approx A$
5. $A \cap B \approx B \cap A$
6. $A \cup B \approx B \cup A$

$$
\begin{array}{lll}
(\{x\} \cup \{y\}) \cap \{y\} & =_{(6)} & (\{y\} \cup \{x\}) \cap \{y\} \\
& =_{(2)} & (\{y\} \cup \{x\}) \cap (\{y\} \cup \emptyset) \\
& =_{(3)} & \{y\} \cup (\{x\} \cap \emptyset) \\
& =_{(1)} & \{y\} \cup \emptyset =_{(2)} \{y\}
\end{array}
$$

Exemple : ensembles

Soit $\mathcal{F} = \{\cap : 2, \cup : 2, \emptyset : 0, \{\cdot\} : 1\}$, $\mathcal{X} = \{A, B, C, x, y\}$, et E contenant les six égalités suivantes :

1. $A \cap \emptyset \approx \emptyset$
2. $A \cup \emptyset \approx A$
3. $A \cup (B \cap C) \approx (A \cup B) \cap (A \cup C)$
4. $A \cup A \approx A$
5. $A \cap B \approx B \cap A$
6. $A \cup B \approx B \cup A$

$$\begin{array}{lll} (A \cap (A \cup B) & =_{(2)} & (A \cup \emptyset) \cap (A \cup B) \\ & =_{(3)} & A \cup (\emptyset \cap B) \\ & =_{(5)} & A \cup (B \cap \emptyset) \\ & =_{(1)} & A \cup \emptyset =_{(2)} A \end{array}$$

Exemple : ensembles

Soit $\mathcal{F} = \{\cap : 2, \cup : 2, \emptyset : 0, \{\cdot\} : 1\}$, $\mathcal{X} = \{A, B, C, x, y\}$, et E contenant les six égalités suivantes :

1. $A \cap \emptyset \approx \emptyset$
2. $A \cup \emptyset \approx A$
3. $A \cup (B \cap C) \approx (A \cup B) \cap (A \cup C)$
4. $A \cup A \approx A$
5. $A \cap B \approx B \cap A$
6. $A \cup B \approx B \cup A$

Soit (7) le théorème précédent :

$$A \cap (A \cup B) = A$$

$$A \cap A =_{(2)} A \cap (A \cup \emptyset) =_{(7)} A$$

Modèles

Définition : une algèbre $\mathcal{A}$ sur $\mathcal{F}$ est constituée d'un ensemble A (son domaine) et, pour tout symbole de fonction f de $\mathcal{F}$ d'arité n, de la donnée d'une application $f^{\mathcal{A}} : A^n \to A$.

Exemple : l'ensemble des termes $\mathcal{T}(\mathcal{F}, \mathcal{X})$ peut être considéré comme une algèbre, si l'on interprète les symboles de fonctions par eux-mêmes.

Définition : un morphisme d'algèbre entre les algèbres $\mathcal{A}$ et $\mathcal{B}$ est une application de A dans B qui est un morphisme pour toutes les interprétations des symboles de fonctions de $\mathcal{F}$.

Exemple : $\mathcal{F} = \{\emptyset, \cap, \cup\}$

	$\mathcal{B}$	$\mathcal{A}$
domaine	$\{0, 1\}$	$\mathbb{N}$
$\emptyset$	0	0
$\cap$	$\wedge$	min
$\cup$	$\vee$	max

morphismes ?

Modèles (suite)

Définition : l'égalité $l \approx r$ est valide dans l'algèbre $\mathcal{A}$ si pour tout morphisme $\phi : \mathcal{T}(\mathcal{F}, \mathcal{X}) \to \mathcal{A}$ on a $\phi(l) = \phi(r)$.

Définition : l'algèbre $\mathcal{A}$ est un modèle de E si toute égalité de E est valide dans $\mathcal{A}$

Définition : l'égalité $s \approx t$ est conséquence sémantique de E (noté $s \approx_E t$) si elle est valide dans tout modèle de E.

Théorème (Birkhoff)

$$s \approx_E t \iff s \overset{*}{\leftrightarrow}_E t$$

Notation : dans la suite on utilisera indifféremment $\approx_E$ et $\overset{*}{\leftrightarrow}_E$.

Plan

Décision

Définition : problème du mot (égalité / unification modulo E)
Étant donnés une théorie équationnelle E et deux termes s et t, a-t-on $s \approx_E t$?

Ce problème est indécidable : on peut coder le comportement d'une machine de Turing avec une théorie équationnelle.

Pourtant, étant donnés

$$E = \begin{cases} \mathrm{car}(\mathrm{cons}(x, y)) \approx x \\ \mathrm{cdr}(\mathrm{cons}(x, y)) \approx y \\ \mathrm{cons}(\mathrm{car}(x), \mathrm{cdr}(x)) \approx x \end{cases}$$

on aimerait pouvoir décider automatiquement si

$$\mathrm{car}(\mathrm{cdr}(\mathrm{cons}(1, \mathrm{cons}(2, \mathrm{nil})))) \approx_E \mathrm{car}(\mathrm{cons}(\mathrm{car}(\mathrm{cons}(2, \mathrm{cons}(1, \mathrm{nil}))), \mathrm{nil}))$$

Solutions partielles

Cas particulier : $E = \emptyset$

Dans ce cas, le problème devient : étant donnés deux termes s et t, existe-t-il une substitution σ telle que $\sigma t = s$?

▷ unification

Autres cas : orienter les égalités en règles de réécriture, de la gauche vers la droite

Exemple : l'ensemble E d'égalités pour les entiers de Peano devient

$$R = \begin{cases} (1) & 0 + x \to x \\ (2) & S(x) + y \to S(x + y) \end{cases}$$

Décision par réécriture

Une règle de réécriture $l \to r$ est une égalité $l \approx r$ telle que l n'est pas une variable, et $Var(r) \subseteq Var(l)$.
La relation $\to_R$ de réécriture est alors la relation $\to_E$ vue précédemment.

