Rewriting

Part 4. Termination of Term Rewriting Systems

Temur Kutsia

RISC, JKU Linz

Termination

Definition 4.1

A term rewriting system R is terminating iff $\to_R$ is terminating, i.e., there is no infinite reduction chain

$$t_0 \to_R t_1 \to_R t_2 \to_R \cdots$$

Termination is Undecidable

The following problem is undecidable:

Given: A finite TRS R.

Question: Is R terminating or not?

Proof by reduction of the uniform halting problem for Turing Machines.

A Decidable Subcase

Definition 4.2
A TRS R is called right-ground iff for all $l \to r \in R$, we have $\mathcal{V}ar(r) = \emptyset$ (i.e., r is ground).

A Decidable Subcase

Lemma 4.1

Let R be a finite right-ground TRS. Then the following statements are equivalent:

1. *R does not terminate.*
2. *There exists a rule $l \to r \in R$ and a term t such that $r \xrightarrow{+}_R t$ and t contains r as a subterm.*

Proof.

$(2 \Rightarrow 1)$ is obvious: 2 yields an infinite reduction

$$r \xrightarrow{+}_R t = t[r]_p \xrightarrow{+}_R t[t]_p = t[t[r]_p]_p \xrightarrow{+}_R \cdots$$

A Decidable Subcase

Lemma 4.1

Let R be a finite right-ground TRS. Then the following statements are equivalent:

1. *R does not terminate.*
2. *There exists a rule $l \to r \in R$ and a term t such that $r \xrightarrow{+}_R t$ and t contains r as a subterm.*

Proof (Cont.)

$(1 \Rightarrow 2)$: By induction on cardinality of R. If R is empty, 1 is false. Assume $|R| > 0$ and consider an infinite reduction
$t_1 \to_R t_2 \to_R \cdots$

A Decidable Subcase

Lemma 4.1

Let R be a finite right-ground TRS. Then the following statements are equivalent:

1. *R does not terminate.*
2. *There exists a rule $l \to r \in R$ and a term t such that $r \xrightarrow{+}_R t$ and t contains r as a subterm.*

Proof (Cont.)

(i) Assume wlog that at least one of the reductions in $t_1 \to_R t_2 \to_R \cdots$ occurs at position ϵ.

(ii) This means that there exist an index i, a rule $l \to r \in R$, and a substitution σ such that $t_i = \sigma(l)$ and $t_{i+1} = \sigma(r) = r$. Therefore, there exists an infinite reduction $r \to_R t_{i+2} \to_R t_{i+3} \to_R \cdots$ starting from r.

A Decidable Subcase

Lemma 4.1

Let R be a finite right-ground TRS. Then the following statements are equivalent:

1. *R does not terminate.*
2. *There exists a rule $l \to r \in R$ and a term t such that $r \xrightarrow{+}_R t$ and t contains r as a subterm.*

Proof (Cont.)

Two cases:

(a) $l \to r$ is not used in this reduction. Then $R \setminus \{l \to r\}$ does not terminate and we can apply the induction hypothesis.

(b) $l \to r$ is used in the reduction. Hence, there exists $j \geq 2$ such that r occurs in t_{i+j} and 2 holds.

Decision Procedure for Termination of Right-Ground TRSs

- Given a finite right-ground TRS $R = \{l_1 \to r_1, \ldots, l_n \to r_n\}$.
- Take the right hand sides $r_1, \ldots, r_n$.
- Simultaneously generate all reduction sequences starting from $r_1, \ldots, r_n$:
  - First generate all sequences of length 1,
  - Then generate all sequences of length 2,
  - etc.
- Either one detects the cycle $r_i \xrightarrow{k}_R t$, $k \geq 1$, where t contains r_i as a subterm (R is not terminating),
- or the process of generating these reductions terminates (R is terminating).

Decision Procedure for Termination of Right-Ground TRSs

- Given a finite right-ground TRS $R = \{l_1 \to r_1, \ldots, l_n \to r_n\}$.
- Take the right hand sides $r_1, \ldots, r_n$.
- Simultaneously generate all reduction sequences starting from $r_1, \ldots, r_n$:
  - First generate all sequences of length 1,
  - Then generate all sequences of length 2,
  - etc.
- Either one detects the cycle $r_i \xrightarrow{k}_R t$, $k \geq 1$, where t contains r_i as a subterm (R is not terminating),
- or the process of generating these reductions terminates (R is terminating).

Theorem 4.1

For finite right-ground TRSs, termination is decidable.

Reduction Orders: A Tool for Proving Termination

- Termination problem is undecidable. There can not be a general procedure that
  - given an arbitrary TRS
  - answers with "yes" if the system is terminating, and with "no" otherwise.

Reduction Orders: A Tool for Proving Termination

- Termination problem is undecidable. There can not be a general procedure that
  - given an arbitrary TRS
  - answers with "yes" if the system is terminating, and with "no" otherwise.
- However, often it is necessary to prove for a particular system that it terminates.

Reduction Orders: A Tool for Proving Termination

- Termination problem is undecidable. There can not be a general procedure that
  - given an arbitrary TRS
  - answers with “yes” if the system is terminating, and with “no” otherwise.
- However, often it is necessary to prove for a particular system that it terminates.
- It is possible to develop tools that facilitate this task. Ideally, it should be possible to automate them.

Reduction Orders: A Tool for Proving Termination

- Termination problem is undecidable. There can not be a general procedure that
  - given an arbitrary TRS
  - answers with "yes" if the system is terminating, and with "no" otherwise.
- However, often it is necessary to prove for a particular system that it terminates.
- It is possible to develop tools that facilitate this task. Ideally, it should be possible to automate them.
- Undecidability of termination implies that such methods can not succeed for all terminating rewrite systems.

Reduction Orders: A Tool for Proving Termination

- Idea: Define a class of strict orders $>$ on terms such that

  $$l > r \text{ for all } (l \to r) \in R$$

  implies termination of R.

Reduction Orders: A Tool for Proving Termination

- Idea: Define a class of strict orders $>$ on terms such that

  $$l > r \text{ for all } (l \to r) \in R$$

  implies termination of R.
- Reduction orders.

Reduction Orders: A Tool for Proving Termination

Definition 4.3

A strict order $>$ on $T(\mathcal{F}, \mathcal{V})$ is called a reduction order iff it is

1. compatible with $\mathcal{F}$-operations: If $s_1 > s_2$, then

   $$f(t_1, \ldots, t_{i-1}, s_1, t_{i+1}, \ldots, t_n) > f(t_1, \ldots, t_{i-1}, s_2, t_{i+1}, \ldots, t_n)$$

   for all $t_1, \ldots, t_{i-1}, s_1, s_2, t_{i+1}, \ldots, t_n \in T(\mathcal{F}, \mathcal{V})$ and $f \in \mathcal{F}^n$,
2. closed under substitutions: If $s_1 > s_2$, then $\sigma(s_1) > \sigma(s_2)$ for all $s_1, s_2 \in T(\mathcal{F}, \mathcal{V})$ and a $T(\mathcal{F}, \mathcal{V})$-substitution σ,
3. well-founded.

