Termination

Motivations

Termination is essential to proof correctness of programs.

But

Termination is an undecidable property.

Undecidability 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 $\texttt{Diag}$ is an 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 function `end`.

Termination of a very simple system

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

is not even trivial!

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

Techniques to show termination

- Reduction orders
  - Particular case : interpretations
  - Example of interpretation : polynomial orders
- Useful orders :
  - Lexicographic order
  - Multi-set order
- Simplification orders
  - General result
  - Example : RPO
- Combination of orders :
  - Motivations
  - Postponement
  - Projection/simulation
- Dependency pairs

Recall

The symbol $f \in \Sigma$ is monotonic w.r.t the relation R iff
$a_i \; R \; b_i$ implies $f(a_1, \ldots, a_i, \ldots, a_n) \; R \; f(a_1, \ldots, b_i, \ldots, a_n)$.

A relation R over $\mathcal{T}(\mathcal{X}, \Sigma)$ is stable by substitution iff
$t \; R \; t'$ implies $\theta(t) \; R \; \theta(t')$ for every substitution θ.

Termination by reduction orders

Pre-order : reflexive and transitive relation.

and thus antisymmetric

Partial order : reflex[illegible] and transitive relation.

Strict order : irreflexive 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 ?

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

How does it work ?

Does $\mathcal{R}$ terminate ?

$$\mathcal{R} = \left\{ \begin{array}{lcl} \mathtt{por}(x, \mathtt{t}) & \to & \mathtt{t} \\ \mathtt{por}(\mathtt{t}, x) & \to & \mathtt{t} \end{array} \right.$$

size of s

The number … decreases….

"$s \succ_1 t$ iff $|s| > |t|$" is not a reduction order (not stable by substitution) :

$\mathtt{por}(x, \mathtt{por}(y, \mathtt{t})) \succ_1$ …

number of x in s

$\mathtt{por}(\mathtt{t}, \mathtt{por}(\mathtt{por}(\mathtt{t}, \mathtt{t}), \mathtt{t}))$ … $, \mathtt{por}(\mathtt{t}, \mathtt{t}))$.

"$s \succ_2 t$ iff $|s| > |t|$ and $\forall x \; |s|_x \geq |t|_x$" 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 $\mathcal{A}$.

[illegible]iven by :

This order is defined on the interpretations of s and t

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

Theorem : 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 each $f/n \in \Sigma$, there is s.t. $f^{\mathcal{P}_{\mathbb{N}}}(a_1, \ldots, a_n) = P_f(a_1, \ldots, a_n)$.

Example : 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.

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

Theorem : Let $\mathcal{P}_{\mathbb{N}}$ be a polynomial 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 a polynomial for every function symbol :
   $P_f(x, y) = x.y$ et $P_g(x, y) = 2.x + y + 1$.
2. 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 $\sigma(x), \sigma(y), \sigma(z)$.
3. Define the domain in which all the inequalities are valid :
   $\mathbb{N} - \{0, 1\}$.

Lexicographic order - particular case

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

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

Example :

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

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 :

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

Theorem : 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)	$\longrightarrow$	n+1
ackerman(m+1,0)	$\longrightarrow$	ackerman(m,1)
ackerman(m+1,n+1)	$\longrightarrow$	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 ?

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

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 $\mathcal{A}$.

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

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 $\forall i, x \succ y_i$.

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

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

Theorem : 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 a finite number of coins of any amount.

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. $t \rhd u$ 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

– $s = f(s_1, \dots, s_n)$ and $t = f(t_1, \dots, t_n)$ and $\forall i \; s_i \trianglerighteq_{emb} t_i$

– $s = f(s_1, \dots, s_n)$ and there is j s.t. $s_j \trianglerighteq_{emb} t$

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

Termination by simplification orders

Lemma : The relation $\triangleright_{emb}$ is contained in every simplification order.

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

Proof. Uses the famous Kruskal's Theorem. ■

And the inverse ?

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

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

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

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

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

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

Exa … ing

reflexive and transitive

Let $\succsim_\Sigma$ be a pre-order over Σ. We associate to each 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

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

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

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

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

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 $\succ_{rpo}$

Theorem : The relation $\succ_{rpo}$ is a WF order.

Theorem : The relation $\succ_{rpo}$ is reduction order.

As a consequence, to prove SN of a given system $\mathcal{R}$, it is sufficient to find an order $\succ_{rpo}$ such that $l \succ_{rpo} r$ for every $l \to r \in \mathcal{R}$.

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

Simple example

$$\mathcal{R} \left\{ \begin{array}{lll} 0 + y & \rightarrow_{r1} & y \\ s(x) + y & \rightarrow_{r2} & s(x + y) \\ 0 * y & \rightarrow_{r3} & 0 \\ s(x) * y & \rightarrow_{r4} & (x * y) + y \end{array} \right.$$

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

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

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

Famous example : cut elimination in intuitionistic logic

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

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 :

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

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

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

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

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

Theorem : 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.

Corollary : If $\mathcal{S}$ is WF, then $\mathcal{S} \cup \rhd$ is WF.

Example

Consider simply typed λ-calculus with the following rules :

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

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 $\eta \cup \Omega \cup \beta$ is SN.

Termination by projection/simulation

Theorem : 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.
   - $a \longrightarrow_{\mathcal{R}_1} b$ implies $\mathcal{T}(a) \longrightarrow^{+}_{\mathcal{S}} \mathcal{T}(b)$,
   - $a \longrightarrow_{\mathcal{R}_2} b$ implies $\mathcal{T}(a) \longrightarrow^{*}_{\mathcal{S}} \mathcal{T}(b)$.

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

(Famous) Example

Consider simply typed extensional λ-calculus

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

Thus for example if $z : A \times B$ and $x : (A \times B) \to (C \to D)$, then

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

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

- Show that $\eta_{exp} \cup sp_{exp}$ is terminating.
- Show that $\beta \cup \pi_1 \cup \pi_2$ is terminating (done).
- Show that $\eta_{exp} \cup sp_{exp}$ is also confluent.
- Define $\mathcal{T}(t)$ as the $\eta_{exp} \cup sp_{exp}$-normal form of t.
- Show that $t \to_{\beta\cup\pi_1\cup\pi_2}\; t'$ implies $\mathcal{T}(t) \to^{+}_{\beta\cup\pi_1\cup\pi_2}\; \mathcal{T}(t')$
- Show that $t \to_{\eta_{exp}\cup sp_{exp}}\; t'$ implies $\mathcal{T}(t) = \mathcal{T}(t)$ (evident).
- Conclude that all the system $\mathcal{R}_1 \cup \mathcal{R}_2$ is SN.

Termination by Dependency Pairs

- The technique is due to Aarts and Giesl.
- The order does not decrease for every step, but for the dependent one
- 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.