Comment décider $\approx_E$ à l'aide de $\to_R$?

$$\begin{array}{ccc} S(0)+0 & \approx_E^? & 0+S(0) \\ {\scriptstyle R}\downarrow{\scriptstyle (2)} & & {\scriptstyle R}\downarrow{\scriptstyle (1)} \\ S(0+0) & & S(0) \\ & & \vdots \\ {\scriptstyle R}\downarrow{\scriptstyle (1)} & & \\ S(0) & = & S(0) \end{array}$$

On cherche donc à calculer des formes normales pour s et t. Une orientation « au hasard » ne suffit pas...

- confluence
- terminaison

Plan

Confluence

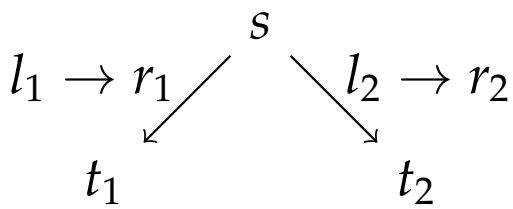
Décision : le problème « Étant donné un système de réécriture E fini, est-il confluent ? » est indécidable (réduction à partir du problème du mot).

Confluence locale : elle est décidable pour les systèmes terminants ! On va donc pouvoir utiliser le lemme de Newman.

$$\text{terminaison} \quad \Longrightarrow \quad (\text{confluence} \equiv \text{confluence locale})$$

Paires critiques

Étant donnés s, t_1 et t_2 tels que



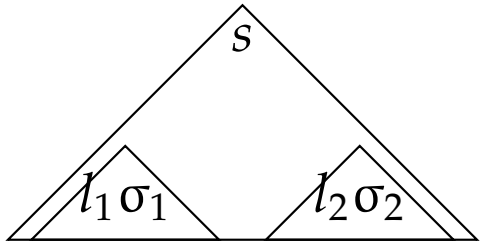
Pour $i = 1, 2$, il existe p_i et σ_i tels que $s|_{p_i} = l_i\sigma_i$ et $t_i = s[r_i\sigma_i]|_{p_i}$.
Comment savoir si

$$t_1 \downarrow t_2 \quad ?$$

Remarque : les deux règles peuvent être deux instances de la même règle, modulo renommage des variables

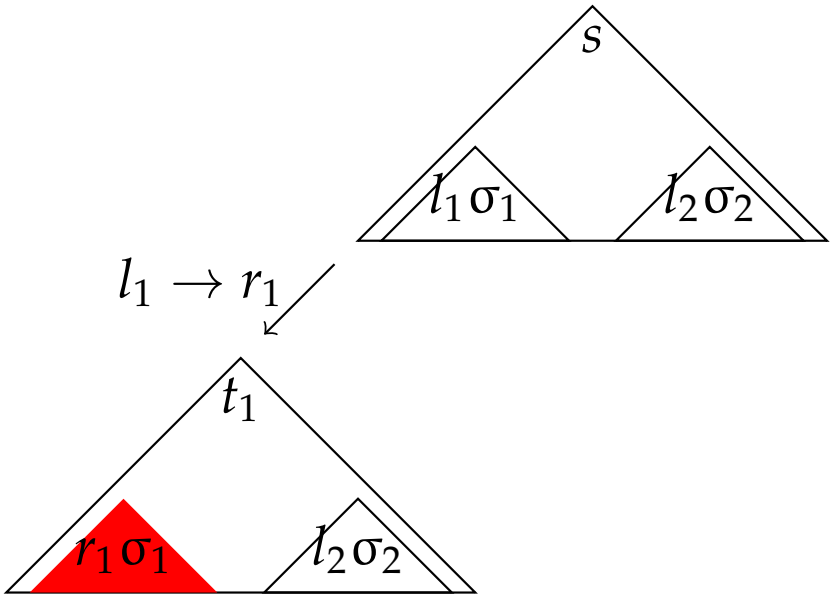
Paires critiques : cas 1

p_1 et p_2 sont dans deux sous-arbres distincts.



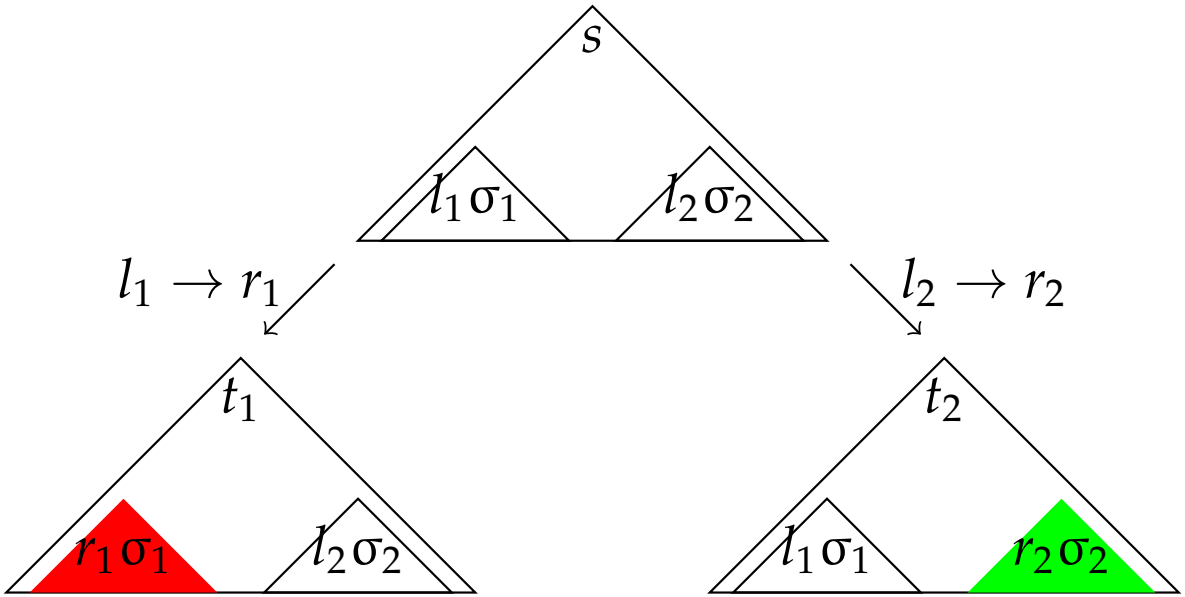
Paires critiques : cas 1

p_1 et p_2 sont dans deux sous-arbres distincts.



