Termination

Motivations

Termination is essential to proof correctness of programs.

But

Termination is an indecidable property.

Indecidability of termination

Let $a_1, a_2, a_3, \ldots$ be an enumeration of all the algorithms on integers. We define the following functions :

$\texttt{end}(i,n) \equiv 1$ if $a_i(n)$ terminates
$\texttt{end}(i,n) \equiv 0$ if $a_i(n)$ $\neg$ terminates

$\texttt{Diag}(i) \equiv$ if $\texttt{end}(i,i) = 1$ then loop
else stop

For every i, $\texttt{Diag}(i)$ terminates iff $a_i(i)$ does not terminate.

But Diag is al algorithm, so that $\exists a_j$ s.t. $\texttt{Diag} = a_j$. We then have $\texttt{Diag}(j)$ terminates iff $a_j(j)$ terminates, that is

$a_j(j)$ terminates iff $a_j(j)$ does not terminate.

Which is the error in the proof ? The existence of the fonction end.

Termination of a very simple system

$$f(g(x), y) \to f(y, y)$$

is not even trivial !

$$f(g(a), g(a)) \to f(g(a), g(a)) \to f(g(a), g(a)) \to \ldots$$

The case of typed lambda calculus

- Not very expressive (extended polynomials, total).
- Termination is not trivial.
- Many different proofs in the literature.
- Even the simplest (arithmetical) proof is subtle.

Strong normalization of typed lambda calculus

Théorème : Every typed term is normalising, i.e. if $\Gamma \vdash t : A$, then $t \in SN_\beta$.

Some General Remarks

- Typing is stable by (typed) substitution : if t is of typed A, and x, u are of typed B, then $t\{x/u\}$ is of typed A.
- SN is not stable by substitution. Example : $x\ x \in SN$, $\lambda y.y\ y \in SN$, but $(x\ x)\{x/\lambda y.y\ y\} = \Delta\ \Delta \notin SN$.
- $t \in SN$
  iff there is no infinite reduction sequence starting at t
  iff every reduction sequence starting at t is finite
  iff $\forall t'\ [(t \to_\beta\ t')$ implies $t' \in SN]$.
- A particular case : $t \in SN$ if t is in normal form.
- The standard order between types is given by
  $A < A \to B$ and $B < A \to B$.
  Thus base types are minimal with respect this order.

- $u \in SN$ iff $\lambda y.u \in SN$.
- $u_1 \dots n_n \in SN$ iff $x\ u_1 \dots u_n \in SN$.
- Given $t \in SN$ we define $\mu(t)$ as the maximal lenght of a reduction sequence starting at t. We observe that $t \to\ t'$ implies $\mu(t') < \mu(t)$.

Third Proof of the SN property

Lemme : If t and u are typed and SN, then $t\{x/u\}$ is SN.

Proof. By induction on $\langle type(u), \mu(t), size(t)\rangle$.

- The base case $\langle$ base type, 0, 1 $\rangle$ is trivial.
- Case $t = \lambda y.v$ is straightforward ($size(t)$ strictly decreases).
- Case $t = y\ \vec{c_n}$ with $x \neq y$ is straightforward ($\mu(t)$ decreases and $size(t)$ strictly decreases.).
- Case $t = x$. We have $x\{x/u\} = u \in SN$ by hypothesis.
- Case $t = x\ b\ \vec{c_n}$. By i.h. $B = b\{x/u\}$ and $C_i = c_i\{x/u\}$ are SN. We want to show that $u\ B\ \vec{C_n}$ is SN. It is sufficient to show that all its reducts are SN. We reason by induction on $\mu(u) + \mu(B) + \Sigma_i\ \mu(C_i)$. The reducts are

- $u'\ B\ \vec{C_n}$, where $u \rightarrow\ u'$. Apply the i.h.
- $u\ B'\ \vec{C_n}$, where $B \rightarrow\ B'$. Apply the i.h.
- $u\ B\ C_1 \ldots C_i' \ldots C_n$, where $C_i \rightarrow\ C_i'$. Apply the i.h.
- $u'\{y/B\}\vec{C_n}$, where $u = \lambda y.u'$. But $u'\{y/B\}\vec{C_n} = (z\vec{C_n})\{z/u'\{y/B\}\}$ and $type(u'\{y/B\}) < type(u)$. We thus conclude by the i.h. since both $z\vec{C_n}$ and $u'\{y/B\}$ are typed and SN.

