# Automated Reasoning

## Week 9. Term Rewriting

Cynthia Kop

Fall 2024

# Beyond predicate logic

# Beyond predicate logic

Recall: predicate logic

$$(\exists x[\mathtt{S}(x) \wedge \forall y[\mathtt{L}(y) \rightarrow \mathtt{A}(x,y)]]) \wedge$$

$$(\forall x[(\mathtt{L}(x) \wedge \mathtt{B}(x)) \rightarrow \neg\exists y[\mathtt{S}(y) \wedge \mathtt{A}(y,x)]]) \rightarrow$$

$$\neg\exists x[\mathtt{L}(x) \wedge \mathtt{B}(x)]$$

# Beyond predicate logic

Recall: predicate logic

$$(\exists x[\mathtt{S}(x) \wedge \forall y[\mathtt{L}(y) \rightarrow \mathtt{A}(x,y)]]) \wedge$$

$$(\forall x[(\mathtt{L}(x) \wedge \mathtt{B}(x)) \rightarrow \neg\exists y[\mathtt{S}(y) \wedge \mathtt{A}(y,x)]]) \ \rightarrow$$

$$\neg\exists x[\mathtt{L}(x) \wedge \mathtt{B}(x)]$$

In practice: equality is important!

$$\forall x[\forall y[\mathtt{suc}(x) = \mathtt{suc}(y) \rightarrow x = y]]$$

$$\exists x[\exists y[x \neq y \wedge \mathtt{Favourite}(x) = \mathtt{AR} \wedge \mathtt{Favourite}(y) = \mathtt{AR}]]$$

# Beyond predicate logic

Recall: predicate logic

$$(\exists x[\mathtt{S}(x) \wedge \forall y[\mathtt{L}(y) \rightarrow \mathtt{A}(x,y)]]) \wedge$$

$$(\forall x[(\mathtt{L}(x) \wedge \mathtt{B}(x)) \rightarrow \neg\exists y[\mathtt{S}(y) \wedge \mathtt{A}(y,x)]]) \rightarrow$$

$$\neg\exists x[\mathtt{L}(x) \wedge \mathtt{B}(x)]$$

In practice: equality is important!

$$\forall x[\forall y[\mathtt{suc}(x) = \mathtt{suc}(y) \rightarrow x = y]]$$

$$\exists x[\exists y[x \neq y \wedge \mathtt{Favourite}(x) = \mathtt{AR} \wedge \mathtt{Favourite}(y) = \mathtt{AR}]]$$

Prerequisite: **term rewriting.**

# Program analysis

Recall: analysing a for loop

# Program analysis

Recall: analysing a for loop

$a := 0$;
for $i := 1$ to $m$ do $a := a + k$

# Program analysis

Recall: analysing a for loop

$a := 0;$
for $i := 1$ to $m$ do $a := a + k$

$a = m * k$ follows by unsatisfiability of:

$$\bigwedge_{j=1}^{n} \neg a_{0,j} \quad \wedge \quad \bigwedge_{i=0}^{m-1} \mathsf{plus}(\vec{a}_i, \vec{k}, \vec{a}_{i+1}) \quad \wedge \neg \mathsf{mul}([\vec{m}], \vec{k}, \vec{a}_m)$$

# Program analysis

Recall: analysing a for loop

$a := 0;$
for $i := 1$ to $m$ do $a := a + k$

$a = m * k$ follows by unsatisfiability of:

$$\bigwedge_{j=1}^{n} \neg a_{0,j} \quad \wedge \quad \bigwedge_{i=0}^{m-1} \mathsf{plus}(\vec{a}_i, \vec{k}, \vec{a}_{i+1}) \quad \wedge \neg \mathsf{mul}([\vec{m}], \vec{k}, \vec{a}_m)$$

Challenge: analysing functional programs

# Program analysis

Recall: analysing a for loop

$a := 0;$
for $i := 1$ to $m$ do $a := a + k$

$a = m * k$ follows by unsatisfiability of:

$$\bigwedge_{j=1}^{n} \neg a_{0,j} \quad \wedge \quad \bigwedge_{i=0}^{m-1} \mathsf{plus}(\vec{a}_i, \vec{k}, \vec{a}_{i+1}) \quad \wedge \neg \mathsf{mul}([\vec{m}], \vec{k}, \vec{a}_m)$$

Challenge: analysing functional programs

```
let rec conc xs ys =
    match xs with
        | [] ⇒ ys
        | h :: t ⇒ h :: (conc t ys)
;;
```

```
let rec rev xs =
    match xs with
        | [] ⇒ []
        | h :: t ⇒ conc (rev t) [h]
;;
```

# Defining terms

Given: a set $\mathcal{V}ar$ of variables: $\{x,\ y,\ z,\ \ldots\}$

# Defining terms

Given: a set $\mathcal{V}ar$ of variables: $\{x,\ y,\ z,\ \ldots\}$

Given: a set $\Sigma$ of **function symbols**: $\{\mathtt{f},\ \mathtt{g},\ \mathtt{h},\ \ldots\}$

# Defining terms

Given: a set $\mathcal{V}ar$ of variables: $\{x, y, z, \ldots\}$

Given: a set $\Sigma$ of **function symbols**: $\{\mathtt{f}, \mathtt{g}, \mathtt{h}, \ldots\}$

Given: an **arity** function, that maps each function symbol to a non-negative integer

# Defining terms

Given: a set $\mathcal{V}ar$ of variables: $\{x,\ y,\ z,\ \ldots\}$

Given: a set $\Sigma$ of **function symbols**: $\{\mathtt{f},\ \mathtt{g},\ \mathtt{h},\ \ldots\}$

Given: an **arity** function, that maps each function symbol to a non-negative integer

- every variable is a term

# Defining terms

Given: a set $\mathcal{V}ar$ of variables: $\{x,\ y,\ z,\ \ldots\}$

Given: a set $\Sigma$ of **function symbols**: $\{\mathtt{f},\ \mathtt{g},\ \mathtt{h},\ \ldots\}$

Given: an **arity** function, that maps each function symbol to a non-negative integer

- every variable is a term
- if $\mathtt{f} \in \Sigma$ and $arity(\mathtt{f}) = n$ and $s_1, \ldots, s_n$ are terms, then $\mathtt{f}(s_1, \ldots, s_n)$ is a term

# Defining terms

Example:

$$\Sigma = \{\texttt{0},\ \texttt{s},\ \texttt{add},\ \texttt{mul}\}$$

# Defining terms

**Example:**

$$\Sigma = \{\mathtt{0},\ \mathtt{s},\ \mathtt{add},\ \mathtt{mul}\}$$

$$\begin{array}{rcl} arity(\mathtt{0}) & = & 0 \\ arity(\mathtt{s}) & = & 1 \\ arity(\mathtt{add}) & = & 2 \\ arity(\mathtt{mul}) & = & 2 \end{array}$$

# Defining terms

Example:

$$\Sigma = \{\texttt{0},\ \texttt{s},\ \texttt{add},\ \texttt{mul}\}$$

$$\begin{aligned}
arity(\texttt{0}) &= 0\\
arity(\texttt{s}) &= 1\\
arity(\texttt{add}) &= 2\\
arity(\texttt{mul}) &= 2
\end{aligned}$$

Terms are for instance:

# Defining terms

Example:

$$\Sigma = \{\mathtt{0},\ \mathtt{s},\ \mathtt{add},\ \mathtt{mul}\}$$

$$\begin{aligned} arity(\mathtt{0}) &= 0 \\ arity(\mathtt{s}) &= 1 \\ arity(\mathtt{add}) &= 2 \\ arity(\mathtt{mul}) &= 2 \end{aligned}$$

Terms are for instance:

- $\mathtt{0}$, $\mathtt{s}(\mathtt{0})$, $\mathtt{s}(\mathtt{s}(\mathtt{0}))$, $\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{0})))$, $\ldots$

# Defining terms

Example:

$$\Sigma = \{\mathtt{0},\ \mathtt{s},\ \mathtt{add},\ \mathtt{mul}\}$$

$$\begin{array}{rcl} arity(\mathtt{0}) & = & 0 \\ arity(\mathtt{s}) & = & 1 \\ arity(\mathtt{add}) & = & 2 \\ arity(\mathtt{mul}) & = & 2 \end{array}$$

Terms are for instance:

- $\mathtt{0}$, $\mathtt{s}(\mathtt{0})$, $\mathtt{s}(\mathtt{s}(\mathtt{0}))$, $\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{0})))$, $\ldots$
- $\mathtt{s}(\mathtt{s}(\mathtt{s}(x)))$

# Defining terms

Example:

$$\Sigma = \{\mathtt{0},\ \mathtt{s},\ \mathtt{add},\ \mathtt{mul}\}$$

$$\begin{array}{rcl} arity(\mathtt{0}) & = & 0 \\ arity(\mathtt{s}) & = & 1 \\ arity(\mathtt{add}) & = & 2 \\ arity(\mathtt{mul}) & = & 2 \end{array}$$

Terms are for instance:

- $\mathtt{0}$, $\mathtt{s}(\mathtt{0})$, $\mathtt{s}(\mathtt{s}(\mathtt{0}))$, $\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{0})))$, ...
- $\mathtt{s}(\mathtt{s}(\mathtt{s}(x)))$
- $\mathtt{add}(\mathtt{s}(\mathtt{0}), \mathtt{add}(y, \mathtt{s}(\mathtt{add}(\mathtt{0}, x))))$

# Defining terms

Example:

$$\Sigma = \{\mathtt{0},\ \mathtt{s},\ \mathtt{add},\ \mathtt{mul}\}$$

$$\begin{array}{rcl} arity(\mathtt{0}) & = & 0 \\ arity(\mathtt{s}) & = & 1 \\ arity(\mathtt{add}) & = & 2 \\ arity(\mathtt{mul}) & = & 2 \end{array}$$

Terms are for instance:

- $\mathtt{0}$, $\mathtt{s}(\mathtt{0})$, $\mathtt{s}(\mathtt{s}(\mathtt{0}))$, $\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{0})))$, ...
- $\mathtt{s}(\mathtt{s}(\mathtt{s}(x)))$
- $\mathtt{add}(\mathtt{s}(\mathtt{0}), \mathtt{add}(y, \mathtt{s}(\mathtt{add}(\mathtt{0}, x))))$
- $\mathtt{mul}(\mathtt{add}(\mathtt{0}, \mathtt{0}), \mathtt{s}(y))$

# Class exercise

Design a signature $(\Sigma, arity)$ that contains lists of natural numbers.

# Class exercise

Design a signature $(\Sigma, arity)$ that contains lists of natural numbers.

Alter the signature to handle lists of integers.

# Rules

# Rules

A rule is a pair of terms, denoted $\ell \Rightarrow r$, such that:

- $\ell$ is not a variable
- all variables in $r$ occur also in $\ell$

# Rules

A rule is a pair of terms, denoted $\ell \Rightarrow r$, such that:

- $\ell$ is not a variable
- all variables in $r$ occur also in $\ell$

Examples:

# Rules

A rule is a pair of terms, denoted $\ell \Rightarrow r$, such that:

- $\ell$ is not a variable
- all variables in $r$ occur also in $\ell$

Examples:

- $\mathtt{add}(x, \mathtt{0}) \Rightarrow x$

# Rules

A rule is a pair of terms, denoted $\ell \Rightarrow r$, such that:

- $\ell$ is not a variable
- all variables in $r$ occur also in $\ell$

Examples:

- $\mathtt{add}(x, \mathtt{0}) \Rightarrow x$
- $\mathtt{add}(x, \mathtt{s}(y)) \Rightarrow \mathtt{s}(\mathtt{add}(x, y))$

# Rules

A rule is a pair of terms, denoted $\ell \Rightarrow r$, such that:

- $\ell$ is not a variable
- all variables in $r$ occur also in $\ell$

Examples:

- $\mathtt{add}(x, \mathtt{0}) \Rightarrow x$
- $\mathtt{add}(x, \mathtt{s}(y)) \Rightarrow \mathtt{s}(\mathtt{add}(x, y))$
- $\mathtt{min}(\mathtt{min}(x)) \Rightarrow x$

# Rules

A rule is a pair of terms, denoted $\ell \Rightarrow r$, such that:

- $\ell$ is not a variable
- all variables in $r$ occur also in $\ell$

Examples:

- $\mathtt{add}(x, \mathtt{0}) \Rightarrow x$
- $\mathtt{add}(x, \mathtt{s}(y)) \Rightarrow \mathtt{s}(\mathtt{add}(x, y))$
- $\mathtt{min}(\mathtt{min}(x)) \Rightarrow x$

Non-examples

# Rules

A rule is a pair of terms, denoted $\ell \Rightarrow r$, such that:

- $\ell$ is not a variable
- all variables in $r$ occur also in $\ell$

Examples:

- $\mathtt{add}(x, \mathtt{0}) \Rightarrow x$
- $\mathtt{add}(x, \mathtt{s}(y)) \Rightarrow \mathtt{s}(\mathtt{add}(x, y))$
- $\mathtt{min}(\mathtt{min}(x)) \Rightarrow x$

Non-examples

- $x \Rightarrow \mathtt{min}(\mathtt{min}(x))$

# Rules

A rule is a pair of terms, denoted $\ell \Rightarrow r$, such that:

- $\ell$ is not a variable
- all variables in $r$ occur also in $\ell$

Examples:

- $\mathtt{add}(x, \mathtt{0}) \Rightarrow x$
- $\mathtt{add}(x, \mathtt{s}(y)) \Rightarrow \mathtt{s}(\mathtt{add}(x, y))$
- $\mathtt{min}(\mathtt{min}(x)) \Rightarrow x$

Non-examples

- $x \Rightarrow \mathtt{min}(\mathtt{min}(x))$
- $\mathtt{mul}(x, \mathtt{0}) \Rightarrow \mathtt{mul}(y, \mathtt{0})$

# Reducing terms

**Intuition:**

$\mathtt{min}(\mathtt{add}(\mathtt{s}(\mathtt{0}), \mathtt{s}(\mathtt{s}(\mathtt{0}))))$ **rewrites** to $\mathtt{min}(\mathtt{s}(\mathtt{add}(\mathtt{s}(\mathtt{0}), \mathtt{s}(\mathtt{0}))))$ using the rule $\mathtt{add}(x, \mathtt{s}(y)) \Rightarrow \mathtt{s}(\mathtt{add}(x, y))$.