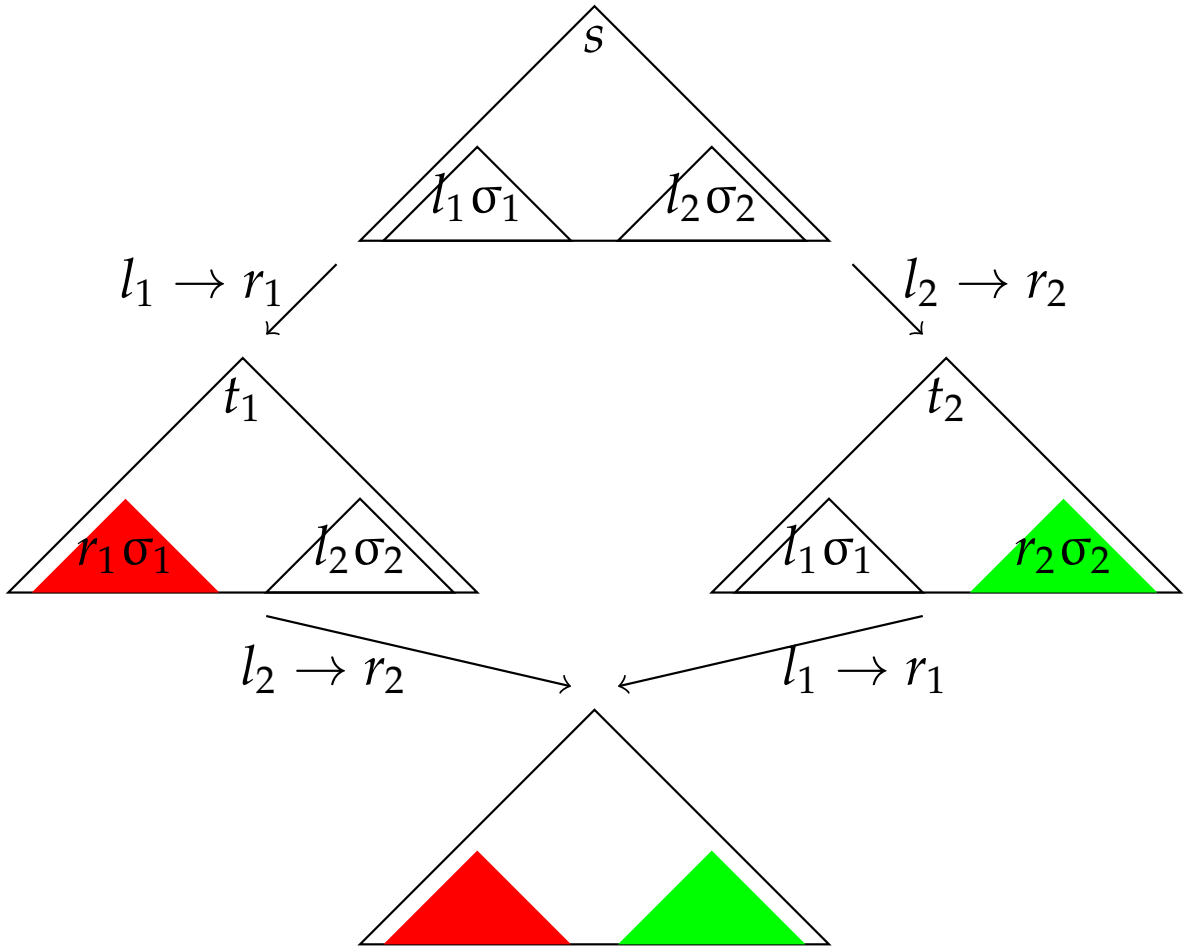
Paires critiques : cas 1

p_1 et p_2 sont dans deux sous-arbres distincts.



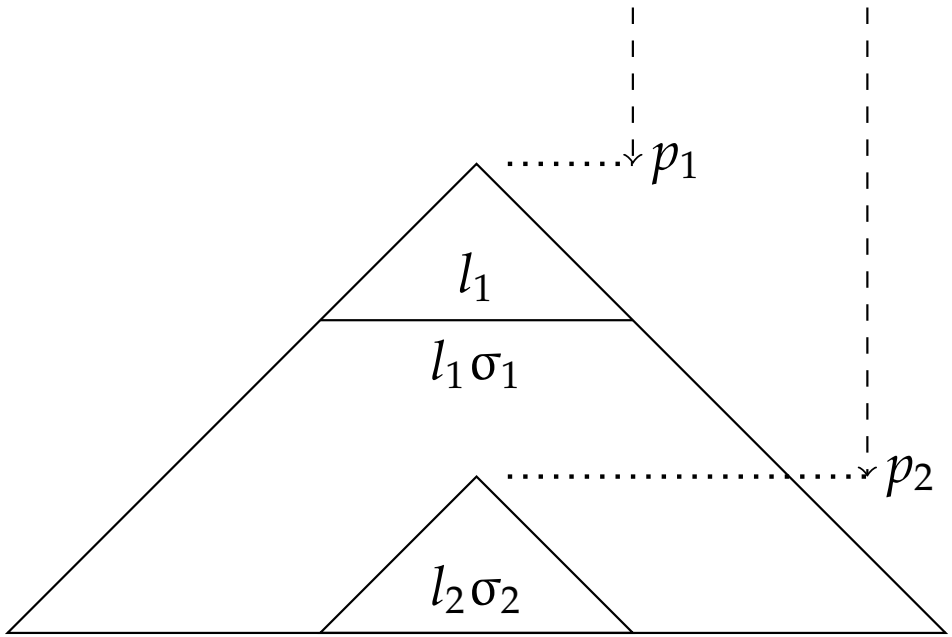
Paires critiques : cas 1

p_1 et p_2 sont dans deux sous-arbres distincts.