- Case $t = (\lambda z.b)\ c\ \vec{d}$. By i.h. $B = b\{x/u\}$ and $C = c\{x/u\}$ and $D_i = d_i\{x/u\}$ are SN. Suppose $t\{x/u\} = (\lambda z.B)\ C\ \vec{D_n} \notin SN$. Then $B\{z/C\}\ \vec{D_n} \notin SN$. But $B\{z/C\}\ \vec{D_n} = (b\{z/c\}\vec{d_n})\{x/u\}$ and $\mu(b\{z/c\}\vec{d_n}) < \mu(t)$. Thus $B\{z/C\}\ \vec{D_n} \in SN$ by the i.h. Contradiction. Thus $t\{x/u\} = (\lambda z.B)\ C\ \vec{D_n} \in SN$.

▪

Théorème : If t is typable, then t is SN.

Proof. By induction on the structure of t.

- Case $t = x$ is trivial.
- Case $t = \lambda y.u$ holds by the i.h.
- For the case $t = u\ v$ use the fact that $t = (z\ v)\{z/u\}$ and apply previous lemma.

▪

How to model ?

Rewrite systems :

$$\begin{array}{lcl} 0 + y & \rightarrow & y \\ s(x) + y & \rightarrow & s(x + y) \\ 0 * y & \rightarrow & 0 \\ s(x) * y & \rightarrow & (x * y) + y \end{array}$$

Rewrite reductions :

$$s(s(s(0))) * s(s(0)) \rightarrow^* s(s(s(s(s(0))))))$$

Rewrite Systems

A signature Σ is a non-empty set of function symbols s.t. every $f \in \Sigma$ has an arity n. We write f/n if the symbol f has arity n.

Let $\mathcal{X}$ be a set of variables and let Σ be a signature. The set $\mathcal{T}(\mathcal{X}, \Sigma)$ of terms over $\mathcal{X}$ and Σ is defined as follows :

- If $x \in \mathcal{X}$, then $x \in \mathcal{T}(\mathcal{X}, \Sigma)$
- If $f/n \in \Sigma$, and $t_1, \ldots, t_n \in \mathcal{T}(\mathcal{X}, \Sigma)$, then $f(t_1, \ldots, t_n) \in \mathcal{T}(\mathcal{X}, \Sigma)$

We write $Var(t)$ for the set of variables of the term t. A term is t is closed if $Var(t) = \emptyset$.

Rewrite Systems

Rewrite rule : a pair $l \to r$ s.t. $\begin{cases} Var(r) \subseteq Var(l) \\ l \text{ is not a variable} \end{cases}$

Rewrite system : a set of rewrite rules.

Rewrite step : A term s $\mathcal{R}$-rewrites to t iff $s \to_{\mathcal{R}} t$ can be derived from the following system :

$$\frac{l \to r \in \mathcal{R} \text{ and } \sigma \text{ is a subst.}}{\sigma(l) \to_{\mathcal{R}} \sigma(r)} \text{ (head)} \qquad \frac{s' \to_{\mathcal{R}} t'}{u[s'] \to_{\mathcal{R}} u[t']} \text{ (context)}$$

Exemple : Consider the following rewrite system

$$\mathcal{R} = \begin{cases} f(x,x) & \to & c \\ a & \to & b \\ f(x,b) & \to & d \end{cases}$$

We construct the following rewrite steps

$f(a,a) \to f(a,b) \to f(b,b) \to c$

$f(a,a) \to f(a,b) \to d$

Basic vocabulary

- A term s is an $\mathcal{R}$-redex iff $s = \theta(l)$ for some $l \to r \in \mathcal{R}$ and some substitution θ.
- A term s is an $\mathcal{R}$-contractum iff $s = \theta(r)$ for some $l \to r \in \mathcal{R}$ and some substitution θ.
- A term t is $\mathcal{R}$-reducible iff there exists s s.t. $t \to_{\mathcal{R}} s$.
  Exemple : The term $f(a,a)$ is reducible, the term c is not reducible.
- A term t is in $\mathcal{R}$-normal form iff it is no $\mathcal{R}$-reducible.
- A term s is a $\mathcal{R}$-normal form of t iff $t \to_{\mathcal{R}}^* s$ and s is in $\mathcal{R}$-normal form.
  Exemple : The terms c and d are normal forms of $f(a,a)$.

Termination notions

- The system $\mathcal{R}$ is weakly normalising (WN) iff every element has at least one $\mathcal{R}$-normal form.
- The system $\mathcal{R}$ terminates or is strongly normalising (SN) or noetherien or well-founded (WF) iff every $\mathcal{R}$-reduction sequence starting at s is finite. We note $s \in SN_{\mathcal{R}}$.