# Reducing terms

Intuition:
$\mathtt{min}(\mathtt{add}(\mathtt{s}(\mathtt{0}), \mathtt{s}(\mathtt{s}(\mathtt{0}))))$ **rewrites** to $\mathtt{min}(\mathtt{s}(\mathtt{add}(\mathtt{s}(\mathtt{0}), \mathtt{s}(\mathtt{0}))))$
using the rule $\mathtt{add}(x, \mathtt{s}(y)) \Rightarrow \mathtt{s}(\mathtt{add}(x, y))$.

Ingredient: a **substitution** is a function from variables to terms.

# Reducing terms

Intuition:
$\mathtt{min}(\mathtt{add}(\mathtt{s}(\mathtt{0}), \mathtt{s}(\mathtt{s}(\mathtt{0}))))$ **rewrites** to $\mathtt{min}(\mathtt{s}(\mathtt{add}(\mathtt{s}(\mathtt{0}), \mathtt{s}(\mathtt{0}))))$ using the rule $\mathtt{add}(x, \mathtt{s}(y)) \Rightarrow \mathtt{s}(\mathtt{add}(x, y))$.

Ingredient: a **substitution** is a function from variables to terms.

$$
\begin{array}{rcl}
x\gamma & = & \gamma(x) \\
\mathtt{f}(s_1, \ldots, s_n)\gamma & = & \mathtt{f}(s_1\gamma, \ldots, s_n\gamma)
\end{array}
$$

# Reducing terms

Intuition:
$\mathtt{min}(\mathtt{add}(\mathtt{s}(\mathtt{0}), \mathtt{s}(\mathtt{s}(\mathtt{0}))))$ rewrites to $\mathtt{min}(\mathtt{s}(\mathtt{add}(\mathtt{s}(\mathtt{0}), \mathtt{s}(\mathtt{0}))))$ using the rule $\mathtt{add}(x, \mathtt{s}(y)) \Rightarrow \mathtt{s}(\mathtt{add}(x, y))$.

Ingredient: a **substitution** is a function from variables to terms.

$$\begin{array}{rcl} x\gamma & = & \gamma(x) \\ \mathtt{f}(s_1, \ldots, s_n)\gamma & = & \mathtt{f}(s_1\gamma, \ldots, s_n\gamma) \end{array}$$

Given: a set of rules $\mathcal{R}$

# Reducing terms

Intuition:
$\mathtt{min}(\mathtt{add}(\mathtt{s}(\mathtt{0}), \mathtt{s}(\mathtt{s}(\mathtt{0}))))$ rewrites to $\mathtt{min}(\mathtt{s}(\mathtt{add}(\mathtt{s}(\mathtt{0}), \mathtt{s}(\mathtt{0}))))$ using the rule $\mathtt{add}(x, \mathtt{s}(y)) \Rightarrow \mathtt{s}(\mathtt{add}(x, y))$.

Ingredient: a **substitution** is a function from variables to terms.

$$\begin{array}{rcl} x\gamma & = & \gamma(x) \\ \mathtt{f}(s_1, \dots, s_n)\gamma & = & \mathtt{f}(s_1\gamma, \dots, s_n\gamma) \end{array}$$

Given: a set of rules $\mathcal{R}$

Define:

- $\ell\gamma \Rightarrow_{\mathcal{R}} r\gamma$ for all $\ell \Rightarrow r \in \mathcal{R}$, all $\gamma$
- $\mathtt{f}(s_1, \dots, s_i, \dots, s_n) \Rightarrow_{\mathcal{R}} \mathtt{f}(s_1, \dots, t_i, \dots, s_n)$ if $s_i \Rightarrow_{\mathcal{R}} t_i$

# Reducing terms

Ingredient: a **substitution** is a function from variables to terms.

$$\begin{array}{rcl} x\gamma & = & \gamma(x) \\ \mathtt{f}(s_1,\dots,s_n)\gamma & = & \mathtt{f}(s_1\gamma,\dots,s_n\gamma) \end{array}$$

Given: a set of rules $\mathcal{R}$

Define:

- $\ell\gamma \Rightarrow_{\mathcal{R}} r\gamma$ for all $\ell \Rightarrow r \in \mathcal{R}$, all $\gamma$
- $\mathtt{f}(s_1,\dots,s_i,\dots,s_n) \Rightarrow_{\mathcal{R}} \mathtt{f}(s_1,\dots,t_i,\dots,s_n)$ if $s_i \Rightarrow_{\mathcal{R}} t_i$

Example:

$$\begin{array}{ccc} \mathtt{min}(\underbrace{\mathtt{add}(\mathtt{s}(\mathtt{0}),\mathtt{s}(\mathtt{s}(\mathtt{0})))}) & \Rightarrow_{\mathcal{R}} & \mathtt{min}(\mathtt{s}(\mathtt{add}(\mathtt{s}(\mathtt{0}),\mathtt{s}(\mathtt{0})))) \\ \mathtt{add}(\;x\;,\mathtt{s}(\;y\;)) & \Rightarrow & \mathtt{s}(\mathtt{add}(\;x\;,\;y\;)) \end{array}$$

# Class exercise

Let

$$\mathcal{R} = \left\{ \begin{array}{rcl} \mathtt{add}(x, \mathtt{s}(y)) & \Rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \\ \mathtt{add}(x, \mathtt{p}(y)) & \Rightarrow & \mathtt{p}(\mathtt{add}(x, y)) \\ \mathtt{add}(x, \mathtt{0}) & \Rightarrow & x \\ \mathtt{s}(\mathtt{p}(x)) & \Rightarrow & x \\ \mathtt{p}(\mathtt{s}(x)) & \Rightarrow & x \end{array} \right\}$$

# Class exercise

Let

$$\mathcal{R} = \left\{ \begin{array}{rcl} \mathtt{add}(x, \mathtt{s}(y)) & \Rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \\ \mathtt{add}(x, \mathtt{p}(y)) & \Rightarrow & \mathtt{p}(\mathtt{add}(x, y)) \\ \mathtt{add}(x, \mathtt{0}) & \Rightarrow & x \\ \mathtt{s}(\mathtt{p}(x)) & \Rightarrow & x \\ \mathtt{p}(\mathtt{s}(x)) & \Rightarrow & x \end{array} \right\}$$

Question: what can we reduce the following term to?

$$\mathtt{s}(\mathtt{add}(\mathtt{0}, \mathtt{p}(\mathtt{s}(\mathtt{0}))))$$

# Class exercise

Let

$$\mathcal{R} = \left\{ \begin{array}{rcl} \mathtt{add}(x, \mathtt{s}(y)) & \Rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \\ \mathtt{add}(x, \mathtt{p}(y)) & \Rightarrow & \mathtt{p}(\mathtt{add}(x, y)) \\ \mathtt{add}(x, \mathtt{0}) & \Rightarrow & x \\ \mathtt{s}(\mathtt{p}(x)) & \Rightarrow & x \\ \mathtt{p}(\mathtt{s}(x)) & \Rightarrow & x \end{array} \right\}$$

Question: what can we reduce the following term to?

$$\mathtt{s}(\mathtt{add}(\mathtt{0}, \mathtt{p}(\mathtt{s}(\mathtt{0}))))$$

Answer: two options!

- $\mathtt{s}(\mathtt{add}(\mathtt{0}, \mathtt{0}))$
- $\mathtt{s}(\mathtt{p}(\mathtt{add}(\mathtt{0}, \mathtt{s}(\mathtt{0}))))$

# Normal form

We are often interested in a reduction to **normal form**.

# Normal form

We are often interested in a reduction to **normal form.**

$$\mathcal{R} = \left\{ \begin{array}{rcl} \mathtt{add}(x, \mathtt{s}(y)) & \Rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \\ \mathtt{add}(x, \mathtt{p}(y)) & \Rightarrow & \mathtt{p}(\mathtt{add}(x, y)) \\ \mathtt{add}(x, \mathtt{0}) & \Rightarrow & x \\ \mathtt{s}(\mathtt{p}(x)) & \Rightarrow & x \\ \mathtt{p}(\mathtt{s}(x)) & \Rightarrow & x \end{array} \right\}$$

# Normal form

We are often interested in a reduction to **normal form.**

$$\mathcal{R} = \left\{ \begin{array}{rcl} \mathtt{add}(x, \mathtt{s}(y)) & \Rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \\ \mathtt{add}(x, \mathtt{p}(y)) & \Rightarrow & \mathtt{p}(\mathtt{add}(x, y)) \\ \mathtt{add}(x, \mathtt{0}) & \Rightarrow & x \\ \mathtt{s}(\mathtt{p}(x)) & \Rightarrow & x \\ \mathtt{p}(\mathtt{s}(x)) & \Rightarrow & x \end{array} \right\}$$

Example:

$$\mathtt{s}(\mathtt{add}(\mathtt{0}, \underline{\mathtt{p}(\mathtt{s}(\mathtt{0}))})) \quad \Rightarrow_{\mathcal{R}} \quad \mathtt{s}(\mathtt{add}(\mathtt{0}, \underline{\mathtt{0}}))$$

# Normal form

We are often interested in a reduction to **normal form.**

$$\mathcal{R} = \left\{ \begin{array}{rcl} \mathtt{add}(x, \mathtt{s}(y)) & \Rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \\ \mathtt{add}(x, \mathtt{p}(y)) & \Rightarrow & \mathtt{p}(\mathtt{add}(x, y)) \\ \mathtt{add}(x, \mathtt{0}) & \Rightarrow & x \\ \mathtt{s}(\mathtt{p}(x)) & \Rightarrow & x \\ \mathtt{p}(\mathtt{s}(x)) & \Rightarrow & x \end{array} \right\}$$

Example:

$$\mathtt{s}(\mathtt{add}(\mathtt{0}, \mathtt{p}(\mathtt{s}(\mathtt{0})))) \quad \Rightarrow_{\mathcal{R}} \quad \underline{\mathtt{s}(\mathtt{add}(\mathtt{0}, \mathtt{0}))}$$

# Normal form

We are often interested in a reduction to **normal form.**

$$\mathcal{R} = \left\{ \begin{array}{rcl} \mathtt{add}(x, \mathtt{s}(y)) & \Rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \\ \mathtt{add}(x, \mathtt{p}(y)) & \Rightarrow & \mathtt{p}(\mathtt{add}(x, y)) \\ \mathtt{add}(x, \mathtt{0}) & \Rightarrow & x \\ \mathtt{s}(\mathtt{p}(x)) & \Rightarrow & x \\ \mathtt{p}(\mathtt{s}(x)) & \Rightarrow & x \end{array} \right\}$$

Example:

$$\begin{array}{rcl} \mathtt{s}(\mathtt{add}(\mathtt{0}, \mathtt{p}(\mathtt{s}(\mathtt{0})))) & \Rightarrow_{\mathcal{R}} & \underline{\mathtt{s}(\mathtt{add}(\mathtt{0}, \mathtt{0}))} \\ & \Rightarrow_{\mathcal{R}} & \underline{\mathtt{s}(\mathtt{0})} \end{array}$$

# Normal form

We are often interested in a reduction to **normal form.**

$$\mathcal{R} = \left\{ \begin{array}{rcl} \mathtt{add}(x, \mathtt{s}(y)) & \Rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \\ \mathtt{add}(x, \mathtt{p}(y)) & \Rightarrow & \mathtt{p}(\mathtt{add}(x, y)) \\ \mathtt{add}(x, \mathtt{0}) & \Rightarrow & x \\ \mathtt{s}(\mathtt{p}(x)) & \Rightarrow & x \\ \mathtt{p}(\mathtt{s}(x)) & \Rightarrow & x \end{array} \right\}$$

Example:

$$\begin{array}{rcl} \mathtt{s}(\mathtt{add}(\mathtt{0}, \mathtt{p}(\mathtt{s}(\mathtt{0})))) & \Rightarrow_{\mathcal{R}} & \mathtt{s}(\mathtt{add}(\mathtt{0}, \mathtt{0})) \\ & \Rightarrow_{\mathcal{R}} & \mathtt{s}(\mathtt{0}) \end{array}$$

Exercise: find a different reduction to normal form for $\mathtt{s}(\mathtt{add}(\mathtt{0}, \mathtt{p}(\mathtt{s}(\mathtt{0}))))$

# Peek-ahead: equational logic

Rules can be seen as **oriented equations**.