Reduction Orders: A Tool for Proving Termination

Example 4.1

- $|t|$: The size of the term t.
- The order $>$ on $T(\mathcal{F}, \mathcal{V})$: $s > t$ iff $|s| > |t|$.

Reduction Orders: A Tool for Proving Termination

Example 4.1

- $|t|$: The size of the term t.
- The order $>$ on $T(\mathcal{F}, \mathcal{V})$: $s > t$ iff $|s| > |t|$.
- $>$ is compatible with $\mathcal{F}$-operations and well-founded.

Reduction Orders: A Tool for Proving Termination

Example 4.1

- $|t|$: The size of the term t.
- The order $>$ on $T(\mathcal{F}, \mathcal{V})$: $s > t$ iff $|s| > |t|$.
- $>$ is compatible with $\mathcal{F}$-operations and well-founded.
- However, $>$ is not a reduction order because it is not closed under substitutions:

$$|f(f(x,x),y)| = 5 > 3 = |f(y,y)|$$

For $\sigma = \{y \mapsto f(x,x)\}$:

$$|\sigma(f(f(x,x),y))| = |f(f(x,x),f(x,x))| = 7,$$
$$|\sigma(f(y,y)| = |f(f(x,x),f(x,x))| = 7.$$

Reduction Orders: A Tool for Proving Termination

Example 4.1 (Cont.)

- $|t|_x$: The number of occurrences of x in t.
- The order $>$ on $T(\mathcal{F}, \mathcal{V})$: $s > t$ iff $|s| > |t|$ and $|s|_x \geq |t|_x$ for all $x \in \mathcal{V}$.

Reduction Orders: A Tool for Proving Termination

Example 4.1 (Cont.)

- $|t|_x$: The number of occurrences of x in t.
- The order $>$ on $T(\mathcal{F}, \mathcal{V})$: $s > t$ iff $|s| > |t|$ and $|s|_x \geq |t|_x$ for all $x \in \mathcal{V}$.
- $>$ is a reduction order.

Why Are Reduction Orders Interesting?

Theorem 4.2

A TRS R terminates iff there exists a reduction order $>$ that satisfies $l > r$ for all $l \to r \in R$.

Why Are Reduction Orders Interesting?

Theorem 4.2

A TRS R terminates iff there exists a reduction order $>$ that satisfies $l > r$ for all $l \to r \in R$.

Proof.

($\Rightarrow$): Assume R terminates. Then $\xrightarrow{+}_R$ is a reduction order, satisfying $l \xrightarrow{+}_R r$ for all $l \to r \in R$.

($\Leftarrow$): $l > r$ implies $t[\sigma(l)]_p > t[\sigma(r)]_p$ for all terms t, substitutions σ, and positions p. Thus, $l > r$ for all $l \to r \in R$ implies $s_1 > s_2$ for all s_1, s_2 with $s_1 \to_R s_2$. Since $>$ is well-founded, there can not be infinite reduction $s_1 \to_R s_2 \to_R s_2 \to_R \cdots$. $\square$

Reduction Orders: An Example

Example 4.2

The TRS

$$R := \{f(x, f(y, x)) \to f(x, y),\ f(x, x) \to x\}$$

is terminating. For the reduction order defined as

$$s > t \text{ iff } |s| > |t| \text{ and } |s|_x \geq |t|_x \text{ for all } x \in \mathcal{V}$$

we have

$$f(x, f(y, x)) > f(x, y),\ f(x, x) > x.$$

Reduction Orders: Example

Example 4.2 (Cont.)

The TRS

$$R \cup \{f(f(x,y),z) \to f(x,f(y,z))\}$$

is also terminating. But this can not be shown by the previous reduction order because

$$f(f(x,y),z) \not> f(x,f(y,z)).$$

Methods for Construction Reduction Orders

- Polynomial orders
- Simplification orders:
  - Recursive path orders
  - Knuth-Bendix orders

Methods for Construction Reduction Orders

- Polynomial orders
- Simplification orders:
  - Recursive path orders
  - Knuth-Bendix orders

Goal: Provide a variety of different reduction orders that can be used to show termination; not only by hand, but also automatically.

Polynomial Orders

Interpretation method. The idea:

- Interpret terms in an $\mathcal{F}$-algebra that is equipped with a well-founded order.
- Compare terms with respect to their interpretations: A term s is larger than a term t iff the interpretation of s is larger than the interpretation of t.

One has to make sure that the ordering on interpretation induces a reduction order on terms.

Polynomial Orders. Interpreting Terms

Definition 4.4

A polynomial interpretation $\mathcal{P}$ of a signature $\mathcal{F}$ is an $\mathcal{F}$-algebra $\mathcal{P} = (A, \{P_f\}_{f \in \mathcal{F}})$ such that

- the carrier set A is a nonempty set of positive integers: $A \subseteq \mathbb{N} \setminus \{0\}$,
- every n-ary function symbol f is associated with a polynomial $P_f(X_1, \ldots, X_n) \in \mathbb{N}[X_1, \ldots, X_n]$ such that for all $a_1, \ldots, a_n \in A$, $f_{\mathcal{P}}(a_1, \ldots, a_n) := P_f(a_1, \ldots, a_n) \in A$.

A well-founded order $>$ on A is the usual order on natural numbers.

Polynomial Orders. Interpreting Terms

Example 4.3

Let $\mathcal{F} = \{\oplus, \odot\}$ consists of two binary function symbols and let $A := \mathbb{N} \setminus \{0, 1\}$. Define

$$P_\oplus(x, y) := 2x + y + 1$$
$$P_\odot(x, y) := xy$$

The mapping from function symbols to polynomial functions can be extended to terms, mapping variables $(x, y, z, \ldots)$ to indeterminates $(X, Y, Z, \ldots)$. For example:

$$t = x \odot (x \oplus y)$$
$$P_t = P_\odot(X, P_\oplus(X, Y)) = X(2X + Y + 1) = 2X^2 + XY + X.$$

Polynomial Orders. Guaranteeing Compatibility

- If in the previous example we had defined $P_{\odot}(x,y) := x^2$, the interpretation would not be compatible with $\mathcal{F}$-operations.
- $3 > 2$, but $\odot_{\mathcal{P}}(2,3) = P_{\odot}(2,3) = 4 = P_{\odot}(2,2) = \odot_{\mathcal{P}}(2,2)$.

Polynomial Orders. Guaranteeing Compatibility

- If in the previous example we had defined $P_\odot(x, y) := x^2$, the interpretation would not be compatible with $\mathcal{F}$-operations.
- $3 > 2$, but $\odot_\mathcal{P}(2, 3) = P_\odot(2, 3) = 4 = P_\odot(2, 2) = \odot_\mathcal{P}(2, 2)$.