Weak vs strong normalisation

$$\mathcal{R} = \begin{cases} f(a) & \rightarrow & c \\ f(x) & \rightarrow & f(a) \end{cases}$$

The system is weakly normalising but not strongly normalising :

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

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

Techniques to show termination

- Reduction orders
  - Particular case : interpretations
  - Example of interpretation : polynomial orders
- Useful orders :
  - Lexicographique order
  - Multi-set order
- Simplification orders
  - General result
  - Example : RPO
- Combination of orders :
  - Motivations
  - Postponment

  - Projection/simulation
- Dependency pairs

Termination by reduction orders

Pre-order : reflexive and transitive relation.

Partial order : reflexive, and thus antisymmetric and transitive relation.

Strict order : ireflexive and transitive relation.

A strict order $\succ$ over a signature Σ is a reduction order iff

1. Each symbol $f \in \Sigma$ is monotone w.r.t $\succ$
2. $\succ$ is stable by substitution
3. $\succ$ is WF

Why reduction orders are important ?

Théorème : A rewriting system $\mathcal{R}$ terminates iff there exists a reduction order $\succ$ s.t. $l \succ r$ for every rewriting rule $l \to r \in \mathcal{R}$.

How does it work ?

Does $\mathcal{R}$ terminate ?

$$\mathcal{R} = \begin{cases} \mathtt{por}(x, \mathtt{t}) & \to & \mathtt{t} \\ \mathtt{por}(\mathtt{t}, x) & \to & \mathtt{t} \end{cases}$$

The number of symbol decreases....

"$s \succ t$ iff $|s| > |t|$" is not a reduction order :

$|\mathtt{por}(x, \mathtt{por}(y, \mathtt{t}))| > |\mathtt{por}(y, y)|$ but
$|\mathtt{por}(\mathtt{t}, \mathtt{por}(\mathtt{por}(\mathtt{t}, \mathtt{t}), \mathtt{t}))| \not> |\mathtt{por}(\mathtt{por}(\mathtt{t}, \mathtt{t}), \mathtt{por}($

"$s \succ t$ iff $|s| > |t|$ and for every variable x we have $|s|_x \geq |t|_x$" is a reduction order.

Interpretation as particular case of reduction order

The reduction order is first defined on the interpretation of terms, and not directly on terms.

Let $\succ_{\mathcal{A}}$ be a WF strict order over the domain of a Σ-algebra $\mathcal{A}$.

$\succ$ is given by :

$s \succ t$ iff $\Phi(s) \succ_{\mathcal{A}} \Phi(t)$ for all homomorphisms $\Phi : \mathcal{T}(\mathcal{X}, \Sigma) \to \mathcal{A}$

Théorème : If for every $f \in \Sigma$, the interpretation $f^{\mathcal{A}}$ is monotone w.r.t. $\succ_{\mathcal{A}}$, then $\succ$ is a reduction order.

Example : polynomial orders

with n indeterminates and coefficients in $\mathbb{N}$

- A domain which is a subset of $\mathbb{N}$
- A polynomial P_f for every $f/n \in \Sigma$, there is s.t. $f^{\mathcal{P}_{\mathbb{N}}}(a_1, \ldots, a_n) = P_f(a_1, \ldots, a_n)$.

Exemple : Let $\Sigma = \{f/2, g/2, a/0\}$. Consider the morphism Φ given polynomials $P_f(x, y) = x.y$ and $P_g(x, y) = 2.x + y + 1$ and $P_a = 2$. Then we have $\Phi(f(a, g(a, a))) = 2.(2.2 + 2 + 1)$.

Problem : Polynomials are not necessarily monotone, for example if $P_f(X, Y) = X^2$ we have $3 > 2$ but $P_f(2, 3) = 4 \not> 4 = P_f(2, 2)$.

Towards a polynomial order as interpretation

A polynomial P is completely monotone iff it depends on all its indeterminates.

Exemple : $P(x, y) = 3.x + y + 2$ and $P(x, y) = x.y$ are all completely monotone.

Théorème : Let $\mathcal{P}_{\mathbb{N}}$ be a polynomail Σ-algebra. If every $f^{\mathcal{P}_{\mathbb{N}}}$ is a completely monotone polynomial, then the order $\succ$ associated to $\succ_{\mathcal{P}_{\mathbb{N}}}$ is a reduction order.

How does it work ?