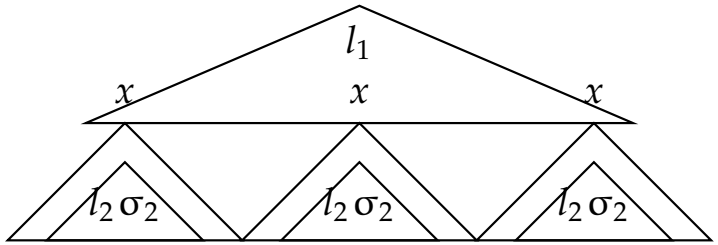


Paires critiques : cas 2

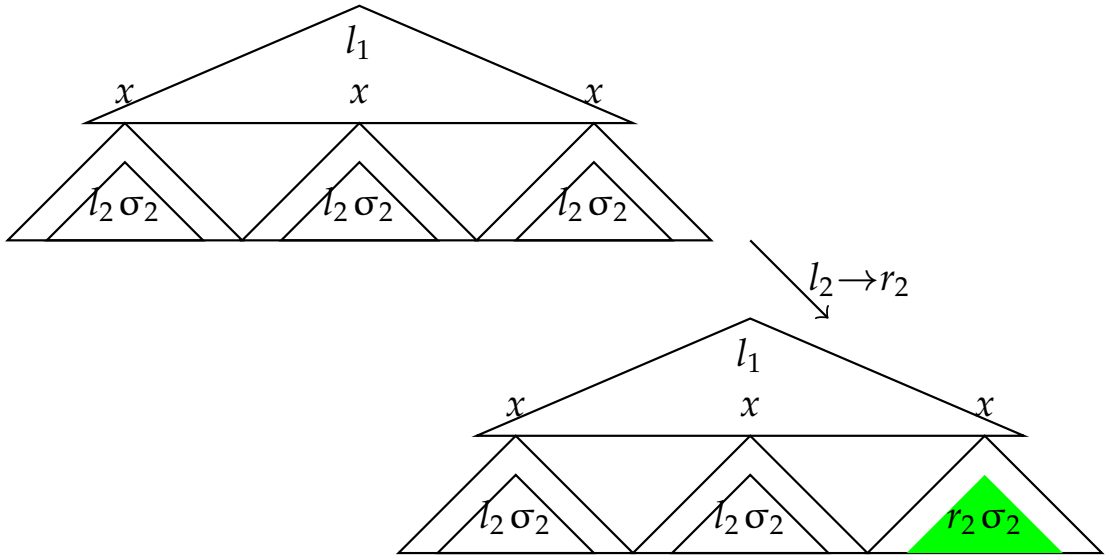
p_1 est un préfixe de p_2, $l_2\sigma_2$ apparaît dans $l_1\sigma_1$ mais pas dans l_1.