Definition 4.5 (Monotony)

- A polynomial $P(X_1, \ldots, X_n) \in \mathbb{N}[X_1, \ldots, X_n]$ is a monotone polynomial iff it depends on all its indeterminates.
- A monotone polynomial interpretation is a polynomial interpretation in which all function symbols are associated with monotone polynomials.

Polynomial Orders. Guaranteeing Compatibility

- If in the previous example we had defined $P_{\odot}(x, y) := x^2$, the interpretation would not be compatible with $\mathcal{F}$-operations.
- $3 > 2$, but $\odot_{\mathcal{P}}(2, 3) = P_{\odot}(2, 3) = 4 = P_{\odot}(2, 2) = \odot_{\mathcal{P}}(2, 2)$.

Definition 4.5 (Monotony)

- A polynomial $P(X_1, \ldots, X_n) \in \mathbb{N}[X_1, \ldots, X_n]$ is a monotone polynomial iff it depends on all its indeterminates.
- A monotone polynomial interpretation is a polynomial interpretation in which all function symbols are associated with monotone polynomials.

X^2 is not a monotone polynomial in $\mathbb{N}[X, Y]$.

Polynomial Orders. Inducing Reduction Order

- Why are monotone polynomial interpretations interesting?

Polynomial Orders. Inducing Reduction Order

- Why are monotone polynomial interpretations interesting?
- They help to define an ordering on terms which is compatible with $\mathcal{F}$-operations (in fact, to define a reduction order).

Polynomial Orders. Inducing Reduction Order

Theorem 4.3

Let $\mathcal{P} = (A, \{f_\mathcal{P}\}_{f\in\mathcal{F}})$ be a monotone polynomial interpretation of $\mathcal{F}$ with the well-founded ordering $>$ on A. Then $a > b$ implies

$$f_\mathcal{P}(a_1, \ldots, a_{i-1}, a, a_{i+1}, \ldots, a_n) > f_\mathcal{P}(a_1, \ldots, a_{i-1}, b, a_{i+1}, \ldots, a_n)$$

for all $f_\mathcal{P}$ and $a, b, a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n \in A$.

Proof.

We can write $P_f \in \mathbb{N}[X_1, \ldots, X_n] = (\mathbb{N}[X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n])[X_i]$ as a polynomial in X_i with coefficients $Q_j \in \mathbb{N}[X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n]$:

$$\begin{aligned} f_\mathcal{P} = P_f = {} & Q_k(X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n)X_i^k + \cdots + \\ & Q_1(X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n)X_i + \\ & Q_0(X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n). \end{aligned}$$

Polynomial Orders. Inducing Reduction Order

Theorem 4.3

Let $\mathcal{P} = (A, \{f_\mathcal{P}\}_{f \in \mathcal{F}})$ *be a monotone polynomial interpretation of* $\mathcal{F}$ *with the well-founded ordering* $>$ *on* A. *Then* $a > b$ *implies*

$$f_\mathcal{P}(a_1, \ldots, a_{i-1}, a, a_{i+1}, \ldots, a_n) > f_\mathcal{P}(a_1, \ldots, a_{i-1}, b, a_{i+1}, \ldots, a_n)$$

for all $f_\mathcal{P}$ *and* $a, b, a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n \in A$.

Proof (cont.)

Since P_f is monotone, it depends on X_i. So, we can assume $k > 0$ and Q_k is not a zero polynomial.

□

Polynomial Orders. Inducing Reduction Order

Theorem 4.3

Let $\mathcal{P} = (A, \{f_\mathcal{P}\}_{f\in\mathcal{F}})$ be a monotone polynomial interpretation of $\mathcal{F}$ with the well-founded ordering $>$ on A. Then $a > b$ implies

$$f_\mathcal{P}(a_1, \ldots, a_{i-1}, a, a_{i+1}, \ldots, a_n) > f_\mathcal{P}(a_1, \ldots, a_{i-1}, b, a_{i+1}, \ldots, a_n)$$

for all $f_\mathcal{P}$ and $a, b, a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n \in A$.

Proof (cont.)

Since P_f is monotone, it depends on X_i. So, we can assume $k > 0$ and Q_k is not a zero polynomial.
Hence, for all $a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n \in A \subseteq \mathbb{N} \setminus \{0\}$, $P_f(a_1, \ldots, a_{i-1}, X_i, a_{i+1}, \ldots, a_n)$ is a polynomial of degree $k > 0$ in X_i with coefficients in $\mathbb{N}$.

□

Polynomial Orders. Inducing Reduction Order

Theorem 4.3

Let $\mathcal{P} = (A, \{f_\mathcal{P}\}_{f \in \mathcal{F}})$ be a monotone polynomial interpretation of $\mathcal{F}$ with the well-founded ordering $>$ on A. Then $a > b$ implies

$$f_\mathcal{P}(a_1, \ldots, a_{i-1}, a, a_{i+1}, \ldots, a_n) > f_\mathcal{P}(a_1, \ldots, a_{i-1}, b, a_{i+1}, \ldots, a_n)$$

for all $f_\mathcal{P}$ and $a, b, a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n \in A$.

Proof (cont.)

Since P_f is monotone, it depends on X_i. So, we can assume $k > 0$ and Q_k is not a zero polynomial.
Hence, for all $a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n \in A \subseteq \mathbb{N} \setminus \{0\}$, $P_f(a_1, \ldots, a_{i-1}, X_i, a_{i+1}, \ldots, a_n)$ is a polynomial of degree $k > 0$ in X_i with coefficients in $\mathbb{N}$.
Therefore, $a > b$ implies $P_f(a_1, \ldots, a_{i-1}, a, a_{i+1}, \ldots, a_n) > P_f(a_1, \ldots, a_{i-1}, b, a_{i+1}, \ldots, a_n)$. □

Polynomial Orders. Inducing Reduction Order

Definition 4.6 (Polynomial Order)

The polynomial interpretation $\mathcal{P}$ of a signature $\mathcal{F}$ induces the following polynomial order $>_{\mathcal{P}}$ on $T(\mathcal{F}, \mathcal{V})$:

$$s >_{\mathcal{P}} t \text{ iff } P_s(a_1, \dots, a_n) > P_t(a_1, \dots, a_n)$$

for all $a_1, \dots, a_n$ in the carrier set of $\mathcal{P}$.

Polynomial Orders. Inducing Reduction Order

Theorem 4.4

The polynomial order $>_{\mathcal{P}}$ induced by a monotone polynomial interpretation $\mathcal{P}$ is a reduction order.

Proof.

$>_{\mathcal{P}}$ is a strict order on $T(\mathcal{F}, \mathcal{V})$.