Does $\mathcal{R}$ terminate ?

$$\mathcal{R} = \left\{ \; f(x, g(y, z)) \quad \rightarrow \quad g(f(x, y), f(x, z)) \right.$$

1. Define the domain : $\mathbb{N} - \{0, 1\}$.
2. Define a polynomial for every function symbol : $P_f(x, y) = x.y$ et $P_g(x, y) = 2.x + y + 1$.
3. Prove that $f(x, g(y, z)) \succ g(f(x, y), f(x, z))$: Prove $\sigma(x).(2.\sigma(y) + \sigma(z) + 1) \succ_{\mathcal{P}_{\mathbb{N}}} 2.\sigma(x).\sigma(y) + \sigma(x).\sigma(z) + 1$ for every $\sigma(x), \sigma(y), \sigma(z) \neq 0, 1$.

Lexicographic order - particular case

Let $(A_1, >_{A_1})$ and $(A_2, >_{A_2})$ be two strict ordered sets.

$$(x, y) >_{lex} (x', y') \text{ iff } (x >_{A_1} x') \text{ or } (x = x' \text{ and } y >_{A_2} y')$$

Exemple :

$$(4, "abc") >_{lex} (3, "abc") >_{lex} (2, "abcde") >_{lex}$$
$$(2, "bcde") >_{lex} (2, "e") >_{lex} (1, "e") >_{lex} (0, \epsilon)$$

Lexicographic order - General case

If every $>_{A_i}$ is a strict order over the set $\mathcal{A}_i$, then $>_{lex}$ is a strict order over $\mathcal{A}_1 \times \ldots \times \mathcal{A}_n$ defined as follows :

$$(x_1, \ldots, x_n) >_{lex} (x'_1, \ldots, x'_n) \text{ iff } \quad \exists 1 \leq j \leq n$$
$$(x_j >_{A_j} x'_j \text{ and } \forall 1 \leq i < j \;\; x_i = x'_i)$$

Théorème : Every order $>_{A_i}$ over $\mathcal{A}_i$ is well-founded iff the lexicographic order $>_{lex}$ over $\mathcal{A}_1 \times \ldots \times \mathcal{A}_n$ is well-founded.

How does it work ?

Does the following program terminate ?

ackerman(0,n)	$\rightarrow$	n+1
ackerman(m+1,0)	$\rightarrow$	ackerman(m,1)
ackerman(m+1,n+1)	$\rightarrow$	ackerman(m,ackerman(m+1,n))

Proof. We show that ackerman(m, n) terminates by induction on (m, n) w.r.t. the lexicographic order. ▪

Another example ?

Does the following program terminate ?

$$\begin{array}{lcl} f(f(x)) & \rightarrow & g(f(x)) \\ g(g(x)) & \rightarrow & f(x) \end{array}$$

Proof. We show that $t \rightarrow u$ implies $(|t|, |t|_f) >_{lex} (|u|, |u|_f)$.

▪

Multi-set order

A multi-set over a set $\mathcal{A}$ is a function $\mathcal{M} : \mathcal{A} \rightarrow \mathbb{N}$. It is finite if $\mathcal{M}(x) > 0$ only for a finite number of elements of $\mathcal{A}$.

Exemple : $\{\!\{a, a, b\}\!\}$.

Let $\mathcal{M}$ and $\mathcal{N}$ be two multi-sets. The multi-set union is defined by $\mathcal{M} \uplus \mathcal{N}(a) = \mathcal{M}(a) + \mathcal{N}(a)$.

Multi-set order

Let $\succ$ a strict order. The associated relation $\succ_{mul}$ is given by the transitive closure of the relation $\succ_{mul}$:

$\mathcal{M} \uplus \{\!\{x\}\!\} \succ_{mul} \mathcal{M} \uplus \{\!\{y_1, \ldots, y_n\}\!\}$, where $n \geq 0$ and $\forall i, x \succ y_i$.

Exemple : $\{\!\{5,3,1,1\}\!\} \succ_{mul} \{\!\{4,3,3,1\}\!\}$.

Since $\{\!\{5,3,1,1\}\!\} \succ_{mul} \{\!\{4,3,3,1,1\}\!\} \succ_{mul} \{\!\{4,3,3,1\}\!\}$

Théorème : Let $\succ$ be a strict order over $\mathcal{A}$, then $\succ$ is WF iff $\succ_{mul}$ is WF.

How does it work ?

A rich but bored man decides to have fun every day with his money (in euros) in the following way :

- either he throw a coin in the fountain,
- or he changes a banknote into an arbitrary amount of coins.

Show that the man necessarily becomes poor.

- Represent the initial amount of money by a multi-set.
- Represent the daily activity of the man by a decreasing order on multi-sets.