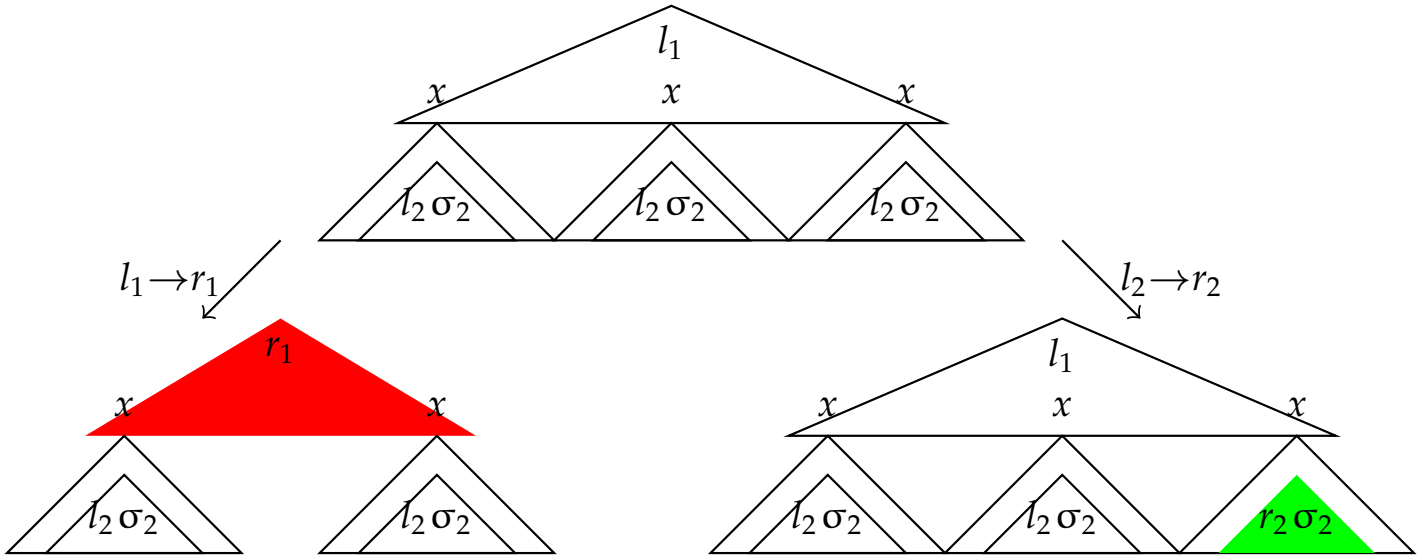
Paires critiques : cas 2



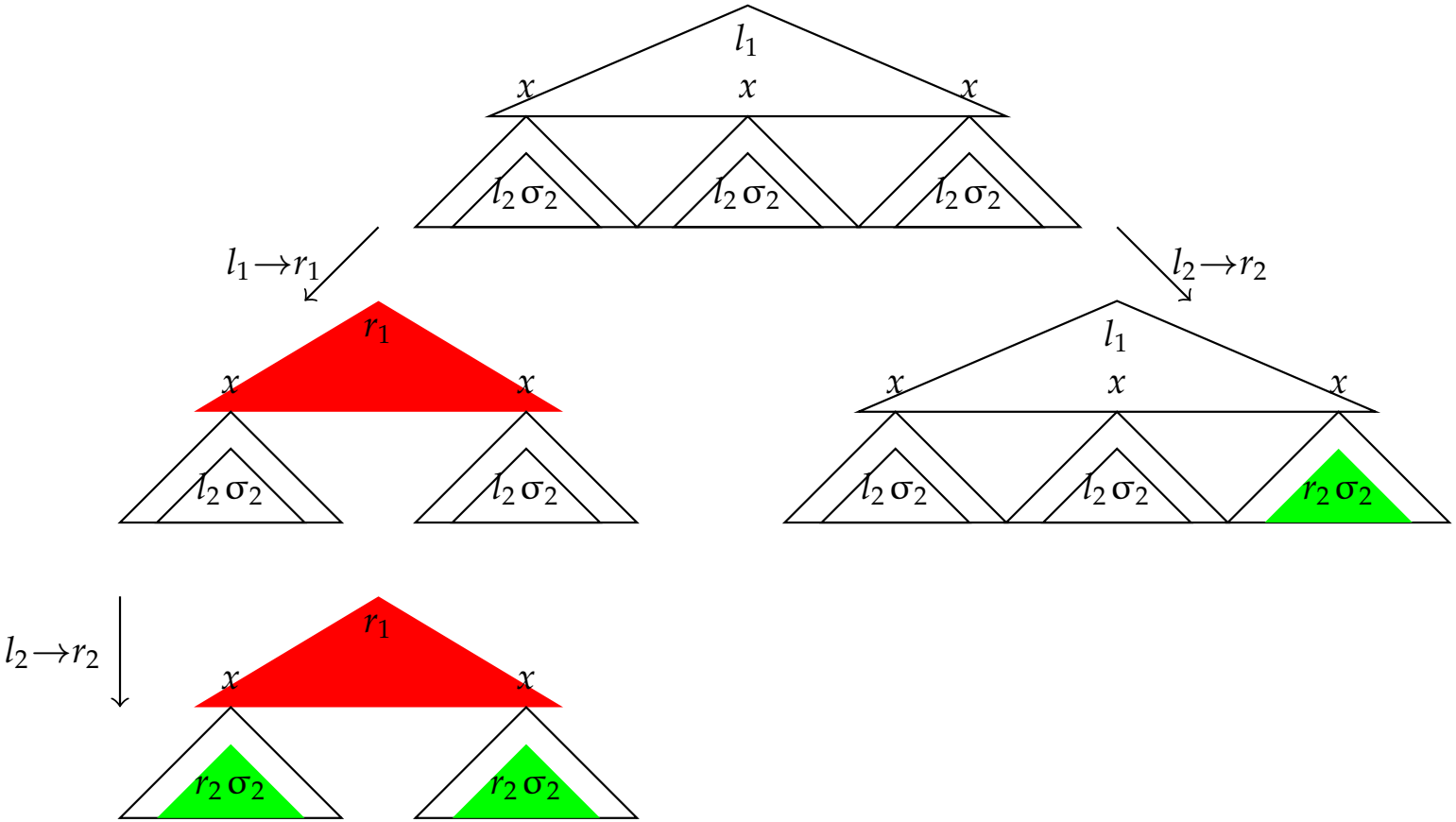
Paires critiques : cas 2



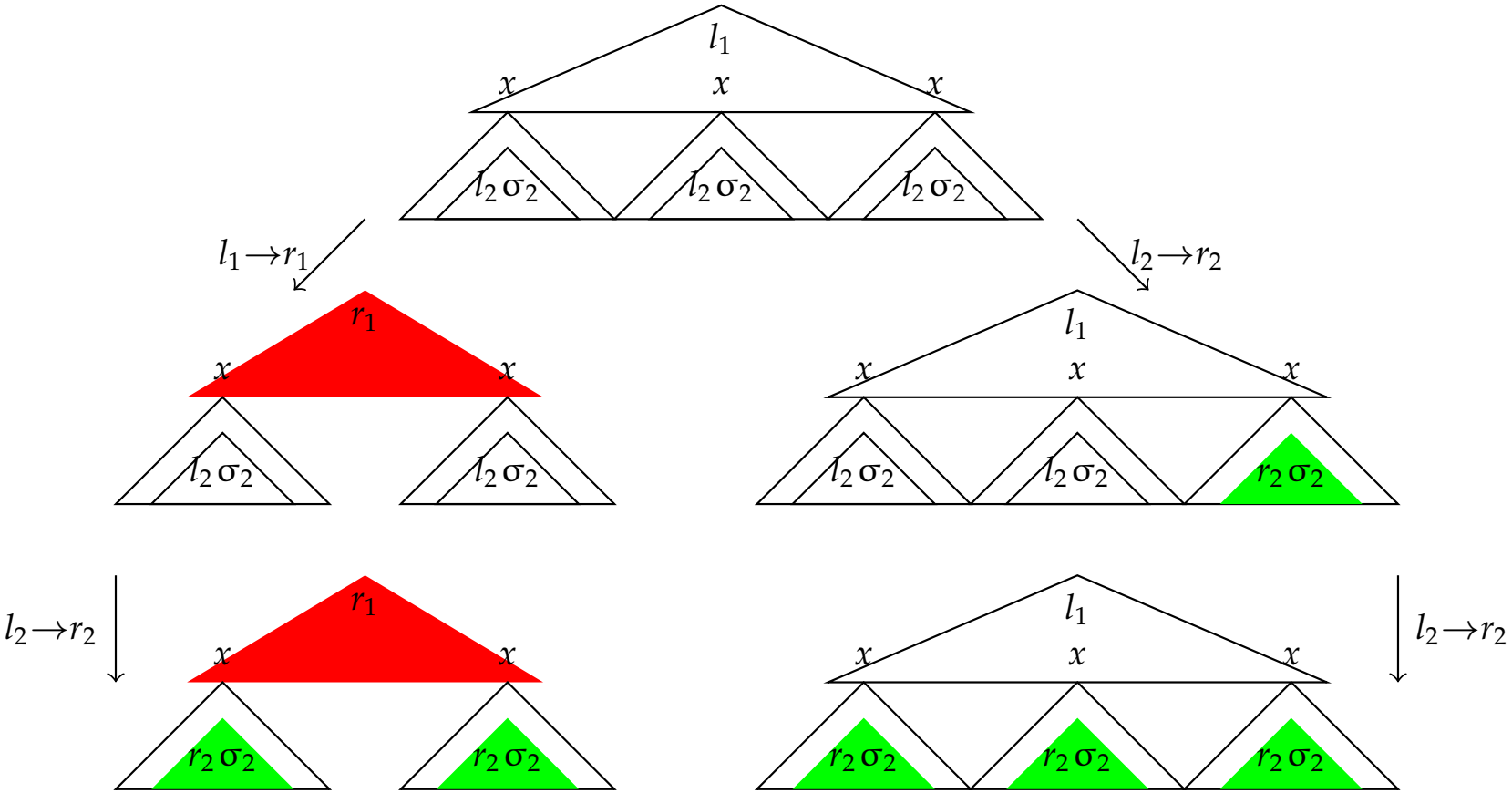
Paires critiques : cas 2



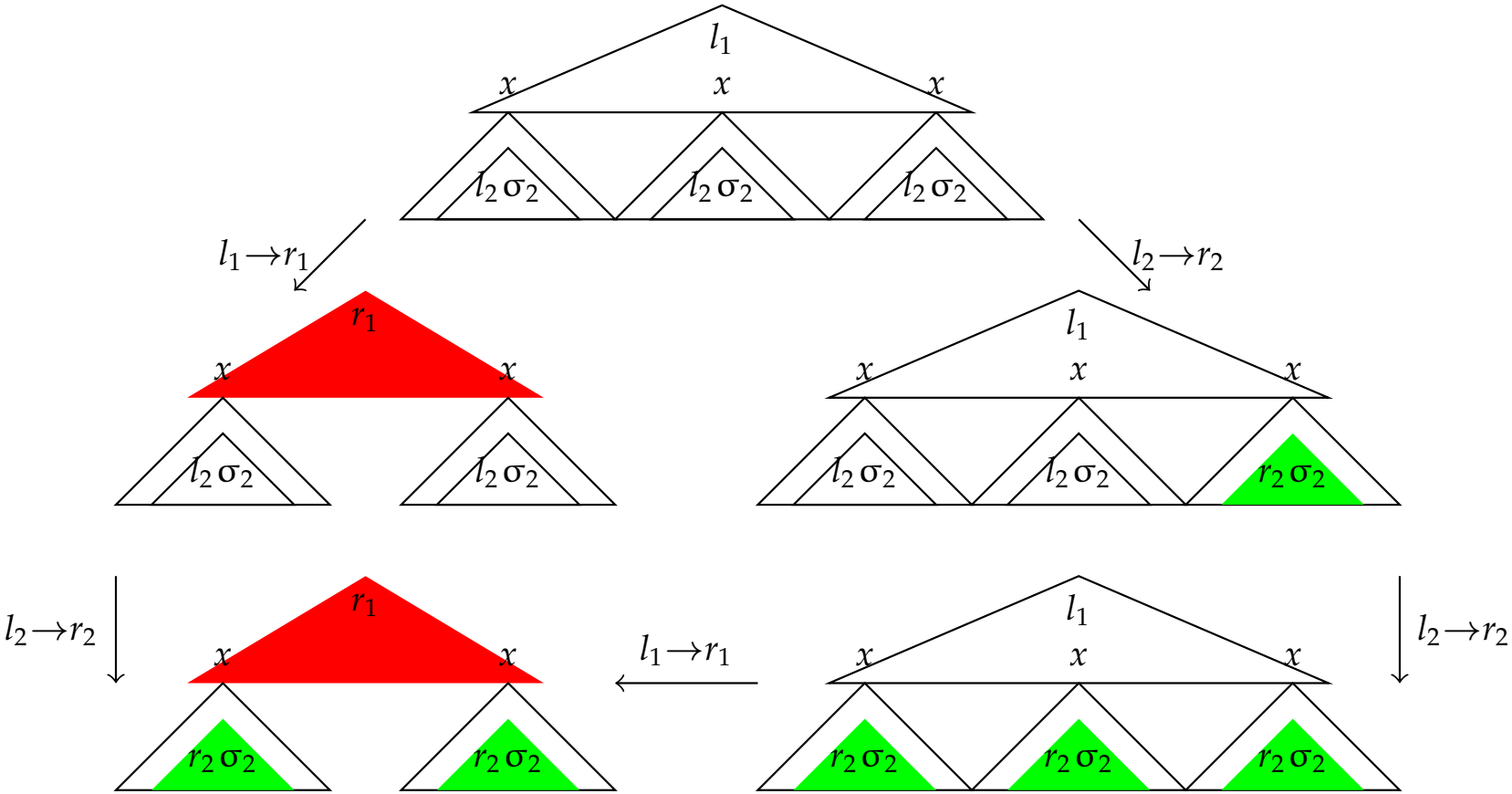
Paires critiques : cas 2



Paires critiques : cas 2

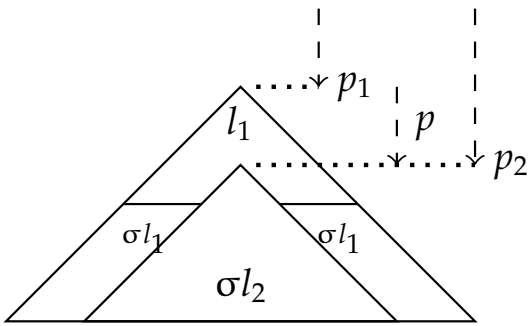


Paires critiques : cas 2



Paires critiques : cas 3

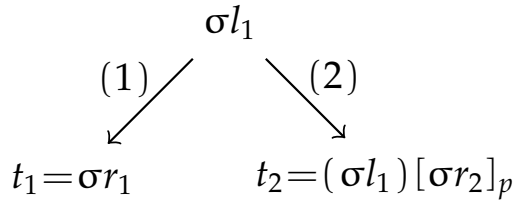
p_1 est un préfixe de p_2, $p_2 = p_1 \cdot p$, σl_2 apparaît à une position de l_1.



Définition : Soient

$$(1)\ l_1 \to r_1$$
$$(2)\ l_2 \to r_2$$

deux règles telles que les variables sont renommées de façon à avoir $Var(l_1, r_1) \cap Var(l_2, r_2) = \emptyset$. Soit σ un unificateur principal pour l_2 et un sous-terme t de l_1 à la position p, $t \notin \mathcal{X}$ (on a donc $\sigma l_2 = \sigma t$)
Si



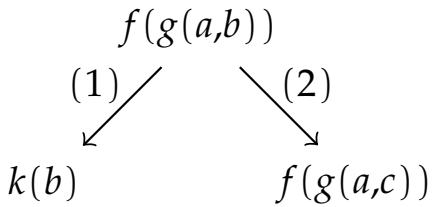
et $t_1 \neq t_2$ alors (t_1, t_2) est une paire critique.

Paires critiques

Exemple :

$$R = \{(1)\, f(g(a,x)) \to k(x), (2)\, g(y,b) \to g(y,c)\}$$

Avec $\sigma = \{x \mapsto b, y \mapsto a\}$ on a



$(k(b), f(g(a,c))$ est une paire critique, qui n'est pas joignable par R.

Proposition

Si les paires critiques sont joignables alors le système de réécriture est localement confluent.

En résumé

Théorème

Un système de réécriture terminant est confluent si et seulement si ses paires critiques sont joignables.

Théorème

La confluence d'un système de réécriture terminant est décidable.