- $>_{\mathcal{P}}$ is well-founded because $>$ is well-founded on the carrier set of $\mathcal{P}$.
- $>_{\mathcal{P}}$ is closed with respect to substitutions because in the definition of polynomial orders we consider all $a_1, \ldots, a_n$ in the carrier set.
- $>_{\mathcal{P}}$ is compatible to $\mathcal{F}$-operations due to Theorem 4.3.

□

Polynomial Orders. Inducing Reduction Order

Example 4.4

- TRS: $R = \{x \odot (y \oplus z) \to (x \odot y) \oplus (x \odot z)\}$.
- Polynomial order induced by

$$A := \mathbb{N} \setminus \{0, 1\},\ P_{\oplus} = 2X + Y + 1,\ P_{\odot} = XY.$$

- The polynomial associated to $l = x \odot (y \oplus z)$:

$$P_l = X(2Y + Z + 1) = 2XY + XZ + X.$$

- The polynomial associated to $r = (x \odot y) \oplus (x \odot z)$:

$$P_r = 2XY + XZ + 1.$$

- Since all elements of A are greater than 1, we have $l >_{\mathcal{P}} r$.

Polynomial Orders

- For a given polynomial order, in general, it is not possible to decide whether it is suitable for showing termination of a given TRS.
- It is a consequence of Hilbert's 10th problem.
- There are automated methods that can (sometimes) show $P >_{\mathcal{A}} Q$ for polynomials $P, Q \in \mathbb{N}[X_1, \ldots, X_n]$.

Polynomial Orders

Questions:

- How to find suitable polynomials?
- How to show that $P > 0$ for a polynomial $P \in \mathbb{Z}[x_1, \ldots, x_n]$?

Polynomial Orders

Questions:

- How to find suitable polynomials?
- How to show that $P > 0$ for a polynomial $P \in \mathbb{Z}[x_1, \ldots, x_n]$?

Modern approach:

1. Choose abstract polynomial interpretations (linear, quadratic, $\ldots$).

Polynomial Orders

Questions:

- How to find suitable polynomials?
- How to show that $P > 0$ for a polynomial $P \in \mathbb{Z}[x_1, \ldots, x_n]$?

Modern approach:

1. Choose abstract polynomial interpretations (linear, quadratic, $\ldots$).
2. Transform rewrite rules into polynomial ordering constraints.

Polynomial Orders

Questions:

- How to find suitable polynomials?
- How to show that $P > 0$ for a polynomial $P \in \mathbb{Z}[x_1, \ldots, x_n]$?

Modern approach:

1. Choose abstract polynomial interpretations (linear, quadratic, $\ldots$).
2. Transform rewrite rules into polynomial ordering constraints.
3. Add monotonicity and well-definedness constraints.

Polynomial Orders

Questions:

- How to find suitable polynomials?
- How to show that $P > 0$ for a polynomial $P \in \mathbb{Z}[x_1, \ldots, x_n]$?

Modern approach:

1. Choose abstract polynomial interpretations (linear, quadratic, $\ldots$).
2. Transform rewrite rules into polynomial ordering constraints.
3. Add monotonicity and well-definedness constraints.
4. Eliminate universally quantified variables requiring their coefficients to be nonnegative and the constant to be positive (sufficient condition).

Polynomial Orders

Questions:

- How to find suitable polynomials?
- How to show that $P > 0$ for a polynomial $P \in \mathbb{Z}[x_1, \ldots, x_n]$?

Modern approach:

1. Choose abstract polynomial interpretations (linear, quadratic, $\ldots$).
2. Transform rewrite rules into polynomial ordering constraints.
3. Add monotonicity and well-definedness constraints.
4. Eliminate universally quantified variables requiring their coefficients to be nonnegative and the constant to be positive (sufficient condition).
5. Translate resulting diophantine constraints to SAT or SMT problem.

Polynomial Orders

Example 4.5

- Rewrite system:

$$\{0 + y \to y, \quad s(x) + y \to s(x + y)\}$$

Polynomial Orders

Example 4.5

- Rewrite system:

$$\{0 + y \to y, \quad s(x) + y \to s(x + y)\}$$

- Interpretations:

$$0_{\mathcal{A}} = a \qquad s_{\mathcal{A}}(x) = bx + c \qquad +_{\mathcal{A}}(x, y) = dx + ey + f$$

Polynomial Orders

Example 4.5

- Rewrite system:

$$\{0 + y \to y, \quad s(x) + y \to s(x + y)\}$$

- Interpretations:

$$0_{\mathcal{A}} = a \qquad s_{\mathcal{A}}(x) = bx + c \qquad +_{\mathcal{A}}(x, y) = dx + ey + f$$

- Polynomial constraints: $\forall X, Y \in \mathbb{N}$

$$da + eY + f > Y$$
$$d(bX + c) + eY + f > b(dX + eY + f) + c$$

Polynomial Orders

Example 4.5

- Rewrite system:

$$\{0 + y \to y, \quad s(x) + y \to s(x + y)\}$$

- Interpretations:

$$0_{\mathcal{A}} = a \qquad s_{\mathcal{A}}(x) = bx + c \qquad +_{\mathcal{A}}(x, y) = dx + ey + f$$

- Polynomial constraints: $\forall X, Y \in \mathbb{N}$

$$\begin{aligned} da + eY + f &> Y \\ d(bX + c) + eY + f &> b(dX + eY + f) + c \end{aligned}$$

$$a \geq 0 \quad b \geq 1 \quad c \geq 0 \quad d \geq 1 \quad e \geq 1 \quad f \geq 0$$

Polynomial Orders

Example 4.5

- Rewrite system:

$$\{0 + y \to y, \quad s(x) + y \to s(x + y)\}$$

- Interpretations:

$$0_{\mathcal{A}} = a \qquad s_{\mathcal{A}}(x) = bx + c \qquad +_{\mathcal{A}}(x, y) = dx + ey + f$$

- Polynomial constraints: $\forall X, Y \in \mathbb{N}$

$$(e - 1)Y + da + f > 0$$
$$(e - be)Y + dc + f - bf - c > 0$$
$$a \geq 0 \quad b \geq 1 \quad c \geq 0 \quad d \geq 1 \quad e \geq 1 \quad f \geq 0$$

Polynomial Orders

Example 4.5

- Rewrite system:

$$\{0 + y \to y, \quad s(x) + y \to s(x + y)\}$$

- Interpretations:

$$0_{\mathcal{A}} = a \qquad s_{\mathcal{A}}(x) = bx + c \qquad +_{\mathcal{A}}(x, y) = dx + ey + f$$

- Diophantine constraints:

$$\begin{aligned} e - 1 \geq 0 \quad & da + f > 0 \\ (e - be) \geq 0 \quad & dc + f - bf - c > 0 \end{aligned}$$

$$a \geq 0 \quad b \geq 1 \quad c \geq 0 \quad d \geq 1 \quad e \geq 1 \quad f \geq 0$$