Other known examples

- Hercules defeats Hydra
- Cut elimination in Gentzen style systems
- Amoebae reproduction
- Recursive Path Orderings

Simplification orders

A simplification order over $\mathcal{T}(\mathcal{X}, \Sigma)$ is an order $\succ$ s.t.

1. All the symbols of Σ are monotone w.r.t $\succ$
2. $\succ$ is stable by substitution
3. $t \triangleright u$ implies $t \succ u$

Example : embedding

The relation $s \trianglerighteq_{emb} t$ holds iff one of the following cases hold
- s and t are the same variable
- $s = f(s_1, \ldots, s_n)$ and $t = f(t_1, \ldots, t_n)$ and $\forall i\ s_i \trianglerighteq_{emb} t_i$
- $s = f(s_1, \ldots, s_n)$ and there is j s.t. $s_j \trianglerighteq_{emb} t$

Exemple : $f(f(h(h(a)), h(x)), f(h(x), a)) \triangleright_{emb} f(f(a, x), x)$

Termination by simplification orders

Lemme : The relation $\triangleright_{emb}$ is contained in every simplication order.

Lemme : If $\succ$ is a simplication order, then it is a reduction order (and thus WF).

Proof. Uses the famous Kruskal's Theorem. ▪

And the inverse ?

Let $\mathcal{R} = f(f(x)) \to f(g(f(x)))$.

The system $\mathcal{R}$ terminates (exercice).

Thus $\to^+_{\mathcal{R}}$ is a reduction order.

Suppose that $\to^+_{\mathcal{R}}$ is also a simplification order.

Then $f(g(f(x))) \trianglerighteq_{emb} f(f(x))$ implies
$f(g(f(x))) \to^+ f(f(x)) \to^+ f(g(f(x))) \ldots$.

Contradiction with the termination of $\mathcal{R}$.

Exa ing

reflexive and transitive

Let $\succsim_\Sigma$ be a pre-order over Σ. We associate to every symbol $f \in \Sigma$ a status in $\{LEX, MUL\}$ s.t. if $f \sim g$, then
- f and g have the same status,
- and if the status is LEX, then f and g have the same arity.

We note $f \in \Sigma_{LEX}$ to indicate that $f \in \Sigma$ has LEX status.

The order $\succ_{rpo}$

Let $\succsim_\Sigma$ be a pre-order over a signature Σ such that $\succ_\Sigma$ is WF.
The RPO is given by $s \succ_{rpo} t$ iff

1. [sub-term] $s = f(s_1, \ldots, s_n)$ and $\exists i$ s.t. $s_i \succ_{rpo} t$ or $s_i = t$ or
2. [Two symbols] $s = f(s_1, \ldots, s_n)$, $t = g(t_1, \ldots, t_m)$ and one of the following conditions is verified
   (a) [precedence] $f \succ_\Sigma g$ and for all j, $s \succ_{rpo} t_j$
   (b) [multi-set] $f \sim_\Sigma g$ have MUL status and $\{\!\{s_1, \ldots, s_n\}\!\} (\succ_{rpo})_{mul} \{\!\{t_1, \ldots, t_m\}\!\}$.
   (c) [lexicographic] $f \sim_\Sigma g$ have LEX status and $(s_1, \ldots, s_n)(\succ_{rpo})_{lex}(t_1, \ldots, t_m)$ and for all j, $s \succ_{rpo} t_j$

Alternative definition of RPO

$$\frac{\exists i \ (s_i \succ_{rpo} t \text{ or } s_i = t)}{f(s_1, \ldots, s_n) \succ_{rpo} t} \text{ [1]}$$

$$\frac{f \succ_\Sigma g \text{ and } \forall j \ s \succ_{rpo} t_j}{s = f(s_1, \ldots, s_n) \succ_{rpo} g(t_1, \ldots, t_m)} \text{ [2.a]}$$

$$\frac{f \sim_\Sigma g \in \Sigma_{MUL} \text{ and } \{\!\{s_1, \ldots, s_n\}\!\} \ (\succ_{rpo})_{mul} \ \{\!\{t_1, \ldots, t_m\}\!\}}{s = f(s_1, \ldots, s_n) \succ_{rpo} g(t_1, \ldots, t_m) = t} \text{ [2.b]}$$

$$\frac{f \sim_\Sigma g \in \Sigma_{LEX} \text{ and } (s_1, \ldots, s_n)(\succ_{rpo})_{lex} \ (t_1, \ldots, t_m) \text{ and } \forall j \ s \succ_{rpo} t_j}{s = f(s_1, \ldots, s_n) \ \succ_{rpo} g(t_1, \ldots, t_m) = t} \text{ [2}$$

Remarks

- Is this definition well-founded ?
- Can we avoid condition $s \succ_{rpo} t_j$ in case LEX [2.c] ?
  We would have that $a \succ_\Sigma a'$ implies $f(a, b) \succ_{rpo} f(a', f(a, b))$
- If all the symbols are LEX, the order is known as LPO.
- If all the symbols are MUL, the order is known as MPO.