# Peek-ahead: equational logic

Rules can be seen as **oriented equations.**

The rules

$$\begin{array}{rcl} \mathtt{add}(\mathtt{0}, y) & \Rightarrow & y \\ \mathtt{add}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \end{array}$$

define the equations:

$$\begin{array}{lrcl} \forall y. & \mathtt{add}(\mathtt{0}, y) & = & y \\ \forall x \forall y. & \mathtt{add}(\mathtt{s}(x), y) & = & \mathtt{s}(\mathtt{add}(x, y)) \end{array}$$

# Peek-ahead: equational logic

Rules can be seen as **oriented equations.**

The rules

$$\begin{array}{rcl} \mathtt{add}(\mathtt{0}, y) & \Rightarrow & y \\ \mathtt{add}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \end{array}$$

define the equations:

$$\begin{array}{rrcl} \forall y. & \mathtt{add}(\mathtt{0}, y) & = & y \\ \forall x \forall y. & \mathtt{add}(\mathtt{s}(x), y) & = & \mathtt{s}(\mathtt{add}(x, y)) \end{array}$$

For any model $(M, [\![\cdot]\!]_\alpha)$ that makes the given equalities true:

If $s \Rightarrow_{\mathcal{R}} t$ then $[\![s]\!]_\alpha = [\![t]\!]_\alpha$ for all $\alpha$.

# Peek-ahead: equational logic

Rules can be seen as **oriented equations.**

The rules

$$\begin{array}{rcl} \mathtt{add}(\mathtt{0}, y) & \Rightarrow & y \\ \mathtt{add}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \end{array}$$

define the equations:

$$\begin{array}{rrcl} \forall y. & \mathtt{add}(\mathtt{0}, y) & = & y \\ \forall x \forall y. & \mathtt{add}(\mathtt{s}(x), y) & = & \mathtt{s}(\mathtt{add}(x, y)) \end{array}$$

For any model $(M, [\![\cdot]\!]_\alpha)$ that makes the given equalities true:

If $s \Rightarrow_{\mathcal{R}} t$ then $[\![s]\!]_\alpha = [\![t]\!]_\alpha$ for all $\alpha$.

Therefore, the computation
$\mathtt{add}(\mathtt{s}(\mathtt{s}(\mathtt{0})), \mathtt{s}(\mathtt{s}(\mathtt{0}))) \Rightarrow^*_{\mathcal{R}} \mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{0}))))$ **proves** that $2 + 2 = 4$!

# Term rewriting and Prover9

We can also reason about reduction in Prover9:

```
formulas(assumptions).
R(a(0,x),x).
R(a(s(x),y),s(a(x,y))).
R(x,y) -> R(a(x,z),a(y,z)).
R(x,y) -> R(a(z,x),a(z,y)).
R(x,y) -> R(s(x),s(y)).
RR(x,x).
(RR(x,y) & R(y,z)) -> RR(x,z).
end_of_list.
formulas(goals).
RR(a(s(s(0)),s(s(0))),s(s(s(s(0))))).
end_of_list.
```

# Functional programming

Computation: reduction to normal form

# Functional programming

Computation: reduction to normal form

```
let rec conc xs ys =
    match xs with
        | [] ⇒ ys
        | h :: t ⇒ h :: (conc t ys)
;;
```

```
let rec rev xs =
    match xs with
        | [] ⇒ []
        | h :: t ⇒ conc (rev t) [h]
;;
```

# Functional programming

Computation: reduction to normal form

```
rev nil        = nil
rev (a:x)      = conc (rev x (a:nil))
conc nil x     = x
conc (a:x) y   = a:(conc x y)
```

# Functional programming

Computation: reduction to normal form

$$
\begin{array}{lcl}
\texttt{rev nil} & = & \texttt{nil} \\
\texttt{rev } (a\texttt{:}x) & = & \texttt{conc (rev } x \texttt{ (}a\texttt{:nil))} \\
\texttt{conc nil } x & = & x \\
\texttt{conc } (a\texttt{:}x) \ y & = & a\texttt{:(conc } x \ y\texttt{)}
\end{array}
$$

Hence, this corresponds to the following term rewriting system:

$$
\begin{array}{rcl}
\texttt{rev}(\texttt{nil}) & \Rightarrow & \texttt{nil} \\
\texttt{rev}(a : \texttt{nil}) & \Rightarrow & \texttt{conc}(\texttt{rev}(x, a : \texttt{nil})) \\
\texttt{conc}(\texttt{nil}, x) & \Rightarrow & x \\
\texttt{conc}(a : x, y) & \Rightarrow & a : \texttt{conc}(x, y)
\end{array}
$$

# Functional programming

$$
\begin{array}{rcl}
\mathtt{rev}(\mathtt{nil}) & \Rightarrow & \mathtt{nil} \\
\mathtt{rev}(a : \mathtt{nil}) & \Rightarrow & \mathtt{conc}(\mathtt{rev}(x, a : \mathtt{nil})) \\
\mathtt{conc}(\mathtt{nil}, x) & \Rightarrow & x \\
\mathtt{conc}(a : x, y) & \Rightarrow & a : \mathtt{conc}(x, y)
\end{array}
$$

Then we have a reduction to normal form:

$$
\begin{array}{ll}
\mathtt{rev}(x : y : \mathtt{nil}) & \Rightarrow \\
\mathtt{conc}(\mathtt{rev}(y : \mathtt{nil}), x : \mathtt{nil}) & \Rightarrow \\
\mathtt{conc}(\mathtt{conc}(\mathtt{rev}(\mathtt{nil}), y : \mathtt{nil}), x : \mathtt{nil}) & \Rightarrow \\
\mathtt{conc}(\mathtt{conc}(\mathtt{nil}, y : \mathtt{nil}), x : \mathtt{nil}) & \Rightarrow \\
\mathtt{conc}(y : \mathtt{nil}, x : \mathtt{nil}) & \Rightarrow \\
y : \mathtt{conc}(\mathtt{nil}, x : \mathtt{nil}) & \Rightarrow \\
y : x : \mathtt{nil} &
\end{array}
$$

# Some nice properties

- $\mathcal{R}$ is **weakly normalising** (WN):
  *every term has a normal form*

# Some nice properties

- $\mathcal{R}$ is **weakly normalising** (WN):
  *every term has a normal form*

- $\mathcal{R}$ is **terminating** ($=$ strongly normalising, SN):
  *no infinite sequence of terms* $t_1, t_2, t_3, \ldots$ *exists such that* $t_i \Rightarrow_{\mathcal{R}} t_{i+1}$ *for all* $i$

# Some nice properties

- $\mathcal{R}$ is **weakly normalising** (WN):
  *every term has a normal form*

- $\mathcal{R}$ is **terminating** (= strongly normalising, SN):
  *no infinite sequence of terms $t_1, t_2, t_3, \ldots$ exists such that $t_i \Rightarrow_{\mathcal{R}} t_{i+1}$ for all $i$*

- $\mathcal{R}$ is **confluent** (= Church-Rosser, CR):
  *if $s \Rightarrow^*_{\mathcal{R}} t$ and $s \Rightarrow^*_{\mathcal{R}} q$ then a term $u$ exists satisfying $t \Rightarrow^*_{\mathcal{R}} u$ and $q \Rightarrow^*_{\mathcal{R}} u$*

# Some nice properties

- $\mathcal{R}$ is **weakly normalising** (WN):
  *every term has a normal form*

- $\mathcal{R}$ is **terminating** (= strongly normalising, SN):
  *no infinite sequence of terms* $t_1, t_2, t_3, \ldots$ *exists such that* $t_i \Rightarrow_{\mathcal{R}} t_{i+1}$ *for all* $i$

- $\mathcal{R}$ is **confluent** (= Church-Rosser, CR):
  *if* $s \Rightarrow^*_{\mathcal{R}} t$ *and* $s \Rightarrow^*_{\mathcal{R}} q$ *then a term* $u$ *exists satisfying* $t \Rightarrow^*_{\mathcal{R}} u$ *and* $q \Rightarrow^*_{\mathcal{R}} u$

- $\mathcal{R}$ is **locally confluent** (= weak Church-Rosser, WCR):
  *if* $s \Rightarrow_{\mathcal{R}} t$ *and* $s \Rightarrow_{\mathcal{R}} q$ *then a term* $u$ *exists satisfying* $t \Rightarrow^*_{\mathcal{R}} u$ *and* $q \Rightarrow^*_{\mathcal{R}} u$

# Strong normalisation implies weak normalisation

# Strong normalisation implies weak normalisation

**Theorem**

If a TRS is terminating, then every term has at least one normal form.

# Strong normalisation implies weak normalisation

**Theorem**

If a TRS is terminating, then every term has at least one normal form.

**Proof:** rewriting as long as possible does not go on forever due to termination.
So it ends in a normal form. □

# Strong normalisation implies weak normalisation

**Theorem**

If a TRS is terminating, then every term has at least one normal form.

**Proof:** rewriting as long as possible does not go on forever due to termination.
So it ends in a normal form. □

The converse is not true: consider the TRS with rules:

$$\begin{array}{lcl} \mathtt{a} & \Rightarrow & \mathtt{a} \\ \mathtt{a} & \Rightarrow & \mathtt{b} \end{array}$$

# Confluence implies uniqueness of normal forms

# Confluence implies uniqueness of normal forms

**Theorem**

If a TRS is confluent, then every term has at most one normal form.

# Confluence implies uniqueness of normal forms

**Theorem**

If a TRS is confluent, then every term has at most one normal form.

**Proof:** Assume $t$ has two normal forms $u, u'$.
Then by confluence there is a $v$ such that $u \Rightarrow^*_{\mathcal{R}} v$ and $u' \Rightarrow^*_{\mathcal{R}} v$.
Since $u, u'$ are normal forms we have $u = v = u'$. $\square$

# Termination + confluence implies existence of unique normal forms

We conclude:

# Termination + confluence implies existence of unique normal forms

We conclude:

**Theorem**

If a TRS is terminating and confluent, then every term has exactly one normal form.

# Termination + confluence implies existence of unique normal forms

We conclude:

**Theorem**

If a TRS is terminating and confluent, then every term has exactly one normal form.

This normal form can be found just by rewriting the term until no further rewriting step is possible.

# Termination + confluence implies existence of unique normal forms

We conclude:

**Theorem**

If a TRS is terminating and confluent, then every term has exactly one normal form.

This normal form can be found just by rewriting the term until no further rewriting step is possible.

This is a very useful combination!

# Decidability of termination

Termination of term rewriting is **undecidable**.

# Decidability of termination

Termination of term rewriting is **undecidable**.

Proof: reduction to the halting problem (in the handout).

# Decidability of termination

Termination of term rewriting is **undecidable**.

Proof: reduction to the halting problem (in the handout).

Nevertheless: often doable!

# Well-founded ordering

Idea: find a **well-founded ordering** $\succ$ and prove that $s \succ t$ whenever $s \Rightarrow_{\mathcal{R}} t$.

# Well-founded ordering

Idea: find a **well-founded ordering** $\succ$ and prove that $s \succ t$ whenever $s \Rightarrow_{\mathcal{R}} t$.

There are many ways to generate such an ordering!