Polynomial Orders

Example 4.5

- Rewrite system:

$$\{0 + y \to y, \quad s(x) + y \to s(x + y)\}$$

- Interpretations:

$$0_{\mathcal{A}} = a \qquad s_{\mathcal{A}}(x) = bx + c \qquad +_{\mathcal{A}}(x, y) = dx + ey + f$$

- Diophantine constraints:

$$\begin{array}{r} e - 1 \geq 0 \quad da + f > 0 \\ (e - be) \geq 0 \quad dc + f - bf - c > 0 \\ a \geq 0 \quad b \geq 1 \quad c \geq 0 \quad d \geq 1 \quad e \geq 1 \quad f \geq 0 \end{array}$$

- Possible solution: $a = 0 \quad b = 1 \quad c = 1 \quad d = 2 \quad e = 1 \quad f = 1$

Polynomial Orders

Example 4.5

- Rewrite system:

$$\{0 + y \to y, \quad s(x) + y \to s(x + y)\}$$

- Interpretations:

$$0_{\mathcal{A}} = 0 \qquad s_{\mathcal{A}}(x) = bx + c \qquad +_{\mathcal{A}}(x, y) = dx + ey + f$$

- Diophantine constraints:

$$\begin{array}{rr} e - 1 \geq 0 & da + f > 0 \\ (e - be) \geq 0 & dc + f - bf - c > 0 \end{array}$$

$$a \geq 0 \quad b \geq 1 \quad c \geq 0 \quad d \geq 1 \quad e \geq 1 \quad f \geq 0$$

- Possible solution: $a = 0 \quad b = 1 \quad c = 1 \quad d = 2 \quad e = 1 \quad f = 1$

Simplification Orders

Motivation: construct reduction orders $>$ for which $s >^? t$ is decidable.

Simplification Orders

Motivation: construct reduction orders $>$ for which $s >^? t$ is decidable.

Definition 4.7

A strict order $>$ on $T(\mathcal{F}, \mathcal{V})$ is called a simplification order iff it is

1. compatible with $\mathcal{F}$-operations: If $s_1 > s_2$, then

   $$f(t_1, \ldots, t_{i-1}, s_1, t_{i+1}, \ldots, t_n) > f(t_1, \ldots, t_{i-1}, s_2, t_{i+1}, \ldots, t_n)$$

   for all $t_1, \ldots, t_{i-1}, s_1, s_2, t_{i+1}, \ldots, t_n \in T(\mathcal{F}, \mathcal{V})$ and $f \in \mathcal{F}^n$,
2. closed under substitutions: If $s_1 > s_2$, then $\sigma(s_1) > \sigma(s_2)$ for all $s_1, s_2 \in T(\mathcal{F}, \mathcal{V})$ and a $T(\mathcal{F}, \mathcal{V})$-substitution σ,
3. satisfies subterm property: $t > t|_p$ for all terms $t \in T(\mathcal{F}, \mathcal{V})$ and all positions $p \in \mathcal{P}os(t) \setminus \{\epsilon\}$.

Simplification Orders

- Our goal is to show that simplification orders are reduction orders (and, thus, can be used to prove termination)
- First we introduce some notions.

Homeomorphic Embedding

Definition 4.8

The homeomorphic embedding $\trianglerighteq_{emb}$ is defined as the reduction relation $\xrightarrow{*}_{R_{emb}}$ induced by the rewrite system

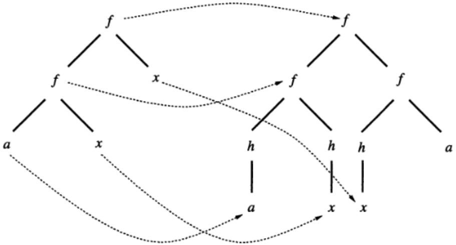
$$R_{emb} := \{f(x_1, \ldots, x_n) \to x_i \mid n \geq 1, f \in \mathcal{F}^n, 1 \leq i \leq n\}.$$

Homeomorphic Embedding

Definition 4.8

The homeomorphic embedding $\trianglerighteq_{emb}$ is defined as the reduction relation $\xrightarrow{*}_{R_{emb}}$ induced by the rewrite system

$$R_{emb} := \{f(x_1, \ldots, x_n) \to x_i \mid n \geq 1, f \in \mathcal{F}^n, 1 \leq i \leq n\}.$$



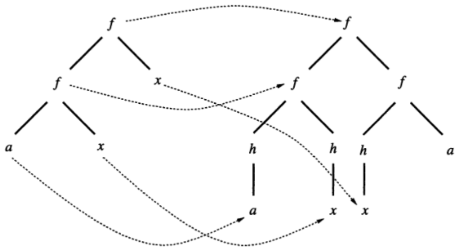
$$f(f(a, x), x) \trianglelefteq_{emb} f(f(h(a), h(x)), f(h(x), a))$$

Homeomorphic Embedding

Definition 4.8

The homeomorphic embedding $\trianglerighteq_{emb}$ is defined as the reduction relation $\xrightarrow{*}_{R_{emb}}$ induced by the rewrite system

$$R_{emb} := \{f(x_1, \ldots, x_n) \to x_i \mid n \geq 1, f \in \mathcal{F}^n, 1 \leq i \leq n\}.$$



$$f(f(a, x), x) \trianglelefteq_{emb} f(f(h(a), h(x)), f(h(x), a))$$

Since R_{emb} is terminating, $\trianglerighteq_{emb}$ is a well-founded partial order.

Well-Partial-Orders, Kruskal's Theorem

Definition 4.9

A partial order $\preceq$ on a set A is a well-partial-order (wpo) iff for every infinite sequence $a_1, a_2, \ldots$ of elements of A there exist indices $i < j$ such that $a_i \preceq a_j$.

Well-Partial-Orders, Kruskal's Theorem

Definition 4.9

A partial order $\preceq$ on a set A is a well-partial-order (wpo) iff for every infinite sequence $a_1, a_2, \ldots$ of elements of A there exist indices $i < j$ such that $a_i \preceq a_j$.

Wpos forbid

- infinite descending chains, and
- infinite anti-chains (infinite sets of incomparable elements).

Well-Partial-Orders, Kruskal's Theorem

Definition 4.9

A partial order $\preceq$ on a set A is a well-partial-order (wpo) iff for every infinite sequence $a_1, a_2, \ldots$ of elements of A there exist indices $i < j$ such that $a_i \preceq a_j$.

Wpos forbid

- infinite descending chains, and
- infinite anti-chains (infinite sets of incomparable elements).

Theorem 4.5 (Kruskal)

For finite $\mathcal{F}$ and $\mathcal{V}$, the relation $\trianglerighteq_{emb}$ is a wpo on $T(\mathcal{F}, \mathcal{V})$.