Property of $\succ_{rpo}$

Théorème : If the pre-order $\succsim_\Sigma$ is WF, then its associated relation $\succ_{rpo}$ is also WF.

The RPO was extended to the higher-order case by Jouannaud and Rubio.

Simple example

$$\mathcal{R}\begin{cases} 0+y & \to_{r1} & y \\ s(x)+y & \to_{r2} & s(x+y) \\ 0*y & \to_{r3} & 0 \\ s(x)*y & \to_{r4} & (x*y)+y \end{cases}$$

- Define $* \succ_\Sigma + \succ_\Sigma s \succ_\Sigma 0$, all with MUL (or LEX) status.
- Show that $l >_{rpo} r$ for every rule $l \to r \in \mathcal{R}$.

Thus for example for rule $s(x) * y \to_{r4} (x * y) + y$

$$\dfrac{* \succ_\Sigma + \qquad \dfrac{* \sim_\Sigma * \qquad \dfrac{\dfrac{x = x}{s(x) \succ_{rpo} x}}{\{\!\{s(x), y\}\!\}\ (\succ_{rpo})_{mul}\ \{\!\{x, y\}\!\}}}{s(x) * y \succ_{rpo} (x * y)} \qquad \dfrac{y = y}{s(x) * y \succ_{rpo} y}}{s(x) * y \succ_{rpo} (x * y) + y}$$

Famous example : cut elimination in intuitionistic logic

$$\begin{array}{lcl} x[x/t] & \to & t \\ y[x/t] & \to & y \\ (\lambda z.u)[x/t] & \to & \lambda z.u[x/t] \\ (y \texttt{ of } u \texttt{ is } w \texttt{ in } v)[x/t] & \to & y \texttt{ of } u[x/t] \texttt{ is } w \texttt{ in } v[x/t] \\ (x \texttt{ of } u \texttt{ is } w \texttt{ in } v)[x/y] & \to & y \texttt{ of } u[x/y] \texttt{ is } w \texttt{ in } v[x/y] \\ (x \texttt{ of } u \texttt{ is } w \texttt{ in } v)[x/\lambda z.t] & \to & v[x/\lambda z.t][w/t[z/u[x/\lambda z.t]]] \\ (x \texttt{ of } u \texttt{ is } w \texttt{ in } v)[x/x' \texttt{ of } t' \texttt{ is } z \texttt{ in } t] & \to & x' \texttt{ of } t' \texttt{ is } z \texttt{ in } ((x \texttt{ of } u \texttt{ is } w \texttt{ in } v)[x/t]) \end{array}$$

Combining orders

Suppose two SN relations $\mathcal{R}_1$ and $\mathcal{R}_2$. What about $\mathcal{R}_1 \cup \mathcal{R}_2$?

Counter-example by Toyama :

$\mathcal{R}_1 = f(x,a,b) \to f(x,x,x)$

$\begin{cases} g(x,y) \to x \end{cases}$

which do not share symbols !

The systems $\mathcal{R}_1$ and $\mathcal{R}_2$ are SN but $\mathcal{R}_1 \cup \mathcal{R}_2$ is not :

$$f(g(a,b),g(a,b),g(a,b)) \to_{\mathcal{R}_2} f(g(a,b),a,g(a,b)) \to_{\mathcal{R}_2}$$
$$f(g(a,b),a,b) \to_{\mathcal{R}_1} f(g(a,b),g(a,b),g(a,b)) \to \ldots$$

Termination by postponement

A relation $\mathcal{R}$ can be postponed w.r.t. a relation $\mathcal{S}$ iff

$$\text{for all } s,t,u \text{ s.t. } \quad s \to_{\mathcal{R}} t \to_{\mathcal{S}} u$$

$$\text{there is } v \quad s \to_{\mathcal{S}}^{+} v \to_{\mathcal{R}\cup\mathcal{S}}^{*} u$$

Théorème : Let $\mathcal{R}$ and $\mathcal{S}$ be two WF relations s.t. $\mathcal{R}$ can be postponed w.r.t. $\mathcal{S}$. Then the relation $\mathcal{R} \cup \mathcal{S}$ is WF.

Corollaire : If $\mathcal{S}$ is WF, then $\mathcal{S} \cup \triangleright$ is WF.