# A finite definition for infinitely many terms?

Difficulty: how to prove $s \succ t$ whenever $s \Rightarrow_{\mathcal{R}} t$?

# A finite definition for infinitely many terms?

Difficulty: how to prove $s \succ t$ whenever $s \Rightarrow_{\mathcal{R}} t$?

$$\begin{array}{rcl} \texttt{add}(\texttt{0}, y) & \Rightarrow & y \\ \texttt{add}(\texttt{s}(x), y) & \Rightarrow & \texttt{s}(\texttt{add}(x, y)) \end{array}$$

Needed: $\texttt{add}(\texttt{0}, \texttt{0}) \succ \texttt{0}$, $\texttt{add}(\texttt{0}, \texttt{add}(x, y)) \succ \texttt{add}(x, y), \ldots$

# A finite definition for infinitely many terms?

Difficulty: how to prove $s \succ t$ whenever $s \Rightarrow_{\mathcal{R}} t$?

$$\begin{array}{rcl} \mathtt{add}(\mathtt{0}, y) & \Rightarrow & y \\ \mathtt{add}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \end{array}$$

Needed: $\mathtt{add}(\mathtt{0}, \mathtt{0}) \succ \mathtt{0}$, $\mathtt{add}(\mathtt{0}, \mathtt{add}(x, y)) \succ \mathtt{add}(x, y), \ldots$

Solution: it suffices to orient the **rules** provided:

# A finite definition for infinitely many terms?

Difficulty: how to prove $s \succ t$ whenever $s \Rightarrow_{\mathcal{R}} t$?

$$\begin{array}{rcl} \mathtt{add}(\mathtt{0}, y) & \Rightarrow & y \\ \mathtt{add}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \end{array}$$

Needed: $\mathtt{add}(\mathtt{0}, \mathtt{0}) \succ \mathtt{0}$, $\mathtt{add}(\mathtt{0}, \mathtt{add}(x, y)) \succ \mathtt{add}(x, y), \dots$

Solution: it suffices to orient the **rules** provided:

- if $s \succ t$ then $s\sigma \succ t\sigma$ for all substitutions $\sigma$ (we say: $\succ$ is **stable**)

# A finite definition for infinitely many terms?

Difficulty: how to prove $s \succ t$ whenever $s \Rightarrow_{\mathcal{R}} t$?

$$\begin{array}{rcl} \mathtt{add}(\mathtt{0}, y) & \Rightarrow & y \\ \mathtt{add}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \end{array}$$

Needed: $\mathtt{add}(\mathtt{0}, \mathtt{0}) \succ \mathtt{0}$, $\mathtt{add}(\mathtt{0}, \mathtt{add}(x, y)) \succ \mathtt{add}(x, y), \dots$

Solution: it suffices to orient the **rules** provided:

- if $s \succ t$ then $s\sigma \succ t\sigma$ for all substitutions $\sigma$
  (we say: $\succ$ is **stable**)
- if $s \succ t$ then $\mathtt{f}(\dots, s, \dots) \succ \mathtt{f}(\dots, t, \dots)$ for all $\mathtt{f}$
  (we say: $\succ$ is **monotonic**)

# A finite definition for infinitely many terms?

Difficulty: how to prove $s \succ t$ whenever $s \Rightarrow_{\mathcal{R}} t$?

$$\begin{array}{rcl} \mathtt{add}(\mathtt{0}, y) & \Rightarrow & y \\ \mathtt{add}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \end{array}$$

Needed: $\mathtt{add}(\mathtt{0}, \mathtt{0}) \succ \mathtt{0}$, $\mathtt{add}(\mathtt{0}, \mathtt{add}(x, y)) \succ \mathtt{add}(x, y), \ldots$

Solution: it suffices to orient the **rules** provided:

- if $s \succ t$ then $s\sigma \succ t\sigma$ for all substitutions $\sigma$ (we say: $\succ$ is **stable**)
- if $s \succ t$ then $\mathtt{f}(\ldots, s, \ldots) \succ \mathtt{f}(\ldots, t, \ldots)$ for all $\mathtt{f}$ (we say: $\succ$ is **monotonic**)

Such an ordering is called a **reduction ordering**.

# LPO

# LPO

Let $\triangleright$ be a **total, well-founded ordering** on the function symbols.

# LPO

Let $\rhd$ be a **total, well-founded ordering** on the function symbols.

We define: $\mathtt{f}(s_1, \ldots, s_n) \succ_{\text{LPO}} t$ if one of the following holds:

# LPO

Let $\rhd$ be a **total, well-founded ordering** on the function symbols.

We define: $\mathtt{f}(s_1, \ldots, s_n) \succ_{\text{LPO}} t$ if one of the following holds:

(sub) $s_i \succeq_{\text{LPO}} t$ for some $i \in \{1, \ldots, n\}$

# LPO

Let $\rhd$ be a **total, well-founded ordering** on the function symbols.

We define: $\mathtt{f}(s_1, \ldots, s_n) \succ_{\mathrm{LPO}} t$ if one of the following holds:

(sub) $s_i \succeq_{\mathrm{LPO}} t$ for some $i \in \{1, \ldots, n\}$

(copy) $t = \mathtt{g}(t_1, \ldots, t_m)$ and $\mathtt{f} \rhd \mathtt{g}$ and $\mathtt{f}(s_1, \ldots, s_n) \succ_{\mathrm{LPO}} t_j$ for all $j \in \{1, \ldots, m\}$

# LPO

Let $\rhd$ be a **total, well-founded ordering** on the function symbols.

We define: $\mathtt{f}(s_1, \ldots, s_n) \succ_{\mathrm{LPO}} t$ if one of the following holds:

(sub) $s_i \succeq_{\mathrm{LPO}} t$ for some $i \in \{1, \ldots, n\}$

(copy) $t = \mathtt{g}(t_1, \ldots, t_m)$ and $\mathtt{f} \rhd \mathtt{g}$ and $\mathtt{f}(s_1, \ldots, s_n) \succ_{\mathrm{LPO}} t_j$ for all $j \in \{1, \ldots, m\}$

(lex) $t = \mathtt{f}(t_1, \ldots, t_n)$ and $\mathtt{f}(s_1, \ldots, s_n) \succ_{\mathrm{LPO}} t_i$ for all $i \in \{1, \ldots, n\}$, and $[s_1, \ldots, s_n](\succ_{\mathrm{LPO}})_{\mathrm{lex}}[t_1, \ldots, t_n]$;

# LPO example

$$
\begin{aligned}
\mathtt{add}(\mathtt{0}, y) &\Rightarrow y \\
\mathtt{add}(\mathtt{s}(x), y) &\Rightarrow \mathtt{s}(\mathtt{add}(x, y)) \\
\mathtt{mul}(\mathtt{0}, y) &\Rightarrow \mathtt{0} \\
\mathtt{mul}(\mathtt{s}(x), y) &\Rightarrow \mathtt{add}(y, \mathtt{mul}(x, y))
\end{aligned}
$$

# LPO example

$$\begin{array}{rcl} \mathtt{add}(\mathtt{0}, y) & \Rightarrow & y \\ \mathtt{add}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \\ \mathtt{mul}(\mathtt{0}, y) & \Rightarrow & \mathtt{0} \\ \mathtt{mul}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{add}(y, \mathtt{mul}(x, y)) \end{array}$$

We choose: $\mathtt{mul} \rhd \mathtt{add} \rhd \mathtt{s} \rhd \mathtt{0}$

# LPO example

$$
\begin{array}{rcl}
\mathtt{add}(\mathtt{0}, y) & \Rightarrow & y \\
\mathtt{add}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \\
\mathtt{mul}(\mathtt{0}, y) & \Rightarrow & \mathtt{0} \\
\mathtt{mul}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{add}(y, \mathtt{mul}(x, y))
\end{array}
$$

We choose: $\mathtt{mul} \rhd \mathtt{add} \rhd \mathtt{s} \rhd \mathtt{0}$

(sub) $s_i \succeq_{\mathrm{LPO}} t$ for some $i \in \{1, \ldots, n\}$

(copy) $t = \mathtt{g}(t_1, \ldots, t_m)$; $\mathtt{f} \rhd \mathtt{g}$; $\mathtt{f}(s_1, \ldots, s_n) \succ_{\mathrm{LPO}}$ each $t_j$

(lex) $t = \mathtt{f}(t_1, \ldots, t_n)$; $s \succ_{\mathrm{LPO}}$ each $t_i$; $\vec{s}(\succ_{\mathrm{LPO}})_{\mathrm{lex}}\vec{t}$;

# LPO example

$$\begin{array}{rcl} \mathtt{add}(\mathtt{0}, y) & \Rightarrow & y \\ \mathtt{add}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \\ \mathtt{mul}(\mathtt{0}, y) & \Rightarrow & \mathtt{0} \\ \mathtt{mul}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{add}(y, \mathtt{mul}(x, y)) \end{array}$$

We choose: $\mathtt{mul} \rhd \mathtt{add} \rhd \mathtt{s} \rhd \mathtt{0}$

(sub) $s_i \succeq_{\mathrm{LPO}} t$ for some $i \in \{1, \ldots, n\}$

(copy) $t = \mathtt{g}(t_1, \ldots, t_m)$; $\mathtt{f} \rhd \mathtt{g}$; $\mathtt{f}(s_1, \ldots, s_n) \succ_{\mathrm{LPO}}$ each $t_j$

(lex) $t = \mathtt{f}(t_1, \ldots, t_n)$; $s \succ_{\mathrm{LPO}}$ each $t_i$; $\vec{s}(\succ_{\mathrm{LPO}})_{\mathrm{lex}}\vec{t}$;

We orient the last rule:

$$\mathtt{mul}(\mathtt{s}(x), y) \succ_{\mathrm{LPO}} \mathtt{add}(y, \mathtt{mul}(x, y))$$

# LPO example

$$\begin{array}{rcl} \mathtt{add}(\mathtt{0}, y) & \Rightarrow & y \\ \mathtt{add}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \\ \mathtt{mul}(\mathtt{0}, y) & \Rightarrow & \mathtt{0} \\ \mathtt{mul}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{add}(y, \mathtt{mul}(x, y)) \end{array}$$

We choose: $\mathtt{mul} \rhd \mathtt{add} \rhd \mathtt{s} \rhd \mathtt{0}$

(sub) $s_i \succeq_{\mathrm{LPO}} t$ for some $i \in \{1, \ldots, n\}$

(copy) $t = \mathtt{g}(t_1, \ldots, t_m)$; $\mathtt{f} \rhd \mathtt{g}$; $\mathtt{f}(s_1, \ldots, s_n) \succ_{\mathrm{LPO}}$ each $t_j$

(lex) $t = \mathtt{f}(t_1, \ldots, t_n)$; $s \succ_{\mathrm{LPO}}$ each $t_i$; $\vec{s}(\succ_{\mathrm{LPO}})_{\mathrm{lex}}\vec{t}$;

We orient the last rule:

$$\mathtt{mul}(\mathtt{s}(x), y) \succ_{\mathrm{LPO}} y$$

$$\mathtt{mul}(\mathtt{s}(x), y) \succ_{\mathrm{LPO}} \mathtt{mul}(x, y)$$

# LPO example

$$\begin{array}{rcl} \mathtt{add}(\mathtt{0}, y) & \Rightarrow & y \\ \mathtt{add}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \\ \mathtt{mul}(\mathtt{0}, y) & \Rightarrow & \mathtt{0} \\ \mathtt{mul}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{add}(y, \mathtt{mul}(x, y)) \end{array}$$

We choose: $\mathtt{mul} \rhd \mathtt{add} \rhd \mathtt{s} \rhd \mathtt{0}$

(sub) $s_i \succeq_{\mathrm{LPO}} t$ for some $i \in \{1, \ldots, n\}$

(copy) $t = \mathtt{g}(t_1, \ldots, t_m)$; $\mathtt{f} \rhd \mathtt{g}$; $\mathtt{f}(s_1, \ldots, s_n) \succ_{\mathrm{LPO}}$ each $t_j$

(lex) $t = \mathtt{f}(t_1, \ldots, t_n)$; $s \succ_{\mathrm{LPO}}$ each $t_i$; $\vec{s}(\succ_{\mathrm{LPO}})_{\mathrm{lex}}\vec{t}$;

We orient the last rule:

$$y \succeq_{\mathrm{LPO}} y$$

$$\mathtt{mul}(\mathtt{s}(x), y) \succ_{\mathrm{LPO}} \mathtt{mul}(x, y)$$

# LPO example

$$\begin{array}{rcl} \mathtt{add}(\mathtt{0}, y) & \Rightarrow & y \\ \mathtt{add}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \\ \mathtt{mul}(\mathtt{0}, y) & \Rightarrow & \mathtt{0} \\ \mathtt{mul}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{add}(y, \mathtt{mul}(x, y)) \end{array}$$