Et pour les systèmes non terminants ?

Avec certaines restrictions (linéarité, absence de paires critiques...) on peut définir des systèmes non terminants mais confluents : application aux langages fonctionnels.

Complétion de Knuth-Bendix

- Si le système comporte des paires critiques (b, c) non joignables, on ajoute la règle $b \to_R c$ ou $c \to_R b$.
- Le système devient confluent.
- On a peut-être perdu la terminaison...

Plan

Terminaison

Décision : le problème « Étant donné un système de réécriture E fini, et un terme t, existe-t-il une suite infinie de réductions issue de t ? » est indécidable : on le réduit au problème de l'arrêt (universel) d'une machine de Turing.

Cas particulier : s'il n'y a pas de variable dans la partie droite des règles, la terminaison devient décidable.

Comment prouver la terminaison pour un système donné ?

- Par « plongement » : on définit $\phi : \mathcal{T}(\mathcal{F}, \mathcal{X}) \to K$, où K est muni d'un ordre bien fondé $\prec$, de sorte que

$$s \to_R t \quad \Longrightarrow \quad \phi(s) \succ \phi(t)$$

  Exemples : $K = \mathbb{N}$, ou un produit lexicographique d'ordres bien fondés.
- On définit un ordre de réduction : ordres polynomiaux, LPO, etc.

Ordre de réduction

Définition : ordre de réécriture

Un ordre de réécriture sur $\mathcal{T}(\mathcal{F}, \mathcal{X})$ est un ordre strict $>$ tel que

