Introduction to Term Rewriting: Techniques and Applications

Termination with the Recursive Path Ordering

Departamento de Sistemas Informáticos y Computación (DSIC)
Universidad Politécnica de Valencia (UPV)

Salvador Lucas
http://www.dsic.upv.es/~slucas

Summer School on Foundations of Information Technologies – FIT 2009
The recursive path ordering (rpo) $\succ_{\text{rpo}}$ is a well-known syntactic ordering over terms [Der79]. Its definition requires two main ingredients:

1. A precedence (i.e., a reflexive and transitive relation) $\succeq_{\mathcal{F}}$ over the signature $\mathcal{F}$ (i.e., $\succeq_{\mathcal{F}} \subseteq \mathcal{F} \times \mathcal{F}$) is given to compare symbols in $\mathcal{F}$.

2. Extensions of $\succ_{\text{rpo}}$ from terms to tuples of terms $(t_1, \ldots, t_n)$ for some $n > 1$. Here, we consider two basic extensions:
   - **Lexicographic** extension: the tuple elements are compared from left to right.
   - **Multiset** extension: the tuple elements are compared disregarding the order in which they have been arranged.
Lexicographic extension of an ordering

**Definition (Lexicographic extension)**

Given a relation \( R \) over a set \( A \), we extend \( R \) to a relation \( R^{\text{lex}} \) on \( A^n \) for some \( n > 0 \) as follows:

\[
(a_1, a_2, \ldots, a_n) \; R^{\text{lex}} \; (b_1, b_2, \ldots, b_n) \; \text{if}
\begin{cases}
    a_1 \; R \; b_1 \; \text{or else} \\
    a_1 = b_1 \; \text{and} \; (a_2, \ldots, a_n) \; R^{\text{lex}} \; (b_2, \ldots, b_n)
\end{cases}
\]

**Example** If \( > \mathbb{N} \) is the usual ordering over natural numbers, we have:

\[
(3, 1) >^{\text{lex}} \mathbb{N} (2, 3) \quad \text{and} \quad (3, 3) >^{\text{lex}} \mathbb{N} (3, 1)
\]

**Proposition**

*If \( > \) is a strict ordering, then \( >^{\text{lex}} \) is a strict ordering. If \( > \) is well-founded, then \( >^{\text{lex}} \) is well-founded.*
Multiset extension of an ordering

Given a set $A$, a multiset $M : A \rightarrow \mathbb{N}$ is a mapping from $A$ into $\mathbb{N}$ representing a collection of elements in $A$ where duplicates are allowed but the ordering does not matter.

Let $\mathcal{M}(A)$ be the set of all multisets over a set $A$.

**Example** The multisets $M = \{1, 2, 2, 3\}$ (formally $M(1) = 1$, $M(2) = 2$, $M(3) = 1$ and $M(n) = 0$ for all $n \in \mathbb{N} - \{1, 2, 3\}$), $\{2, 1, 2, 3\}$ and $\{2, 2, 1, 3\}$ are the same.
Multiset extension of an ordering

Definition (Multiset extension)

Given a relation $R$ over a set $A$, we extend $R$ to a relation $R_{mul}$ on $\mathcal{M}(A)$ as follows: $M R_{mul} N$ if $N = (M - X) \cup Y$ for multisets $X, Y \in \mathcal{M}(A)$ such that

$$\emptyset \not= X \subseteq M \quad \text{and} \quad \forall y \in Y, \exists x \in X, x R y.$$

Example We have:

\[
\begin{align*}
\{\{3, 5\}\} & \succ_{N}^{mul} \{\{3, 4, 4, 2\}\} & \{\{3, 3, 4, 0\}\} & \succ_{N}^{mul} \{\{3, 4\}\} \\
\{\{3, 3, 4, 0\}\} & \succ_{N}^{mul} \{\{3, 2, 2, 1, 1, 1, 4, 0\}\} & \{\{3, 3, 4, 0\}\} & \succ_{N}^{mul} \{\{3, 3, 3, 3, 2, 2\}\} \\
\{\{3, 3, 4, 0\}\} & \succ_{N}^{mul} \{\emptyset\}
\end{align*}
\]

Thus, if $\succ$ is an ordering, we have $M \succ_{N}^{mul} N$ if we can obtain $M$ out from $N$ by either:

1. **Removing** some (possibly duplicated) elements $x \in X$, or
2. **Replacing** some elements $x \in X$ by new but *smaller* elements $y \in Y$. 

Salvador Lucas

FIT’09
Multiset extension of an ordering

**Definition (Multiset ordering over tuples)**

Given an ordering $>$ over a set $A$, we extend $>$ to an ordering $>^{mul}$ on tuples in $A^n$ (for some $n > 0$) as follows:

$$(a_1,\ldots,a_n) >^{mul} (b_1,\ldots,b_n) \text{ if } \{a_1,\ldots,a_n\} >^{mul} \{b_1,\ldots,b_n\}$$

**Proposition**

*If $>$ is a strict ordering, then $>^{mul}$ is a strict ordering. If $>$ is well-founded, then $>^{mul}$ is well-founded.*
Combining lexicographic and multiset extensions

In the definition of recursive path ordering, we use the following notion:

**Definition (Status function)**

Let $\mathcal{F}$ be a signature. Given $f \in \mathcal{F}$, a *status* function $\tau$ for $\mathcal{F}$ returns

$$\tau(f) \in \{\text{lex}_{\pi_f}, \text{mul}\}$$

(where $\pi_f$ is a permutation of $\{1, \ldots, \text{ar}(f)\}$) for each $f \in \mathcal{F}$.