We choose: $\mathtt{mul} \rhd \mathtt{add} \rhd \mathtt{s} \rhd \mathtt{0}$

(sub) $s_i \succeq_{\mathrm{LPO}} t$ for some $i \in \{1, \ldots, n\}$

(copy) $t = \mathtt{g}(t_1, \ldots, t_m)$; $\mathtt{f} \rhd \mathtt{g}$; $\mathtt{f}(s_1, \ldots, s_n) \succ_{\mathrm{LPO}}$ each $t_j$

(lex) $t = \mathtt{f}(t_1, \ldots, t_n)$; $s \succ_{\mathrm{LPO}}$ each $t_i$; $\vec{s}(\succ_{\mathrm{LPO}})_{\mathrm{lex}}\vec{t}$;

We orient the last rule:

$$\mathtt{mul}(\mathtt{s}(x), y) \succ_{\mathrm{LPO}} \mathtt{mul}(x, y)$$

# LPO example

$$\begin{aligned}
\mathtt{add}(\mathtt{0}, y) &\Rightarrow y \\
\mathtt{add}(\mathtt{s}(x), y) &\Rightarrow \mathtt{s}(\mathtt{add}(x, y)) \\
\mathtt{mul}(\mathtt{0}, y) &\Rightarrow \mathtt{0} \\
\mathtt{mul}(\mathtt{s}(x), y) &\Rightarrow \mathtt{add}(y, \mathtt{mul}(x, y))
\end{aligned}$$

We choose: $\mathtt{mul} \rhd \mathtt{add} \rhd \mathtt{s} \rhd \mathtt{0}$

(sub) $s_i \succeq_{\mathrm{LPO}} t$ for some $i \in \{1, \ldots, n\}$
(copy) $t = \mathtt{g}(t_1, \ldots, t_m)$; $\mathtt{f} \rhd \mathtt{g}$; $\mathtt{f}(s_1, \ldots, s_n) \succ_{\mathrm{LPO}}$ each $t_j$
(lex) $t = \mathtt{f}(t_1, \ldots, t_n)$; $s \succ_{\mathrm{LPO}}$ each $t_i$; $\vec{s}(\succ_{\mathrm{LPO}})_{\mathrm{lex}}\vec{t}$;

We orient the last rule:

$$\mathtt{mul}(\mathtt{s}(x), y) \succ_{\mathrm{LPO}} x$$

$$\mathtt{mul}(\mathtt{s}(x), y) \succ_{\mathrm{LPO}} y$$

$$\mathtt{s}(x) \succ_{\mathrm{LPO}} x$$

# LPO example

$$
\begin{aligned}
\mathtt{add}(\mathtt{0}, y) &\Rightarrow y \\
\mathtt{add}(\mathtt{s}(x), y) &\Rightarrow \mathtt{s}(\mathtt{add}(x, y)) \\
\mathtt{mul}(\mathtt{0}, y) &\Rightarrow \mathtt{0} \\
\mathtt{mul}(\mathtt{s}(x), y) &\Rightarrow \mathtt{add}(y, \mathtt{mul}(x, y))
\end{aligned}
$$

We choose: $\mathtt{mul} \rhd \mathtt{add} \rhd \mathtt{s} \rhd \mathtt{0}$

(sub) $s_i \succeq_{\text{LPO}} t$ for some $i \in \{1, \ldots, n\}$
(copy) $t = \mathtt{g}(t_1, \ldots, t_m)$; $\mathtt{f} \rhd \mathtt{g}$; $\mathtt{f}(s_1, \ldots, s_n) \succ_{\text{LPO}}$ each $t_j$
(lex) $t = \mathtt{f}(t_1, \ldots, t_n)$; $s \succ_{\text{LPO}}$ each $t_i$; $\vec{s}(\succ_{\text{LPO}})_{\text{lex}}\vec{t}$;

We orient the last rule:

$$\mathtt{s}(x) \succeq_{\text{LPO}} x$$

$$\mathtt{mul}(\mathtt{s}(x), y) \succ_{\text{LPO}} y$$

$$\mathtt{s}(x) \succ_{\text{LPO}} x$$

# LPO example

$$\begin{array}{rcl} \mathtt{add}(\mathtt{0}, y) & \Rightarrow & y \\ \mathtt{add}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \\ \mathtt{mul}(\mathtt{0}, y) & \Rightarrow & \mathtt{0} \\ \mathtt{mul}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{add}(y, \mathtt{mul}(x, y)) \end{array}$$

We choose: $\mathtt{mul} \rhd \mathtt{add} \rhd \mathtt{s} \rhd \mathtt{0}$

(sub) $s_i \succeq_{\text{LPO}} t$ for some $i \in \{1, \ldots, n\}$

(copy) $t = \mathtt{g}(t_1, \ldots, t_m)$; $\mathtt{f} \rhd \mathtt{g}$; $\mathtt{f}(s_1, \ldots, s_n) \succ_{\text{LPO}}$ each $t_j$

(lex) $t = \mathtt{f}(t_1, \ldots, t_n)$; $s \succ_{\text{LPO}}$ each $t_i$; $\vec{s}(\succ_{\text{LPO}})_{\text{lex}}\vec{t}$;

We orient the last rule:

$$\mathtt{mul}(\mathtt{s}(x), y) \succ_{\text{LPO}} y$$

$$\mathtt{s}(x) \succ_{\text{LPO}} x$$

# LPO example

$$
\begin{array}{rcl}
\mathtt{add}(\mathtt{0}, y) & \Rightarrow & y \\
\mathtt{add}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \\
\mathtt{mul}(\mathtt{0}, y) & \Rightarrow & \mathtt{0} \\
\mathtt{mul}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{add}(y, \mathtt{mul}(x, y))
\end{array}
$$

We choose: $\mathtt{mul} \rhd \mathtt{add} \rhd \mathtt{s} \rhd \mathtt{0}$

$$
\begin{array}{rl}
\text{(sub)} & s_i \succeq_{\text{LPO}} t \text{ for some } i \in \{1, \ldots, n\} \\
\text{(copy)} & t = \mathtt{g}(t_1, \ldots, t_m);\ \mathtt{f} \rhd \mathtt{g};\ \mathtt{f}(s_1, \ldots, s_n) \succ_{\text{LPO}} \text{ each } t_j \\
\text{(lex)} & t = \mathtt{f}(t_1, \ldots, t_n);\ s \succ_{\text{LPO}} \text{ each } t_i;\ \vec{s}(\succ_{\text{LPO}})_{\text{lex}}\vec{t};
\end{array}
$$

We orient the last rule:

$$y \succeq_{\text{LPO}} y$$

$$\mathtt{s}(x) \succ_{\text{LPO}} x$$

# LPO example

$$\begin{array}{rcl} \mathtt{add}(\mathtt{0},y) & \Rightarrow & y \\ \mathtt{add}(\mathtt{s}(x),y) & \Rightarrow & \mathtt{s}(\mathtt{add}(x,y)) \\ \mathtt{mul}(\mathtt{0},y) & \Rightarrow & \mathtt{0} \\ \mathtt{mul}(\mathtt{s}(x),y) & \Rightarrow & \mathtt{add}(y,\mathtt{mul}(x,y)) \end{array}$$

We choose: $\mathtt{mul} \rhd \mathtt{add} \rhd \mathtt{s} \rhd \mathtt{0}$

(sub) $s_i \succeq_{\mathrm{LPO}} t$ for some $i \in \{1, \ldots, n\}$

(copy) $t = \mathtt{g}(t_1, \ldots, t_m)$; $\mathtt{f} \rhd \mathtt{g}$; $\mathtt{f}(s_1, \ldots, s_n) \succ_{\mathrm{LPO}}$ each $t_j$

(lex) $t = \mathtt{f}(t_1, \ldots, t_n)$; $s \succ_{\mathrm{LPO}}$ each $t_i$; $\vec{s}(\succ_{\mathrm{LPO}})_{\mathrm{lex}}\vec{t}$;

We orient the last rule:

$$\mathtt{s}(x) \succ_{\mathrm{LPO}} x$$

# LPO example

$$\begin{array}{rcl} \mathtt{add}(\mathtt{0}, y) & \Rightarrow & y \\ \mathtt{add}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \\ \mathtt{mul}(\mathtt{0}, y) & \Rightarrow & \mathtt{0} \\ \mathtt{mul}(\mathtt{s}(x), y) & \Rightarrow & \mathtt{add}(y, \mathtt{mul}(x, y)) \end{array}$$

We choose: $\mathtt{mul} \rhd \mathtt{add} \rhd \mathtt{s} \rhd \mathtt{0}$

(sub) $s_i \succeq_{\mathrm{LPO}} t$ for some $i \in \{1, \dots, n\}$

(copy) $t = \mathtt{g}(t_1, \dots, t_m)$; $\mathtt{f} \rhd \mathtt{g}$; $\mathtt{f}(s_1, \dots, s_n) \succ_{\mathrm{LPO}}$ each $t_j$

(lex) $t = \mathtt{f}(t_1, \dots, t_n)$; $s \succ_{\mathrm{LPO}}$ each $t_i$; $\vec{s}(\succ_{\mathrm{LPO}})_{\mathrm{lex}}\vec{t}$;

We orient the last rule:

$$x \succeq_{\mathrm{LPO}} x$$

# Soundness of LPO

**Theorem**

If $\ell \succ_{\mathrm{LPO}} r$ for all rules in $\mathcal{R}$, then the TRS with rules $\mathcal{R}$ is terminating.

# Soundness of LPO

**Theorem**

If $\ell \succ_{\text{LPO}} r$ for all rules in $\mathcal{R}$, then the TRS with rules $\mathcal{R}$ is terminating.

**Proof.** $\succ_{\text{LPO}}$ is:

# Soundness of LPO

**Theorem**

If $\ell \succ_{\mathrm{LPO}} r$ for all rules in $\mathcal{R}$, then the TRS with rules $\mathcal{R}$ is terminating.

**Proof.** $\succ_{\mathrm{LPO}}$ is:

- stable: if $s \succ_{\mathrm{LPO}} t$ then $s\sigma \succ_{\mathrm{LPO}} t\sigma$

# Soundness of LPO

**Theorem**

If $\ell \succ_{\mathrm{LPO}} r$ for all rules in $\mathcal{R}$, then the TRS with rules $\mathcal{R}$ is terminating.

**Proof.** $\succ_{\mathrm{LPO}}$ is:

- stable: if $s \succ_{\mathrm{LPO}} t$ then $s\sigma \succ_{\mathrm{LPO}} t\sigma$
- monotonic: if $s \succ_{\mathrm{LPO}} t$ then $\mathtt{f}(\ldots, s, \ldots) \succ_{\mathrm{LPO}} \mathtt{f}(\ldots, t, \ldots)$

# Soundness of LPO

**Theorem**

If $\ell \succ_{\mathrm{LPO}} r$ for all rules in $\mathcal{R}$, then the TRS with rules $\mathcal{R}$ is terminating.

**Proof.** $\succ_{\mathrm{LPO}}$ is:

- stable: if $s \succ_{\mathrm{LPO}} t$ then $s\sigma \succ_{\mathrm{LPO}} t\sigma$
- monotonic: if $s \succ_{\mathrm{LPO}} t$ then $\mathtt{f}(\dots, s, \dots) \succ_{\mathrm{LPO}} \mathtt{f}(\dots, t, \dots)$
- well-founded: there is no infinite decreasing sequence

# Ackermann example

# Ackermann example

Consider the **Ackermann function:**

$$A(m,n) = \begin{cases} n+1 & \text{if } m = 0 \\ A(m-1,1) & \text{if } m > 0 \text{ and } n = 0 \\ A(m-1, A(m, n-1)) & \text{if } m > 0 \text{ and } n > 0 \end{cases}$$

# Ackermann example

Consider the **Ackermann function:**

$$A(m,n) = \begin{cases} n+1 & \text{if } m = 0 \\ A(m-1,1) & \text{if } m > 0 \text{ and } n = 0 \\ A(m-1, A(m, n-1)) & \text{if } m > 0 \text{ and } n > 0 \end{cases}$$

Looks simple, but $A(4,2)$ has 19,729 decimal digits.

# Ackermann example

Consider the **Ackermann function:**

$$A(m,n) = \begin{cases} n+1 & \text{if } m = 0 \\ A(m-1,1) & \text{if } m > 0 \text{ and } n = 0 \\ A(m-1, A(m, n-1)) & \text{if } m > 0 \text{ and } n > 0 \end{cases}$$

Looks simple, but $A(4,2)$ has 19,729 decimal digits.