1. $>$ est compatible avec les symboles de fonction : si $s_1 > s_2$ alors $f(t_1, \ldots, t_{i-1}, s_1, t_{i+1}, \ldots, t_n) > f(t_1, \ldots, t_{i-1}, s_2, t_{i+1}, \ldots, t_n)$.
2. $>$ est compatible avec les substitutions : si $s_1 > s_2$ alors pour toute substitution σ, $\sigma s_1 > \sigma s_2$.

Définition : un ordre de réduction est un ordre de réécriture bien fondé.

Exemple : $s > t$ ssi $|s| > |t|$ et pour toute variable x, $|s|_x > |t|_x$

Ordre polynomial

Idée : plonger les termes dans les entiers positifs.

Définition : interprétation polynomiale
Une interprétation polynomiale est une application de $\mathcal{T}(\mathcal{F}, \mathcal{X})$ dans $\mathbb{N} \setminus \{0\}$ telle que tout symbole de fonction f d'arité n est interprété par une fonction polynôme de $\mathbb{N}[X_1, \ldots, X_n]$.

Définition : interprétation polynomiale monotone
Tout symbole de fonction f est interprété par une fonction polynôme strictement croissante en chacun de ses arguments.

Ordre polynomial

Proposition

Une interprétation polynomiale monotone fournit un ordre de réduction sur $\mathcal{T}(\mathcal{F}, \mathcal{X})$.

Proposition

Savoir si $P > Q$ pour un ensemble infini d'entiers est indécidable.

Proposition

Si la terminaison d'un système de réécriture est garantie par un ordre polynomial, alors il existe une constante $c > 0$ telle que toute suite de réductions partant du terme t est de longueur bornée par $2^{2^{c|t|}}$.

Lexicographic Path Order

On se donne un ordre total $>$ sur les éléments de $\mathcal{F}$.