First example

Consider the simply typed λ-calculus with the following rules :

$$\begin{array}{lll} (\lambda x.M)N & \to_\beta & M\{x/N\} \\ \lambda x.M\ x & \to_\eta & M \\ M & \to_\Omega & \star \end{array}$$

Let $\mathcal{R} = \eta \cup \Omega$ and $\mathcal{S} = \beta$. Now,

- Show that $\eta \cup \Omega$ can be postponed w.r.t. β.
- Since β is SN, then conclude that $\beta \cup \eta \cup \Omega$ is SN.

Second example

Consider λ-calculus with the following rules :

$$\begin{array}{lll} (\lambda x.M)N & \to_\beta & M\{x/N\} \\ \pi_1(\langle M,N\rangle) & \to_{\pi_1} & M \\ \pi_2(\langle M,N\rangle) & \to_{\pi_2} & N \end{array}$$

Let $\mathcal{R} = \pi_1 \cup \pi_2$ and $\mathcal{S} = \beta$. Now,

- Show that $\pi_1 \cup \pi_2$ can be postponed w.r.t. β.
- Conclude that if β is SN, then $\beta \cup \pi_1 \cup \pi_2$ is SN.

Termination by projection/simulation

Théorème : Let $\mathcal{R}_1, \mathcal{R}_2$ be two relations over O s.t.

1. $\mathcal{R}_2$ terminates
2. There is a simulation $\mathcal{T} : O \to O'$ and a relation $\mathcal{S}$ over O' s.t.
   - $a \to_{\mathcal{R}_1} b$ implies $\mathcal{T}(a) \to_{\mathcal{S}}^{+} \mathcal{T}(b)$,
   - $a \to_{\mathcal{R}_2} b$ implies $\mathcal{T}(a) \to_{\mathcal{S}}^{*} \mathcal{T}(b)$.

Then, If $\mathcal{S}$ terminates, $(\mathcal{R}_1 \cup \mathcal{R}_2)$ termines also.

(Famous) Example

Consider the simply typed extensional λ-calculus

$$
\begin{array}{lll}
(\lambda x.M)\ N & \to_{\beta} & M\{x/N\} \\
\pi_1\langle M, N\rangle & \to_{\pi_1} & M \\
\pi_2\langle M, N\rangle & \to_{\pi_2} & N \\
M & \to_{\eta_{exp}} & \lambda x.Mx \quad \text{if} \begin{cases} M \text{ is of functional type} \\ M \text{ is not a } \lambda\text{-abstraction} \\ M \text{ is not applied in } C[M] \end{cases} \\
M & \to_{sp_{exp}} & \langle \pi_1(M), \pi_2(M)\rangle \quad \text{if} \begin{cases} M \text{ is of product type} \\ M \text{ is not a pair} \\ M \text{ is not projected in } C[M] \end{cases}
\end{array}
$$

Thus for exam[illegible] $(A \times B) \to (C \to D)$, then

the variable is applied

$$I\ x\ z \to_{\beta}\ x\ z\ \to_{sp_{exp}}\ x\ \langle \pi_1(z), \pi_2(z)\rangle \to_{\eta_{exp}}\ \lambda y.(x\ \langle \pi_1(z), \pi_2(z)\rangle)\ y$$

Let $\mathcal{R}_1 = \beta \cup \pi_1 \cup \pi_2$ and $\mathcal{R}_2 = \eta_{exp} \cup sp_{exp}$. Now,

- Show that $\beta \cup \pi_1 \cup \pi_2$ is terminating (done).
- Show that $\eta \cup sp$ is confluent and SN.
- Define $\mathcal{T}(t)$ as the $\eta \cup sp$-normal form of t.
- Show that $t \to_{\beta\cup\pi_1\cup\pi_2}\ t'$ implies $\mathcal{T}(t) \to^{+}_{\beta\cup\pi_1\cup\pi_2} \mathcal{T}(t')$
- Conclude that all the system is SN.

Termination by Depedency Pairs

- The technique is due to Aarts and Giesl.
- The order does not decrease for every step, but for the critical ones.
- The technique is very suitable for functional programming.
- It was extended to higher-order by Sakai and Kusakari.
- It was extended to abstract rewriting by Lengrand.

Termination by Depedency Pairs

Let $\mathcal{R}$ be a rewriting system.

The set of defined symbols of $\mathcal{R}$ is given by :

$\mathcal{D} = \{f \mid f(l_1, \ldots, l_n) \to\ r \in \mathcal{R}\}$.

The set of constructor symbols of $\mathcal{R}$ is given by : $\mathcal{C} = \Sigma \setminus \mathcal{D}$.