Homeomorphic Embedding

Lemma 4.2

Let $>$ be a simplification order on $T(\mathcal{F},\mathcal{V})$ and let $s,t \in T(\mathcal{F},\mathcal{V})$. Then $s \trianglerighteq_{emb} t$ implies $s \geq t$.

Proof.

Since $>$ satisfies the subterm property, we have $f(x_1,\dots,x_i,\dots,x_n) > x_i$ for all $n \geq 1$, $f \in \mathcal{F}^n$, $1 \leq i \leq n$.
Therefore, $R_{emb} \subseteq >$.
Since $\geq$ is reflexive, transitive, closed under substitutions and compatible with $\mathcal{F}$-operations, this implies

$$\trianglerighteq_{emb} = \xrightarrow{*}_{R_{emb}} \subseteq \geq .$$

□

Simplification Orders Are Reduction Orders

Theorem 4.6

Let $\mathcal{F}$ be a finite signature. Then every simplification order on $T(\mathcal{F}, \mathcal{V})$ is a reduction order.

Simplification Orders Are Reduction Orders

Theorem 4.6

Let $\mathcal{F}$ be a finite signature. Then every simplification order on $T(\mathcal{F}, \mathcal{V})$ is a reduction order.

Proof.

We just need to show that every simplification order is well-founded. Assume the opposite: Let $t_1 > t_2 > \cdots$ be an infinite descending chain in $T(\mathcal{F}, \mathcal{V})$, where $>$ is a simplification ordering.

Simplification Orders Are Reduction Orders

Theorem 4.6

Let $\mathcal{F}$ be a finite signature. Then every simplification order on $T(\mathcal{F}, \mathcal{V})$ is a reduction order.

Proof (cont.)

1. Prove by contradiction that $\mathcal{V}ar(t_1) \supseteq \mathcal{V}ar(t_2) \supseteq \cdots$.
   Assume $x \in \mathcal{V}ar(t_{i+1}) \setminus \mathcal{V}ar(t_i)$ and let $\sigma := \{x \mapsto t_i\}$. Then

   $$\begin{aligned} \sigma(t_i) &> \sigma(t_{i+1}) && (> \text{ is closed under substitutions}) \\ \sigma(t_{i+1}) &\geq t_i && (t_i \text{ is a subterm of } \sigma(t_{i+1})) \\ t_i &= \sigma(t_i) && (x \notin \mathcal{V}ar(t_i)) \end{aligned}$$

   Hence, $\sigma(t_i) > \sigma(t_i)$: a contradiction.
   We get $t_1, t_2, \ldots \in T(\mathcal{F}, \mathcal{X})$ for a finite $\mathcal{X} = \mathcal{V}ar(t_1)$.

Simplification Orders Are Reduction Orders

Theorem 4.6

Let $\mathcal{F}$ be a finite signature. Then every simplification order on $T(\mathcal{F}, \mathcal{V})$ is a reduction order.

Proof (cont.)

2. We got $t_1, t_2, \ldots \in T(\mathcal{F}, \mathcal{X})$ for a finite $\mathcal{X} = \mathcal{V}ar(t_1)$.
   Kruskal's Theorem implies that there exist $i < j$ such that $t_j \trianglerighteq_{emb} t_i$.
   Lemma 4.2 implies $t_i \leq t_j$, which is a contradiction since we know that $t_i > t_{i+1} > \cdots > t_j$.

The obtained contradiction shows that $>$ is well-founded.

Not All Reduction Orders Are Simplification Orders

Example 4.6

Let $\mathcal{F} = \{f, g\}$, where f and g are unary. Consider the TRS

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

- R terminates (why?). Therefore, $\xrightarrow{+}_R$ is a reduction order.
- Show that $\xrightarrow{+}_R$ is not a simplification order.
- Assume the opposite. Then from $f(g(f(x))) \trianglerighteq_{emb} f(f(x))$, by Lemma 4.2, we have $f(g(f(x))) \xrightarrow{*}_R f(f(x))$.
- $f(g(f(x))) \xrightarrow{*}_R f(f(x))$ and $f(f(x)) \to f(g(f(x)))$ imply that R is non-terminating: a contradiction.

Hence, $\xrightarrow{+}_R$ is a reduction order, which is not a simplification order.

Lexicographic Path Order

Main idea behind recursive path orders:

- Two terms are compared by first comparing their root symbols.
- Then recursively comparing the collections of their immediate subterms.

Lexicographic Path Order

Main idea behind recursive path orders:

- Two terms are compared by first comparing their root symbols.
- Then recursively comparing the collections of their immediate subterms.
- Collections seen as multisets yields the multiset path order. (Not considered in this course.)

Lexicographic Path Order

Main idea behind recursive path orders:

- Two terms are compared by first comparing their root symbols.
- Then recursively comparing the collections of their immediate subterms.
- Collections seen as multisets yields the multiset path order. (Not considered in this course.)
- Collections seen as tuples yields the lexicographic path order.

Lexicographic Path Order

Main idea behind recursive path orders:

- Two terms are compared by first comparing their root symbols.
- Then recursively comparing the collections of their immediate subterms.
- Collections seen as multisets yields the multiset path order. (Not considered in this course.)
- Collections seen as tuples yields the lexicographic path order.
- Combination of multisets and tuples yields the recursive path order with status. (Not considered in this course.)

Lexicographic Path Order

Definition 4.10

Let $\mathcal{F}$ be a finite signature and $>$ be a strict order on $\mathcal{F}$ (called the precedence). The lexicographic path order $>_{lpo}$on $T(\mathcal{F}, \mathcal{V})$ induced by $>$ is defined as follows:

$s >_{lpo} t$ iff

(LPO1) $t \in \mathcal{V}ar(s)$ and $t \neq s$, or

(LPO2) $s = f(s_1, \ldots, s_m)$, $t = g(t_1, \ldots, t_n)$, and

- (LPO2a) $s_i \geq_{lpo} t$ for some i, $1 \leq i \leq m$, or
- (LPO2b) $f > g$ and $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, or
- (LPO2c) $f = g$, $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, and there exists i, $1 \leq i \leq m$ such that $s_1 = t_1, \ldots s_{i-1} = t_{i-1}$ and $s_i >_{lpo} t_i$.

$\geq_{lpo}$ stands for the reflexive closure of $>_{lpo}$.

Lexicographic Path Order

$s >_{lpo} t$ iff

(LPO1) $t \in \mathcal{V}ar(s)$ and $t \neq s$, or

(LPO2) $s = f(s_1, \ldots, s_m)$, $t = g(t_1, \ldots, t_n)$, and

- (LPO2a) $s_i \geq_{lpo} t$ for some i, $1 \leq i \leq m$, or
- (LPO2b) $f > g$ and $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, or
- (LPO2c) $f = g$, $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, and there exists i, $1 \leq i \leq m$ such that $s_1 = t_1, \ldots s_{i-1} = t_{i-1}$ and $s_i >_{lpo} t_i$.