Expressed as a TRS:

$$\begin{array}{rcl} \mathtt{A}(\mathtt{0}, x) & \Rightarrow & \mathtt{s}(x) \\ \mathtt{A}(\mathtt{s}(x), \mathtt{0}) & \Rightarrow & \mathtt{A}(x, \mathtt{s}(\mathtt{0})) \\ \mathtt{A}(\mathtt{s}(x), \mathtt{s}(y)) & \Rightarrow & \mathtt{A}(x, \mathtt{A}(\mathtt{s}(x), y)) \end{array}$$

# Ackermann example

$$
\begin{array}{rcl}
\mathtt{A}(\mathtt{0}, x) & \Rightarrow & \mathtt{s}(x) \\
\mathtt{A}(\mathtt{s}(x), \mathtt{0}) & \Rightarrow & \mathtt{A}(x, \mathtt{s}(\mathtt{0})) \\
\mathtt{A}(\mathtt{s}(x), \mathtt{s}(y)) & \Rightarrow & \mathtt{A}(x, \mathtt{A}(\mathtt{s}(x), y))
\end{array}
$$

Choose: $\mathtt{A} \rhd \mathtt{s}$.

# Ackermann example

$$\begin{array}{rcl} \mathtt{A}(\mathtt{0}, x) & \Rightarrow & \mathtt{s}(x) \\ \mathtt{A}(\mathtt{s}(x), \mathtt{0}) & \Rightarrow & \mathtt{A}(x, \mathtt{s}(\mathtt{0})) \\ \mathtt{A}(\mathtt{s}(x), \mathtt{s}(y)) & \Rightarrow & \mathtt{A}(x, \mathtt{A}(\mathtt{s}(x), y)) \end{array}$$

Choose: $\mathtt{A} \triangleright \mathtt{s}$.

For example, for the last rule:

$$\mathtt{A}(\mathtt{s}(x), \mathtt{s}(y)) \succ_{\mathrm{LPO}} \mathtt{A}(x, \mathtt{A}(\mathtt{s}(x), y))$$

# Ackermann example

$$\begin{array}{rcl} \mathtt{A}(\mathtt{0}, x) & \Rightarrow & \mathtt{s}(x) \\ \mathtt{A}(\mathtt{s}(x), \mathtt{0}) & \Rightarrow & \mathtt{A}(x, \mathtt{s}(\mathtt{0})) \\ \mathtt{A}(\mathtt{s}(x), \mathtt{s}(y)) & \Rightarrow & \mathtt{A}(x, \mathtt{A}(\mathtt{s}(x), y)) \end{array}$$

Choose: $\mathtt{A} \rhd \mathtt{s}$.

For example, for the last rule:

$$\mathtt{A}(\mathtt{s}(x), \mathtt{s}(y)) \succ_{\text{LPO}} x$$

$$\mathtt{A}(\mathtt{s}(x), \mathtt{s}(y)) \succ_{\text{LPO}} \mathtt{A}(\mathtt{s}(x), y)$$

$$\mathtt{s}(x) \succ_{\text{LPO}} x$$

# Ackermann example

$$
\begin{array}{rcl}
\mathtt{A}(\mathtt{0}, x) & \Rightarrow & \mathtt{s}(x) \\
\mathtt{A}(\mathtt{s}(x), \mathtt{0}) & \Rightarrow & \mathtt{A}(x, \mathtt{s}(\mathtt{0})) \\
\mathtt{A}(\mathtt{s}(x), \mathtt{s}(y)) & \Rightarrow & \mathtt{A}(x, \mathtt{A}(\mathtt{s}(x), y))
\end{array}
$$

Choose: $\mathtt{A} \rhd \mathtt{s}$.

For example, for the last rule:

$$\mathtt{s}(x) \succeq_{\mathrm{LPO}} x$$

$$\mathtt{A}(\mathtt{s}(x), \mathtt{s}(y)) \succ_{\mathrm{LPO}} \mathtt{A}(\mathtt{s}(x), y)$$

$$\mathtt{s}(x) \succ_{\mathrm{LPO}} x$$

# Ackermann example

$$\begin{array}{rcl} \mathtt{A}(\mathtt{0}, x) & \Rightarrow & \mathtt{s}(x) \\ \mathtt{A}(\mathtt{s}(x), \mathtt{0}) & \Rightarrow & \mathtt{A}(x, \mathtt{s}(\mathtt{0})) \\ \mathtt{A}(\mathtt{s}(x), \mathtt{s}(y)) & \Rightarrow & \mathtt{A}(x, \mathtt{A}(\mathtt{s}(x), y)) \end{array}$$

Choose: $\mathtt{A} \rhd \mathtt{s}$.

For example, for the last rule:

$$\mathtt{A}(\mathtt{s}(x), \mathtt{s}(y)) \succ_{\mathrm{LPO}} \mathtt{A}(\mathtt{s}(x), y)$$

$$\mathtt{s}(x) \succ_{\mathrm{LPO}} x$$

# Ackermann example

$$\begin{array}{rcl} \mathtt{A}(\mathtt{0}, x) & \Rightarrow & \mathtt{s}(x) \\ \mathtt{A}(\mathtt{s}(x), \mathtt{0}) & \Rightarrow & \mathtt{A}(x, \mathtt{s}(\mathtt{0})) \\ \mathtt{A}(\mathtt{s}(x), \mathtt{s}(y)) & \Rightarrow & \mathtt{A}(x, \mathtt{A}(\mathtt{s}(x), y)) \end{array}$$

Choose: $\mathtt{A} \rhd \mathtt{s}$.

For example, for the last rule:

$$\mathtt{A}(\mathtt{s}(x), \mathtt{s}(y)) \succ_{\mathrm{LPO}} \mathtt{s}(x)$$

$$\mathtt{A}(\mathtt{s}(x), \mathtt{s}(y)) \succ_{\mathrm{LPO}} y$$

$$\mathtt{s}(y) \succ_{\mathrm{LPO}} y$$

$$\mathtt{s}(x) \succ_{\mathrm{LPO}} x$$

# Ackermann example

$$
\begin{array}{rcl}
\mathtt{A}(\mathtt{0}, x) & \Rightarrow & \mathtt{s}(x) \\
\mathtt{A}(\mathtt{s}(x), \mathtt{0}) & \Rightarrow & \mathtt{A}(x, \mathtt{s}(\mathtt{0})) \\
\mathtt{A}(\mathtt{s}(x), \mathtt{s}(y)) & \Rightarrow & \mathtt{A}(x, \mathtt{A}(\mathtt{s}(x), y))
\end{array}
$$

Choose: $\mathtt{A} \rhd \mathtt{s}$.

For example, for the last rule:

$$\mathtt{s}(x) \succeq_{\mathrm{LPO}} \mathtt{s}(x)$$

$$\mathtt{A}(\mathtt{s}(x), \mathtt{s}(y)) \succ_{\mathrm{LPO}} y$$

$$\mathtt{s}(y) \succ_{\mathrm{LPO}} y$$

$$\mathtt{s}(x) \succ_{\mathrm{LPO}} x$$

# Ackermann example

$$\begin{array}{rcl} \mathtt{A}(\mathtt{0}, x) & \Rightarrow & \mathtt{s}(x) \\ \mathtt{A}(\mathtt{s}(x), \mathtt{0}) & \Rightarrow & \mathtt{A}(x, \mathtt{s}(\mathtt{0})) \\ \mathtt{A}(\mathtt{s}(x), \mathtt{s}(y)) & \Rightarrow & \mathtt{A}(x, \mathtt{A}(\mathtt{s}(x), y)) \end{array}$$

Choose: $\mathtt{A} \rhd \mathtt{s}$.

For example, for the last rule:

$$\mathtt{A}(\mathtt{s}(x), \mathtt{s}(y)) \succ_{\mathrm{LPO}} y$$

$$\mathtt{s}(y) \succ_{\mathrm{LPO}} y$$

$$\mathtt{s}(x) \succ_{\mathrm{LPO}} x$$

# Ackermann example

$$\begin{array}{rcl} \mathtt{A}(\mathtt{0}, x) & \Rightarrow & \mathtt{s}(x) \\ \mathtt{A}(\mathtt{s}(x), \mathtt{0}) & \Rightarrow & \mathtt{A}(x, \mathtt{s}(\mathtt{0})) \\ \mathtt{A}(\mathtt{s}(x), \mathtt{s}(y)) & \Rightarrow & \mathtt{A}(x, \mathtt{A}(\mathtt{s}(x), y)) \end{array}$$

Choose: $\mathtt{A} \rhd \mathtt{s}$.

For example, for the last rule:

$$\mathtt{s}(y) \succeq_{\mathrm{LPO}} y$$

$$\mathtt{s}(y) \succ_{\mathrm{LPO}} y$$

$$\mathtt{s}(x) \succ_{\mathrm{LPO}} x$$

# Ackermann example

$$
\begin{array}{rcl}
\mathtt{A}(\mathtt{0}, x) & \Rightarrow & \mathtt{s}(x) \\
\mathtt{A}(\mathtt{s}(x), \mathtt{0}) & \Rightarrow & \mathtt{A}(x, \mathtt{s}(\mathtt{0})) \\
\mathtt{A}(\mathtt{s}(x), \mathtt{s}(y)) & \Rightarrow & \mathtt{A}(x, \mathtt{A}(\mathtt{s}(x), y))
\end{array}
$$

Choose: $\mathtt{A} \rhd \mathtt{s}$.

For example, for the last rule:

$$\mathtt{s}(y) \succ_{\mathrm{LPO}} y$$

$$\mathtt{s}(x) \succ_{\mathrm{LPO}} x$$

# Ackermann example

$$
\begin{array}{rcl}
\mathtt{A}(\mathtt{0}, x) & \Rightarrow & \mathtt{s}(x) \\
\mathtt{A}(\mathtt{s}(x), \mathtt{0}) & \Rightarrow & \mathtt{A}(x, \mathtt{s}(\mathtt{0})) \\
\mathtt{A}(\mathtt{s}(x), \mathtt{s}(y)) & \Rightarrow & \mathtt{A}(x, \mathtt{A}(\mathtt{s}(x), y))
\end{array}
$$

Choose: $\mathtt{A} \rhd \mathtt{s}$.

For example, for the last rule:

$$y \succeq_{\mathrm{LPO}} y$$

$$\mathtt{s}(x) \succ_{\mathrm{LPO}} x$$

# Ackermann example

$$
\begin{array}{rcl}
\mathtt{A}(\mathtt{0}, x) & \Rightarrow & \mathtt{s}(x) \\
\mathtt{A}(\mathtt{s}(x), \mathtt{0}) & \Rightarrow & \mathtt{A}(x, \mathtt{s}(\mathtt{0})) \\
\mathtt{A}(\mathtt{s}(x), \mathtt{s}(y)) & \Rightarrow & \mathtt{A}(x, \mathtt{A}(\mathtt{s}(x), y))
\end{array}
$$

Choose: $\mathtt{A} \rhd \mathtt{s}$.

For example, for the last rule:

$$\mathtt{s}(x) \succ_{\mathrm{LPO}} x$$

# Ackermann example

$$\begin{array}{rcl} \mathtt{A}(\mathtt{0}, x) & \Rightarrow & \mathtt{s}(x) \\ \mathtt{A}(\mathtt{s}(x), \mathtt{0}) & \Rightarrow & \mathtt{A}(x, \mathtt{s}(\mathtt{0})) \\ \mathtt{A}(\mathtt{s}(x), \mathtt{s}(y)) & \Rightarrow & \mathtt{A}(x, \mathtt{A}(\mathtt{s}(x), y)) \end{array}$$

Choose: $\mathtt{A} \rhd \mathtt{s}$.

For example, for the last rule:

$$x \succeq_{\mathrm{LPO}} x$$

# Ackermann example

$$
\begin{array}{rcl}
\mathtt{A}(\mathtt{0}, x) & \Rightarrow & \mathtt{s}(x) \\
\mathtt{A}(\mathtt{s}(x), \mathtt{0}) & \Rightarrow & \mathtt{A}(x, \mathtt{s}(\mathtt{0})) \\
\mathtt{A}(\mathtt{s}(x), \mathtt{s}(y)) & \Rightarrow & \mathtt{A}(x, \mathtt{A}(\mathtt{s}(x), y))
\end{array}
$$

Choose: $\mathtt{A} \rhd \mathtt{s}$.

We conclude termination.

# An automatic solution?

Problem: how to know if rules are oriented by LPO? (And find $\rhd$?)