Exemple :

$$
\mathcal{R} = \begin{cases} x + 0 & \to & x \\ x + s(y) & \to & s(x + y) \\ x * 0 & \to & 0 \\ x * s(y) & \to & x * y + x \end{cases}
$$

A dependency pair of a rule $l \to r \in \mathcal{R}$ is a pair of the form $\langle l, f(\vec{s})\rangle$, where $f(\vec{s})$ is a subterm of r and $f \in \mathcal{D}$.

The set $PD(\mathcal{R})$ of dependency pairs of a system $\mathcal{R}$ is the union of the dependency pairs of all the rules of $\mathcal{R}$.

Exemple : The dependency pairs of $\mathcal{R}$ are

$$\langle x + s(y), x + y\rangle$$
$$\langle x * s(y), x * y\rangle$$
$$\langle x * s(y), x * y + x\rangle$$

Rewriting in $PD(\mathcal{R})$

Remark :

1. If $t \models\!\Rightarrow_{\mathcal{R}} v$, then $t \to_{\mathcal{R}} v$.
2. If $t \Rightarrow_{\mathcal{R}} v$, then $t \to_{\mathcal{R}} v$.
3. If $u \models\!\Rightarrow_{PD(\mathcal{R})} v$, then exists a term t and a position $p \in Pos(t)$ s.t. $u \models\!\Rightarrow_{\mathcal{R}} t[v]_p$ (donc $u \to_{\mathcal{R}} t[v]_p$).

Dependency sequence

A dependency sequence of $\mathcal{R}$ is a sequence $u_0, v_0, u_1, v_1, \ldots$ where

$u_0 \Rightarrow^*_{\mathcal{R}} v_0 \models\!\Rightarrow_{PD(\mathcal{R})} u_1 \Rightarrow^*_{\mathcal{R}} v_1 \models\!\Rightarrow_{PD(\mathcal{R})} \Rightarrow^*_{\mathcal{R}} u_2 \ldots$

to which we can associate a $\mathcal{R}$-reduction sequence

$u_0 \to^*_{\mathcal{R}} v_0 \models\!\Rightarrow_{\mathcal{R}} v'_0[u_1]_{p_0} \to^*_{\mathcal{R}} v'_0[v_1]_{p_0} \models\!\Rightarrow_{\mathcal{R}} v'_0[v'_1[u_2]_{p_1}]_{p_0} \to^*_{\mathcal{R}} ..$

Corollaire : (Completeness) If $\mathcal{R}$ terminates, then every dependency sequence of $\mathcal{R}$ is finite.

Soundness of the method

Lemme : Let $\mathcal{R}$ a non-terminating system. Then every non-terminating term contains a non-terminating subterm $f(\vec{u})$ where $f \in \mathcal{D}$ and $\vec{u}$ is a vector of terminating terms.

Lemme : If every dependency sequence of $\mathcal{R}$ is finite and the vector of terms $\vec{u}$ is terminating, then for every $f \in \mathcal{D}$, $f(\vec{u})$ is terminating.

Théorème : (Soundness) If every dependency sequence of $\mathcal{R}$ is finite, then $\mathcal{R}$ terminates.

Termination by Dependency Paris

Théorème : A rewriting system $\mathcal{R}$ terminates if there exists a pre-order $\gtrsim$ which is stable by substitutions et monotone s.t. its strict part $\succ$ is stable by substitutions and well-founded and s.t.
- $l \gtrsim r$ for every rule $l \to r \in \mathcal{R}$
- $s \succ t$ for every dependency pair $\langle s, t \rangle$

Exemple :

$$\begin{aligned} |0| &= 1 \\ |s(x)| &= |x| + 1 \\ |x + y| &= |x| + |y| \\ |x * y| &= 2 * |x| * |y| + 1 \end{aligned}$$

$$\begin{aligned} |x + 0| = |x| + 1 &\succ 1 = |0| \\ |x + s(y)| = |x| + |y| + 1 &\gtrsim |x| + |y| + 1 = |s(x + y)| \\ |x * 0| = 2 * |x| + 1 &\succ 1 = |0| \\ |x * s(y)| = 2 * |x| * |y| + 2 * |x| + 1 &\succ 2 * |x| * |y| + |x| + 1 = |x * y \\ |x + s(y)| = 2 * |x| * |y| + 2 * |x| + 1 &\succ |x| + |y| = |x + y| \\ |x * s(y)| = 2 * |x| * |y| + 2 * |x| + 1 &\succ 2 * |x| * |y| + 1 = |x * y| \end{aligned}$$