Example 4.7

$\mathcal{F} = \{f, i, e\}$, f is binary, i is unary, e is constant, with $i > f > e$.

Lexicographic Path Order

$s >_{lpo} t$ iff

(LPO1) $t \in \mathcal{V}ar(s)$ and $t \neq s$, or

(LPO2) $s = f(s_1, \ldots, s_m)$, $t = g(t_1, \ldots, t_n)$, and

(LPO2a) $s_i \geq_{lpo} t$ for some i, $1 \leq i \leq m$, or

(LPO2b) $f > g$ and $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, or

(LPO2c) $f = g$, $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, and there exists i, $1 \leq i \leq m$ such that $s_1 = t_1, \ldots s_{i-1} = t_{i-1}$ and $s_i >_{lpo} t_i$.

Example 4.7

$\mathcal{F} = \{f, i, e\}$, f is binary, i is unary, e is constant, with $i > f > e$.

- $f(x, e) >_{lpo} x$ by (LPO1)

Lexicographic Path Order

$s >_{lpo} t$ iff

(LPO1) $t \in \mathcal{V}ar(s)$ and $t \neq s$, or

(LPO2) $s = f(s_1, \ldots, s_m)$, $t = g(t_1, \ldots, t_n)$, and

(LPO2a) $s_i \geq_{lpo} t$ for some i, $1 \leq i \leq m$, or

(LPO2b) $f > g$ and $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, or

(LPO2c) $f = g$, $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, and there exists i, $1 \leq i \leq m$ such that $s_1 = t_1, \ldots s_{i-1} = t_{i-1}$ and $s_i >_{lpo} t_i$.

Example 4.7

$\mathcal{F} = \{f, i, e\}$, f is binary, i is unary, e is constant, with $i > f > e$.

- $f(x, e) >_{lpo} x$ by (LPO1)
- $i(e) >_{lpo} e$ by (LPO2), because $e \geq_{lpo} e$.

Lexicographic Path Order

$s >_{lpo} t$ iff

(LPO1) $t \in \mathcal{V}ar(s)$ and $t \neq s$, or

(LPO2) $s = f(s_1, \ldots, s_m)$, $t = g(t_1, \ldots, t_n)$, and

- (LPO2a) $s_i \geq_{lpo} t$ for some i, $1 \leq i \leq m$, or
- (LPO2b) $f > g$ and $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, or
- (LPO2c) $f = g$, $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, and there exists i, $1 \leq i \leq m$ such that $s_1 = t_1, \ldots s_{i-1} = t_{i-1}$ and $s_i >_{lpo} t_i$.

Example 4.7 (Cont.)

$\mathcal{F} = \{f, i, e\}$, f is binary, i is unary, e is constant, with $i > f > e$.

Lexicographic Path Order

$s >_{lpo} t$ iff

(LPO1) $t \in \mathcal{V}ar(s)$ and $t \neq s$, or

(LPO2) $s = f(s_1, \ldots, s_m)$, $t = g(t_1, \ldots, t_n)$, and

(LPO2a) $s_i \geq_{lpo} t$ for some i, $1 \leq i \leq m$, or

(LPO2b) $f > g$ and $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, or

(LPO2c) $f = g$, $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, and there exists i, $1 \leq i \leq m$ such that $s_1 = t_1, \ldots s_{i-1} = t_{i-1}$ and $s_i >_{lpo} t_i$.

Example 4.7 (Cont.)

$\mathcal{F} = \{f, i, e\}$, f is binary, i is unary, e is constant, with $i > f > e$.

- $i(f(x, y)) >^{?}_{lpo} f(i(x), i(y))$:

Lexicographic Path Order

$s >_{lpo} t$ iff

(LPO1) $t \in \mathcal{V}ar(s)$ and $t \neq s$, or

(LPO2) $s = f(s_1, \ldots, s_m)$, $t = g(t_1, \ldots, t_n)$, and

(LPO2a) $s_i \geq_{lpo} t$ for some i, $1 \leq i \leq m$, or

(LPO2b) $f > g$ and $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, or

(LPO2c) $f = g$, $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, and there exists i, $1 \leq i \leq m$ such that $s_1 = t_1, \ldots s_{i-1} = t_{i-1}$ and $s_i >_{lpo} t_i$.

Example 4.7 (Cont.)

$\mathcal{F} = \{f, i, e\}$, f is binary, i is unary, e is constant, with $i > f > e$.

- $i(f(x, y)) >^?_{lpo} f(i(x), i(y))$:
  - Since $i > f$, (LPO2b) reduces it to the problems: $i(f(x, y)) >^?_{lpo} i(x)$ and $i(f(x, y)) >^?_{lpo} i(y)$.

Lexicographic Path Order

$s >_{lpo} t$ iff

(LPO1) $t \in \mathcal{V}ar(s)$ and $t \neq s$, or

(LPO2) $s = f(s_1, \ldots, s_m)$, $t = g(t_1, \ldots, t_n)$, and

(LPO2a) $s_i \geq_{lpo} t$ for some i, $1 \leq i \leq m$, or

(LPO2b) $f > g$ and $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, or

(LPO2c) $f = g$, $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, and there exists i, $1 \leq i \leq m$ such that $s_1 = t_1, \ldots s_{i-1} = t_{i-1}$ and $s_i >_{lpo} t_i$.

Example 4.7 (Cont.)

$\mathcal{F} = \{f, i, e\}$, f is binary, i is unary, e is constant, with $i > f > e$.

- $i(f(x, y)) >^?_{lpo} f(i(x), i(y))$:
  - Since $i > f$, (LPO2b) reduces it to the problems: $i(f(x, y)) >^?_{lpo} i(x)$ and $i(f(x, y)) >^?_{lpo} i(y)$.
  - $i(f(x, y)) >^?_{lpo} i(x)$ is reduced by (LPO2c) to $i(f(x, y)) >^?_{lpo} x$ and $f(x, y) >^?_{lpo} x$, which hold by (LPO1).

Lexicographic Path Order

$s >_{lpo} t$ iff

(LPO1) $t \in \mathcal{V}ar(s)$ and $t \neq s$, or

(LPO2) $s = f(s_1, \dots, s_m)$, $t = g(t_1, \dots, t_n)$, and

(LPO2a) $s_i \geq_{lpo} t$ for some i, $1 \leq i \leq m$, or

(LPO2b) $f > g$ and $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, or

(LPO2c) $f = g$, $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, and there exists i, $1 \leq i \leq m$ such that $s_1 = t_1, \dots s_{i-1} = t_{i-1}$ and $s_i >_{lpo} t_i$.

Example 4.7 (Cont.)

$\mathcal{F} = \{f, i, e\}$, f is binary, i is unary, e is constant, with $i > f > e$.