# An automatic solution?

Problem: how to know if rules are oriented by LPO? (And find $\rhd$?)

Problem reformulation:

*Do there exist a symbol ordering $\rhd$,*
*and a sequence of proof steps*
*such that the given inequalities $s \succ_{\mathrm{LPO}} t$ all hold?*

# An automatic solution?

Problem: how to know if rules are oriented by LPO? (And find $\rhd$?)

Problem reformulation:

*Do there exist a symbol ordering $\rhd$,*
*and a sequence of proof steps*
*such that the given inequalities $s \succ_{\mathrm{LPO}} t$ all hold?*

This is a SAT problem!

# An automatic solution?

Problem: how to know if rules are oriented by LPO? (And find $\rhd$?)

Problem reformulation:

*Do there exist a symbol ordering* $\rhd$*,*
*and a sequence of proof steps*
*such that the given inequalities* $s \succ_{\text{LPO}} t$ *all hold?*

This is a SAT problem!

Idea:

# An automatic solution?

Problem: how to know if rules are oriented by LPO? (And find $\rhd$?)

Problem reformulation:

*Do there exist a symbol ordering $\rhd$,*
*and a sequence of proof steps*
*such that the given inequalities $s \succ_{\text{LPO}} t$ all hold?*

This is a SAT problem!

Idea:

- encode the ordering using variables $\langle \mathtt{f} \rhd \mathtt{g} \rangle$ for all $\mathtt{f}, \mathtt{g}$

# An automatic solution?

Problem: how to know if rules are oriented by LPO? (And find $\rhd$?)

Problem reformulation:

*Do there exist a symbol ordering $\rhd$,*
*and a sequence of proof steps*
*such that the given inequalities $s \succ_{\text{LPO}} t$ all hold?*

This is a SAT problem!

Idea:

- encode the ordering using variables $\langle \mathtt{f} \rhd \mathtt{g} \rangle$ for all $\mathtt{f}, \mathtt{g}$
- require: $\neg\langle \mathtt{f} \rhd \mathtt{f} \rangle$ for all $\mathtt{f}$

# An automatic solution?

Problem: how to know if rules are oriented by LPO? (And find $\rhd$?)

Problem reformulation:

*Do there exist a symbol ordering $\rhd$,*
*and a sequence of proof steps*
*such that the given inequalities $s \succ_{\mathrm{LPO}} t$ all hold?*

This is a SAT problem!

Idea:

- encode the ordering using variables $\langle \mathtt{f} \rhd \mathtt{g} \rangle$ for all $\mathtt{f}, \mathtt{g}$
- require: $\neg\langle \mathtt{f} \rhd \mathtt{f} \rangle$ for all $\mathtt{f}$
- require: $\langle \mathtt{f} \rhd \mathtt{g} \rangle \leftrightarrow \neg\langle \mathtt{g} \rhd \mathtt{f} \rangle$ for all $\mathtt{f}, \mathtt{g}$ with $\mathtt{f} \neq \mathtt{g}$

# An automatic solution?

Problem: how to know if rules are oriented by LPO? (And find $\rhd$?)

Problem reformulation:

*Do there exist a symbol ordering $\rhd$,*
*and a sequence of proof steps*
*such that the given inequalities $s \succ_{\mathrm{LPO}} t$ all hold?*

This is a SAT problem!

Idea:

- encode the ordering using variables $\langle \mathtt{f} \rhd \mathtt{g} \rangle$ for all $\mathtt{f}, \mathtt{g}$
- require: $\neg\langle \mathtt{f} \rhd \mathtt{f} \rangle$ for all $\mathtt{f}$
- require: $\langle \mathtt{f} \rhd \mathtt{g} \rangle \leftrightarrow \neg\langle \mathtt{g} \rhd \mathtt{f} \rangle$ for all $\mathtt{f}, \mathtt{g}$ with $\mathtt{f} \neq \mathtt{g}$
- require: $\langle \mathtt{f} \rhd \mathtt{g} \rangle \wedge \langle \mathtt{g} \rhd \mathtt{h} \rangle \to \langle \mathtt{f} \rhd \mathtt{h} \rangle$ for all $\mathtt{f}, \mathtt{g}, \mathtt{h}$

# An automatic solution?

**Problem:** how to know if rules are oriented by LPO? (And find $\rhd$?)

**Problem reformulation:**

*Do there exist a symbol ordering $\rhd$,*
*and a sequence of proof steps*
*such that the given inequalities $s \succ_{\mathrm{LPO}} t$ all hold?*

This is a SAT problem!

**Idea:**

- encode the ordering using variables $\langle \mathtt{f} \rhd \mathtt{g} \rangle$ for all $\mathtt{f}, \mathtt{g}$
- require: $\neg\langle \mathtt{f} \rhd \mathtt{f} \rangle$ for all $\mathtt{f}$
- require: $\langle \mathtt{f} \rhd \mathtt{g} \rangle \leftrightarrow \neg\langle \mathtt{g} \rhd \mathtt{f} \rangle$ for all $\mathtt{f}, \mathtt{g}$ with $\mathtt{f} \neq \mathtt{g}$
- require: $\langle \mathtt{f} \rhd \mathtt{g} \rangle \wedge \langle \mathtt{g} \rhd \mathtt{h} \rangle \rightarrow \langle \mathtt{f} \rhd \mathtt{h} \rangle$ for all $\mathtt{f}, \mathtt{g}, \mathtt{h}$
- encode the requirements for each inequality $\ell \succ_{\mathrm{LPO}} r$!

# Automation as one big formula

# Automation as one big formula

$$\mathtt{f}(\mathtt{g}(x)) \succ_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x))$$

# Automation as one big formula

$$
\begin{array}{ll}
\mathtt{g}(x) \succeq_{\text{LPO}} \mathtt{h}(\mathtt{f}(x)) & \vee \\
(\ \langle \mathtt{f} \rhd \mathtt{h} \rangle \wedge \mathtt{f}(\mathtt{g}(x)) \succ_{\text{LPO}} \mathtt{f}(x)\ ) &
\end{array}
$$

# Automation as one big formula

$$
\begin{array}{ll}
x \succeq_{\text{LPO}} \mathtt{h}(\mathtt{f}(x)) & \vee \\
(\ \langle \mathtt{g} \rhd \mathtt{h} \rangle \wedge \mathtt{g}(x) \succ_{\text{LPO}} \mathtt{f}(x)\ ) & \vee \\
(\ \langle \mathtt{f} \rhd \mathtt{h} \rangle \wedge (\ \mathtt{g}(x) \succeq_{\text{LPO}} \mathtt{f}(x) \vee & \\
\qquad\qquad (\ \mathtt{f}(\mathtt{g}(x)) \succ_{\text{LPO}} x \wedge \mathtt{g}(x) \succ_{\text{LPO}} x\ )\ )\ ) &
\end{array}
$$

# Automation as one big formula

$$
\begin{array}{l}
\bot \\
( \langle \mathtt{g} \rhd \mathtt{h} \rangle \land \quad (x \succeq_{\mathrm{LPO}} \mathtt{f}(x) \lor (\langle \mathtt{g} \rhd \mathtt{f} \rangle \land \mathtt{g}(x) \succ_{\mathrm{LPO}} x)\ ) \\
( \langle \mathtt{f} \rhd \mathtt{h} \rangle \land \ ( \ (x \succeq_{\mathrm{LPO}} \mathtt{f}(x) \lor (\langle \mathtt{g} \rhd \mathtt{f} \rangle \land \mathtt{g}(x) \succ_{\mathrm{LPO}} x)\ ) \lor (\top \land \top)\ )\ )
\end{array}
$$

# Automation as one big formula

$$
\begin{array}{l}
\bot \hfill \lor \\
(\ \langle \mathtt{g} \rhd \mathtt{h} \rangle \land \quad (\bot \lor (\langle \mathtt{g} \rhd \mathtt{f} \rangle \land \top)\ ) \hfill \lor \\
(\ \langle \mathtt{f} \rhd \mathtt{h} \rangle \land \ (\ (\bot \lor (\langle \mathtt{g} \rhd \mathtt{f} \rangle \land \top)\ ) \lor (\top \land \top)\ )\ )
\end{array}
$$

# Automation as one big formula

$$
\begin{array}{l}
\bot \hfill \vee \\
(\langle \mathtt{g} \rhd \mathtt{h} \rangle \wedge \quad (\bot \vee (\langle \mathtt{g} \rhd \mathtt{f} \rangle \wedge \top))) \hfill \vee \\
(\langle \mathtt{f} \rhd \mathtt{h} \rangle \wedge \ ( \ (\bot \vee (\langle \mathtt{g} \rhd \mathtt{f} \rangle \wedge \top)) \vee (\top \wedge \top)))
\end{array}
$$

This approach has some downsides:

# Automation as one big formula

$$
\begin{array}{l}
\bot \quad \vee \\
(\,\langle \mathtt{g} \rhd \mathtt{h}\rangle \wedge (\bot \vee (\langle \mathtt{g} \rhd \mathtt{f}\rangle \wedge \top))\,) \quad \vee \\
(\,\langle \mathtt{f} \rhd \mathtt{h}\rangle \wedge (\,(\bot \vee (\langle \mathtt{g} \rhd \mathtt{f}\rangle \wedge \top)) \vee (\top \wedge \top)\,)\,)
\end{array}
$$

This approach has some downsides:

- How to recover the proof that each $\ell \succ_{\mathrm{LPO}} r$?

# Automation as one big formula

$$
\begin{array}{l}
\bot \hspace{20em} \vee \\
(\, \langle \mathtt{g} \rhd \mathtt{h} \rangle \wedge \quad (\bot \vee (\langle \mathtt{g} \rhd \mathtt{f} \rangle \wedge \top)\,) \,) \hspace{6em} \vee \\
(\, \langle \mathtt{f} \rhd \mathtt{h} \rangle \wedge \; (\; (\bot \vee (\langle \mathtt{g} \rhd \mathtt{f} \rangle \wedge \top)\,) \vee (\top \wedge \top)\,)\,)
\end{array}
$$

This approach has some downsides:

- How to recover the proof that each $\ell \succ_{\mathrm{LPO}} r$?
- Risk of exponential blowup.

# Automation as one big formula

$$
\begin{array}{l}
\bot \hfill \vee \\
(\ \langle \mathtt{g} \rhd \mathtt{h} \rangle \wedge \qquad (\bot \vee (\langle \mathtt{g} \rhd \mathtt{f} \rangle \wedge \top)\ ) \hfill \vee \\
(\ \langle \mathtt{f} \rhd \mathtt{h} \rangle \wedge \ (\ (\bot \vee (\langle \mathtt{g} \rhd \mathtt{f} \rangle \wedge \top)\ ) \vee (\top \wedge \top)\ )\ )
\end{array}
$$

This approach has some downsides:

- How to recover the proof that each $\ell \succ_{\mathrm{LPO}} r$?
- Risk of exponential blowup.

$$
\begin{array}{l}
\bot \\
(\ \langle \mathtt{g} \rhd \mathtt{h} \rangle \wedge \qquad (x \succeq_{\mathrm{LPO}} \mathtt{f}(x) \vee (\langle \mathtt{g} \rhd \mathtt{f} \rangle \wedge \underline{\mathtt{g}(x) \succ_{\mathrm{LPO}} x})\ ) \\
(\ \langle \mathtt{f} \rhd \mathtt{h} \rangle \wedge \ (\ (x \succeq_{\mathrm{LPO}} \mathtt{f}(x) \vee (\langle \mathtt{g} \rhd \mathtt{f} \rangle \wedge \underline{\mathtt{g}(x) \succ_{\mathrm{LPO}} x})\ ) \vee (\top \wedge \top
\end{array}
$$

# Automation through “defining formulas”

Important observation:

# Automation through “defining formulas”

Important observation:

*Whenever* $a \succ_{\mathrm{LPO}} b$ *is used in the proof of* $s \succ_{\mathrm{LPO}} t$*:*
$a$ *is a subterm of* $s$*, and* $b$ *is a subterm of* $t$*.*

# Automation through "defining formulas"

## Important observation:

*Whenever* $a \succ_{\mathrm{LPO}} b$ *is used in the proof of* $s \succ_{\mathrm{LPO}} t$*:*
$a$ *is a subterm of* $s$*, and* $b$ *is a subterm of* $t$*.*

Aside from the variables $\langle \texttt{f} \rhd \texttt{g} \rangle$, we introduce:

For every subterm $a$ of $s$;

for every subterm $b$ of $t$;

for every relation $\sharp$ in $\{\succ_{\mathrm{LPO}}, \succ_{\mathrm{LPO}}^{\text{sub}}, \succ_{\mathrm{LPO}}^{\text{copy}}, \succ_{\mathrm{LPO}}^{\text{lex}}\}$:

a variable $\langle a \sharp b \rangle$.