Définition : LPO

Si $s = f(s_1, \ldots, s_n)$ et $t = g(t_1, \ldots, t_m)$, on a $s >_{lpo} t$ si

1. $f > g$ et $s >_{lpo} t_i$ pour tout $i \in \{1, \ldots, m\}$, ou
2. $f = g$ et $s >_{lpo} t_i$ pour tout $i \in \{1, \ldots, m\}$ et $s_1 \ldots s_n >_{lpo}^{lex} t_1 \ldots t_m$, ou
3. il existe $i \in \{1, \ldots, m\}$ tel que $s_i = t$ ou $s_i >_{lpo} t$,

avec $s_1 \ldots s_n >_{lpo}^{lex} t_1 \ldots t_m$ si

1. $n \neq 0$ et $m = 0$, ou
2. $s_1 >_{lpo} t_1$, ou
3. $s_1 = t_1$ et $s_2 \ldots s_n >_{lpo}^{lex} t_2 \ldots t_m$.

LPO : exemple

Soit R le système de réécriture suivant :

$$R = \begin{cases} (1) & 0 + x \to x \\ (2) & S(x) + y \to S(x + y) \end{cases}$$

Prouvons que R termine à l'aide d'un LPO :
On choisit $>$ tel que $+ > S$.
On montre que chaque règle respecte l'ordre :

1. $0 + x >_{lpo} x$ par le cas (3) du LPO.
2. $S(x) + y >_{lpo} S(x + y)$ par le cas (1) du LPO car
   - $+ > S$ et
   - $S(x) + y >_{lpo} x + y$ par le cas (2) du LPO car
     - $+ = +$ et
     - $S(x) + y >_{lpo} x$ et
     - $S(x) + y >_{lpo} y$ et
     - $S(x), y >_{lpo}^{lex} x, y$ car $S(x) >_{lpo} x$ par le cas (3) de $>_{lpo}^{lex}$.

LPO : propriétés

Proposition

Pour un LPO donné, savoir si $s > t$ est décidable en temps polynomial en la taille de s et t.

Proposition

La question de savoir si la terminaison d'un système de réécriture donné peut être prouvée grâce à un LPO est NP-complète.

Exercice

Donner une définition équationnelle des files comprenant au moins la file vide (nil) et les opérations suivantes :

- consultation de l'élément de tête (head),
- retrait de l'élément de tête (pop),
- ajout d'un élément en queue (add).

Sur cette définition, prouver que

$$\mathrm{pop}(\mathrm{add}(\mathrm{head}(\mathrm{add}(a, \mathrm{add}(b, \mathrm{nil}))), \mathrm{add}(a, \mathrm{nil}))) \not\approx_E \mathrm{pop}(\mathrm{add}(a, \mathrm{add}(b, \mathrm{nil})))$$

Solution

Une théorie équationnelle possible pour les files est :

$$\begin{aligned}
&\mathrm{head}(\mathrm{add}(x, \mathrm{nil})) \approx x \\
&\mathrm{head}(\mathrm{add}(x, \mathrm{add}(y, z)) \approx \mathrm{head}(\mathrm{add}(y, z)) \\
&\mathrm{pop}(\mathrm{add}(x, \mathrm{nil})) \approx \mathrm{nil} \\
&\mathrm{pop}(\mathrm{add}(x, \mathrm{add}(y, z))) \approx \mathrm{add}(x, \mathrm{pop}(\mathrm{add}(y, z)))
\end{aligned}$$

Pour prouver que

$$\mathrm{pop}(\mathrm{add}(\mathrm{head}(\mathrm{add}(a, \mathrm{add}(b, \mathrm{nil}))), \mathrm{add}(a, \mathrm{nil}))) \not\approx_E \mathrm{pop}(\mathrm{add}(a, \mathrm{add}(b, \mathrm{nil})))$$

il faut...

Solution

1. orienter E en R :

$$\begin{aligned}
&\mathrm{head}(\mathrm{add}(x, \mathrm{nil})) \to x\\
&\mathrm{head}(\mathrm{add}(x, \mathrm{add}(y, z)) \to \mathrm{head}(\mathrm{add}(y, z))\\
&\mathrm{pop}(\mathrm{add}(x, \mathrm{nil})) \to \mathrm{nil}\\
&\mathrm{pop}(\mathrm{add}(x, \mathrm{add}(y, z))) \to \mathrm{add}(x, \mathrm{pop}(\mathrm{add}(y, z)))
\end{aligned}$$

2. Prouver que R termine (ordre LPO et précédence pop $>$ add).
3. Prouver que R est confluent. C'est le cas puisqu'il n'y a pas de paires critiques.
4. Montrer qu'il n'existe pas de terme t tel que
$\mathrm{pop}(\mathrm{add}(\mathrm{head}(\mathrm{add}(a, \mathrm{add}(b, \mathrm{nil}))), \mathrm{add}(a, \mathrm{nil}))) \to_R^* t$
et
$\mathrm{pop}(\mathrm{add}(a, \mathrm{add}(b, \mathrm{nil}))) \to_R^* t$.
C'est le cas puisque

$$\mathrm{pop}(\mathrm{add}(\mathrm{head}(\mathrm{add}(a, \mathrm{add}(b, \mathrm{nil}))), \mathrm{add}(a, \mathrm{nil}))) \to_R^* \mathrm{add}(b, \mathrm{nil})$$

et

$$\mathrm{pop}(\mathrm{add}(a, \mathrm{add}(b, \mathrm{nil}))) \to_R^* \mathrm{add}(a, \mathrm{nil})$$

Réécriture et outils de vérification

- Beaucoup de problèmes indécidables : pas de procédures de décision
- Outils de vérification basés (au moins en partie) sur la réécriture et les spécifications équationnelles
  - VERSA (algèbres de processus temps-réel)
  - Larch Prover (en particulier *hardware*)
  - Maude
- Dans les assistants de preuve en général : la réécriture est appliquée manuellement, ou automatiquement de façon heuristique