- $i(f(x,y)) >^?_{lpo} f(i(x), i(y))$:
  - Since $i > f$, (LPO2b) reduces it to the problems:
    $i(f(x,y)) >^?_{lpo} i(x)$ and $i(f(x,y)) >^?_{lpo} i(y)$.
  - $i(f(x,y)) >^?_{lpo} i(x)$ is reduced by (LPO2c) to
    $i(f(x,y)) >^?_{lpo} x$ and $f(x,y) >^?_{lpo} x$, which hold by (LPO1).
  - $i(f(x,y)) >_{lpo} i(y)$ is shown similarly.

Lexicographic Path Order

$s >_{lpo} t$ iff

(LPO1) $t \in \mathcal{V}ar(s)$ and $t \neq s$, or

(LPO2) $s = f(s_1, \ldots, s_m)$, $t = g(t_1, \ldots, t_n)$, and

(LPO2a) $s_i \geq_{lpo} t$ for some i, $1 \leq i \leq m$, or

(LPO2b) $f > g$ and $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, or

(LPO2c) $f = g$, $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, and there exists i, $1 \leq i \leq m$ such that $s_1 = t_1, \ldots s_{i-1} = t_{i-1}$ and $s_i >_{lpo} t_i$.

Example 4.7 (Cont.)

$\mathcal{F} = \{f, i, e\}$, f is binary, i is unary, e is constant, with $i > f > e$.

Lexicographic Path Order

$s >_{lpo} t$ iff

(LPO1) $t \in \mathcal{V}ar(s)$ and $t \neq s$, or

(LPO2) $s = f(s_1, \dots, s_m)$, $t = g(t_1, \dots, t_n)$, and

- (LPO2a) $s_i \geq_{lpo} t$ for some i, $1 \leq i \leq m$, or
- (LPO2b) $f > g$ and $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, or
- (LPO2c) $f = g$, $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, and there exists i, $1 \leq i \leq m$ such that $s_1 = t_1, \dots s_{i-1} = t_{i-1}$ and $s_i >_{lpo} t_i$.

Example 4.7 (Cont.)

$\mathcal{F} = \{f, i, e\}$, f is binary, i is unary, e is constant, with $i > f > e$.

- $f(f(x, y), z) >^?_{lpo} f(x, f(y, z)))$. By (LPO2c) with $i = 1$:

Lexicographic Path Order

$s >_{lpo} t$ iff

(LPO1) $t \in \mathcal{V}ar(s)$ and $t \neq s$, or

(LPO2) $s = f(s_1, \ldots, s_m)$, $t = g(t_1, \ldots, t_n)$, and

(LPO2a) $s_i \geq_{lpo} t$ for some i, $1 \leq i \leq m$, or

(LPO2b) $f > g$ and $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, or

(LPO2c) $f = g$, $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, and there exists i, $1 \leq i \leq m$ such that $s_1 = t_1, \ldots s_{i-1} = t_{i-1}$ and $s_i >_{lpo} t_i$.

Example 4.7 (Cont.)

$\mathcal{F} = \{f, i, e\}$, f is binary, i is unary, e is constant, with $i > f > e$.

- $f(f(x, y), z) >^?_{lpo} f(x, f(y, z)))$. By (LPO2c) with $i = 1$:
  - $f(f(x, y), z) >_{lpo} x$ because of (LPO1).

Lexicographic Path Order

$s >_{lpo} t$ iff

(LPO1) $t \in \mathcal{V}ar(s)$ and $t \neq s$, or

(LPO2) $s = f(s_1, \ldots, s_m)$, $t = g(t_1, \ldots, t_n)$, and

(LPO2a) $s_i \geq_{lpo} t$ for some i, $1 \leq i \leq m$, or

(LPO2b) $f > g$ and $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, or

(LPO2c) $f = g$, $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, and there exists i, $1 \leq i \leq m$ such that $s_1 = t_1, \ldots s_{i-1} = t_{i-1}$ and $s_i >_{lpo} t_i$.

Example 4.7 (Cont.)

$\mathcal{F} = \{f, i, e\}$, f is binary, i is unary, e is constant, with $i > f > e$.

- $f(f(x, y), z) >^{?}_{lpo} f(x, f(y, z)))$. By (LPO2c) with $i = 1$:
  - $f(f(x, y), z) >_{lpo} x$ because of (LPO1).
  - $f(f(x, y), z) >^{?}_{lpo} f(y, z)$: By (LPO2c) with $i = 1$:
    - $f(f(x, y), z) >_{lpo} y$ and $f(f(x, y), z) >_{lpo} z$ by (LPO1).
    - $f(x, y) >_{lpo} y$ by (LPO1).

Lexicographic Path Order

$s >_{lpo} t$ iff

(LPO1) $t \in \mathcal{V}ar(s)$ and $t \neq s$, or

(LPO2) $s = f(s_1, \ldots, s_m)$, $t = g(t_1, \ldots, t_n)$, and

(LPO2a) $s_i \geq_{lpo} t$ for some i, $1 \leq i \leq m$, or

(LPO2b) $f > g$ and $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, or

(LPO2c) $f = g$, $s >_{lpo} t_j$ for all j, $1 \leq j \leq n$, and there exists i, $1 \leq i \leq m$ such that $s_1 = t_1, \ldots s_{i-1} = t_{i-1}$ and $s_i >_{lpo} t_i$.

Example 4.7 (Cont.)

$\mathcal{F} = \{f, i, e\}$, f is binary, i is unary, e is constant, with $i > f > e$.

- $f(f(x,y),z) >^?_{lpo} f(x, f(y,z)))$. By (LPO2c) with $i = 1$:
  - $f(f(x,y),z) >_{lpo} x$ because of (LPO1).
  - $f(f(x,y),z) >^?_{lpo} f(y,z)$: By (LPO2c) with $i = 1$:
    - $f(f(x,y),z) >_{lpo} y$ and $f(f(x,y),z) >_{lpo} z$ by (LPO1).
    - $f(x,y) >_{lpo} y$ by (LPO1).
  - $f(x,y) >_{lpo} x$ by (LPO1).

LPO Is a Simplification Order

Theorem 4.7

For any strict order $>$ on $\mathcal{F}$, the induced lexicographic path order $>_{lpo}$ is a simplification order on $T(\mathcal{F}, \mathcal{V})$.

Proof.

See Baader and Nipkow, pp. 119–120. □

Properties of LPO

For a finite signature $\mathcal{F}$, terms $s, t \in T(\mathcal{F}, \mathcal{V})$, finite TRS R over $T(\mathcal{F}, \mathcal{V})$:

- For a given lpo $>_{lpo}$, the question whether $s >_{lpo} t$ can be decided in time polynomial in the size s and t.
- The question whether termination of R can be shown by some lpo $T(\mathcal{F}, \mathcal{V})$ is an NP-complete problem.

LPO and Polynomial Interpretations Are Not Comparable