The meaning of $\text{lex}_{\pi_f}$ in the definition above is: *when comparing tuples of terms* $\vec{s} = (s_1, s_2, \ldots, s_n)$ and $\vec{t} = (t_1, t_2, \ldots, t_k)$ *using* $\triangleright_{\text{lex}_{\pi_f}}$, *first apply* $\pi_f$ *to both* $\vec{s}$ *and* $\vec{t}$; *then use the lexicographic extension as defined above*.

**Example**

1. $\text{lex}_{\pi_f}$ with $\pi_f = (1, 2, \ldots, k)$ for a $k$-ary symbol $f$ corresponds to the usual *left-to-right* lexicographic comparison (and we often just write $\text{lex}$).

2. $\text{lex}_{\pi_f}$ with $\pi_f = (k, k - 1, \ldots, 1)$ for a $k$-ary symbol $f$ corresponds to the *right-to-left* lexicographic comparison.
The Recursive Path Ordering (RPO)

Definition (Recursive Path Ordering with status)

Let $\mathcal{F}$ be a signature, $\succ_{\mathcal{F}}$ be a strict ordering (precedence) on $\mathcal{F}$, and $\tau$ be a status function for $\mathcal{F}$. An ordering $>_{rpo}$ (recursive path ordering) on $\mathcal{T}(\mathcal{F}, \mathcal{X})$ is given as follows: for all terms $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$, $s >_{rpo} t$ if $s = f(s_1, \ldots, s_m)$ and

1. either $s_i = t$ or $s_i >_{rpo} t$ for some $s_i, 1 \leq i \leq m$, or
2. $t = g(t_1, \ldots, t_n)$, $s >_{rpo} t_i$ for all $i, 1 \leq i \leq n$ and either
   1. $f \succ_{\mathcal{F}} g$, or
   2. $f = g$ and $(s_1, \ldots, s_n) >_{rpo}^{\tau(f)} (t_1, \ldots, t_n)$

Roughly speaking, $s >_{rpo} t$ if

1. There is a subterm $s'$ of $s$ which is bigger than (or equal to) $t$, or
2. $s$ is bigger than all the immediate subterms $t_i$ of $t$ and either
   - $f$ is bigger than $g$ (according to the precedence $\succ_{\mathcal{F}}$), or
   - $f$ equals $g$ but the tuple $(s_1, \ldots, s_n)$ accompanying $f$ in $s$ is bigger than the one accompanying $g$ in $t$ (i.e., $(t_1, \ldots, t_n)$).
The Recursive Path Ordering (RPO)

**Example** For the signature $\mathcal{F} = \{ack, s, 0\}$ with $ar(ack) = 2$, $ar(s) = 1$, $ar(0) = 0$ and the precedence $\succ$ given by $ack \succ s$, we have:

- $ack(0, M) \succ_{rpo} s(M)$ because $ack(0, M) \succ_{rpo} M$ (this is because $M$ is a subterm of $ack(0, M)$), and $ack \succ s$.
- We have that $ack(s(M), 0) \succ_{rpo} ack(M, s(0))$. On one side,

$$ack(s(M), 0) \succ_{rpo} M,$$

because $M$ is a subterm of $ack(s(M), 0)$ and

$$ack(s(M), 0) \succ_{rpo} s(0)$$

(because $ack \succ s$ and $ack(s(M), 0) \succ_{rpo} 0$). On the other hand, we clearly have:

$$(s(M), 0) \succ_{\text{lex}}^{rpo} (M, s(0)).$$
The Recursive Path Ordering (RPO)

**Theorem (RPO as a reduction ordering)**

Let $\mathcal{F}$ be a finite signature and $\succ$ be a strict precedence over $\mathcal{F}$. Then, $\succ_{\text{rpo}}$ is a reduction ordering.

Therefore, we can use the recursive path ordering for proving termination of TRSs over finite signatures.

**Example** The following TRS $\mathcal{R}$ encoding Ackermann’s function:

- $\text{ack}(0, M) \rightarrow s(M)$
- $\text{ack}(s(M), 0) \rightarrow \text{ack}(M, s(0))$
- $\text{ack}(s(M), s(N)) \rightarrow \text{ack}(M, \text{ack}(s(M), N))$

can be proved terminating by an RPO (i.e., it is RPO-terminating).
Termination with the Recursive Path Ordering

The Recursive Path Ordering (RPO)

**Definition (RPO-termination)**

A TRS $\mathcal{R} = (\mathcal{F}, R)$ is *RPO-terminating* if there is an RPO $\succ_{rpo}$ for $\mathcal{F}$ which is *compatible* with the rules of $\mathcal{R}$, i.e., $\ell \succ_{rpo} r$ for all $\ell \rightarrow r \in R$.

The following results provide information about the *complexity* of proving RPO-termination.

**Theorem (Comparing terms using RPO)**

Let $\mathcal{F}$ be a signature, $s, t \in T(\mathcal{F}, \mathcal{X})$, and $\succ_{rpo}$ be an RPO for $\mathcal{F}$. We can decide whether $s \succ_{rpo} t$ in *polynomial* time over the *size* of $s$ and $t$.

**Theorem (Deciding RPO-termination)**

*RPO-termination of finite TRSs is *decidable* but *NP-complete*. 
N. Dershowitz.

A note on simplification orderings.