# Automation through "defining formulas"

Important observation:

> *Whenever $a \succ_{\text{LPO}} b$ is used in the proof of $s \succ_{\text{LPO}} t$: $a$ is a subterm of $s$, and $b$ is a subterm of $t$.*

Aside from the variables $\langle \mathtt{f} \rhd \mathtt{g} \rangle$, we introduce:

For every subterm $a$ of $s$;

for every subterm $b$ of $t$;

for every relation $\sharp$ in $\{\succ_{\text{LPO}}, \succ_{\text{LPO}}^{\text{sub}}, \succ_{\text{LPO}}^{\text{copy}}, \succ_{\text{LPO}}^{\text{lex}}\}$:

a variable $\langle a \sharp b \rangle$.

Denote $\langle a \succeq_{\text{LPO}} b \rangle$ for either $\top$ (if $a = b$) or $\langle a \succ_{\text{LPO}} b \rangle$.

# Defining formulas per variable

For each variable $\langle a \sharp b \rangle$ we now require the **defining formula**.

# Defining formulas per variable

For each variable $\langle a \sharp b \rangle$ we now require the **defining formula.**

For example, if $a = \mathtt{f}(x, s, y)$ and $b = \mathtt{f}(x, t, u)$ with $s \neq t$:

# Defining formulas per variable

For each variable $\langle a \sharp b \rangle$ we now require the **defining formula.**

For example, if $a = \mathtt{f}(x, s, y)$ and $b = \mathtt{f}(x, t, u)$ with $s \neq t$:

- $\langle a \succ_{\mathrm{LPO}} b \rangle \to \langle a \succ_{\mathrm{LPO}}^{\textsf{sub}} b \rangle \vee \langle a \succ_{\mathrm{LPO}}^{\textsf{copy}} b \rangle \vee \langle a \succ_{\mathrm{LPO}}^{\textsf{lex}} b \rangle$

# Defining formulas per variable

For each variable $\langle a \sharp b \rangle$ we now require the **defining formula.**

For example, if $a = \mathtt{f}(x, s, y)$ and $b = \mathtt{f}(x, t, u)$ with $s \neq t$:

- $\langle a \succ_{\mathrm{LPO}} b \rangle \to \langle a \succ_{\mathrm{LPO}}^{\mathsf{sub}} b \rangle \vee \langle a \succ_{\mathrm{LPO}}^{\mathsf{copy}} b \rangle \vee \langle a \succ_{\mathrm{LPO}}^{\mathsf{lex}} b \rangle$
- $\langle a \succ_{\mathrm{LPO}}^{\mathsf{sub}} b \rangle \to \langle x \succeq_{\mathrm{LPO}} b \rangle \vee \langle s \succeq_{\mathrm{LPO}} b \rangle \vee \langle y \succeq_{\mathrm{LPO}} b \rangle$

# Defining formulas per variable

For each variable $\langle a \sharp b \rangle$ we now require the **defining formula.**

For example, if $a = \mathtt{f}(x, s, y)$ and $b = \mathtt{f}(x, t, u)$ with $s \neq t$:

- $\langle a \succ_{\mathrm{LPO}} b \rangle \to \langle a \succ_{\mathrm{LPO}}^{\text{sub}} b \rangle \vee \langle a \succ_{\mathrm{LPO}}^{\text{copy}} b \rangle \vee \langle a \succ_{\mathrm{LPO}}^{\text{lex}} b \rangle$
- $\langle a \succ_{\mathrm{LPO}}^{\text{sub}} b \rangle \to \langle x \succeq_{\mathrm{LPO}} b \rangle \vee \langle s \succeq_{\mathrm{LPO}} b \rangle \vee \langle y \succeq_{\mathrm{LPO}} b \rangle$
- $\neg \langle a \succ_{\mathrm{LPO}}^{\text{copy}} b \rangle$ (since $\mathtt{f} \rhd \mathtt{f}$ cannot hold)

# Defining formulas per variable

For each variable $\langle a \sharp b \rangle$ we now require the **defining formula.**

For example, if $a = \mathtt{f}(x, s, y)$ and $b = \mathtt{f}(x, t, u)$ with $s \neq t$:

- $\langle a \succ_{\mathrm{LPO}} b \rangle \to \langle a \succ_{\mathrm{LPO}}^{\textsf{sub}} b \rangle \vee \langle a \succ_{\mathrm{LPO}}^{\textsf{copy}} b \rangle \vee \langle a \succ_{\mathrm{LPO}}^{\textsf{lex}} b \rangle$
- $\langle a \succ_{\mathrm{LPO}}^{\textsf{sub}} b \rangle \to \langle x \succeq_{\mathrm{LPO}} b \rangle \vee \langle s \succeq_{\mathrm{LPO}} b \rangle \vee \langle y \succeq_{\mathrm{LPO}} b \rangle$
- $\neg \langle a \succ_{\mathrm{LPO}}^{\textsf{copy}} b \rangle$ (since $\mathtt{f} \rhd \mathtt{f}$ cannot hold)
- $\langle a \succ_{\mathrm{LPO}}^{\textsf{lex}} b \rangle \to \langle a \succ_{\mathrm{LPO}} x \rangle \wedge \langle a \succ_{\mathrm{LPO}} t \rangle \wedge \langle a \succ_{\mathrm{LPO}} u \rangle \wedge \langle s \succ_{\mathrm{LPO}} t \rangle$

# Defining formulas: formally

For all subterms $a$ of $s$ and $b$ of $t$, add the following formulas:

# Defining formulas: formally

For all subterms $a$ of $s$ and $b$ of $t$, add the following formulas:

- if $a = b$, then $\neg\langle a \sharp b\rangle$ for all $\sharp \in \{\succ_{\mathrm{LPO}}, \succ^{\text{sub}}_{\mathrm{LPO}}, \succ^{\text{copy}}_{\mathrm{LPO}}, \succ^{\text{lex}}_{\mathrm{LPO}}\}$
- otherwise, if $a$ is a variable: $\neg\langle a \sharp b\rangle$ for all $\sharp \in \{\succ_{\mathrm{LPO}}, \succ^{\text{sub}}_{\mathrm{LPO}}, \succ^{\text{copy}}_{\mathrm{LPO}}, \succ^{\text{lex}}_{\mathrm{LPO}}\}$
- otherwise, if $a = \mathtt{f}(a_1, \dots, a_n)$:
  - $\langle a \succ_{\mathrm{LPO}} b\rangle \rightarrow \langle a \succ^{\text{sub}}_{\mathrm{LPO}} b\rangle \vee \langle a \succ^{\text{copy}}_{\mathrm{LPO}} b\rangle \vee \langle a \succ^{\text{lex}}_{\mathrm{LPO}} b\rangle$
  - $\langle a \succ^{\text{sub}}_{\mathrm{LPO}} b\rangle \rightarrow \langle a_1 \succeq_{\mathrm{LPO}} b\rangle \vee \dots \vee \langle a_n \succeq_{\mathrm{LPO}} b\rangle$
  - if $b = \mathtt{f}(b_1, \dots, b_n)$, and $i$ is the lowest index such that $a_i \neq b_i$, then:
    - $\neg\langle a \succ^{\text{copy}}_{\mathrm{LPO}} b\rangle$
    - $\langle a \succ^{\text{lex}}_{\mathrm{LPO}} b\rangle \rightarrow \langle a \succ_{\mathrm{LPO}} b_1\rangle \wedge \dots \wedge \langle a \succ_{\mathrm{LPO}} b_n\rangle \wedge \langle a_i \succ_{\mathrm{LPO}} b_i\rangle$
  - otherwise, if $b = \mathtt{g}(b_1, \dots, b_m)$ with $\mathtt{f} \neq \mathtt{g}$ then:
    - $\langle a \succ^{\text{copy}}_{\mathrm{LPO}} b\rangle \rightarrow \langle \mathtt{f} \rhd \mathtt{g}\rangle \wedge \langle a \succ_{\mathrm{LPO}} b_1\rangle \wedge \dots \wedge \langle a \succ_{\mathrm{LPO}} b_m\rangle$
    - $\neg\langle a \succ^{\text{lex}}_{\mathrm{LPO}} b\rangle$
  - otherwise, $\neg\langle a \sharp b\rangle$ for $\sharp \in \{\succ^{\text{copy}}_{\mathrm{LPO}}, \succ^{\text{lex}}_{\mathrm{LPO}}\}$

# Example: $\mathtt{f}(\mathtt{g}(x)) \succ_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x))$

# Example: $\mathtt{f}(\mathtt{g}(x)) \succ_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x))$

$$
\begin{array}{rcl}
\langle \mathtt{f}(\mathtt{g}(x)) \succ_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x)) \rangle & \to & \langle \mathtt{f}(\mathtt{g}(x)) \succ^{\text{sub}}_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x)) \rangle \vee \\
 & & \langle \mathtt{f}(\mathtt{g}(x)) \succ^{\text{copy}}_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x)) \rangle \vee \\
 & & \langle \mathtt{f}(\mathtt{g}(x)) \succ^{\text{lex}}_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x)) \rangle \vee \\
\langle \mathtt{f}(\mathtt{g}(x)) \succ^{\text{sub}}_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x)) \rangle & \to & \langle \mathtt{g}(x) \succ_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x)) \rangle \\
\langle \mathtt{f}(\mathtt{g}(x)) \succ^{\text{copy}}_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x)) \rangle & \to & \langle \mathtt{f} \rhd \mathtt{h} \rangle \wedge \mathtt{f}(\mathtt{g}(x)) \succ_{\mathrm{LPO}} \mathtt{f}(x) \rangle \\
\neg \langle \mathtt{f}(\mathtt{g}(x)) \succ^{\text{lex}}_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x)) \rangle & & \\
\langle \mathtt{g}(x) \succ_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x)) \rangle & \to & \langle \mathtt{g}(x) \succ^{\text{sub}}_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x)) \rangle \vee \\
 & & \langle \mathtt{g}(x) \succ^{\text{copy}}_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x)) \rangle \vee \\
 & & \langle \mathtt{g}(x) \succ^{\text{lex}}_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x)) \rangle \\
\langle \mathtt{g}(x) \succ^{\text{copy}}_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x)) \rangle & \to & \langle \mathtt{g} \rhd \mathtt{h} \rangle \wedge \langle \mathtt{g}(x) \succ_{\mathrm{LPO}} \mathtt{f}(x) \rangle \\
\langle \mathtt{g}(x) \succ_{\mathrm{LPO}} \mathtt{f}(x) \rangle & \to & \langle \mathtt{g}(x) \succ^{\text{sub}}_{\mathrm{LPO}} \mathtt{f}(x) \rangle \vee \langle \mathtt{g}(x) \succ^{\text{copy}}_{\mathrm{LPO}} \mathtt{f}(x) \rangle \vee \\
\langle \mathtt{g}(x) \succ^{\text{copy}}_{\mathrm{LPO}} \mathtt{f}(x) \rangle & \to & \langle \mathtt{g} \rhd \mathtt{f} \rangle \wedge \mathtt{g}(x) \succ_{\mathrm{LPO}} x \rangle \\
\langle \mathtt{g}(x) \succ_{\mathrm{LPO}} x \rangle & \to & \langle \mathtt{g}(x) \succ^{\text{sub}}_{\mathrm{LPO}} x \rangle \vee \langle \mathtt{g}(x) \succ^{\text{copy}}_{\mathrm{LPO}} x \rangle \vee \langle \mathtt{g}(x) \succ \\
\langle \mathtt{g}(x) \succ^{\text{sub}}_{\mathrm{LPO}} x \rangle & \to & \top \\
 & \ldots & 
\end{array}
$$

# Finishing up the SAT encoding

We also require:

$$\langle \ell \succ_{\mathrm{LPO}} r \rangle$$

since the topmost inequality should hold.

# Finishing up the SAT encoding

We also require:

$$\langle \ell \succ_{\text{LPO}} r \rangle$$

since the topmost inequality should hold.

The full formula has:

- $|\ell| * |r| * 4 + 1$ formulas per rule
- $\mathcal{O}(|\Sigma|^3)$ formulas to define $\rhd$

# Finishing up the SAT encoding

We also require:

$$\langle \ell \succ_{\mathrm{LPO}} r \rangle$$

since the topmost inequality should hold.

The full formula has:

- $|\ell| * |r| * 4 + 1$ formulas per rule
- $\mathcal{O}(|\Sigma|^3)$ formulas to define $\triangleright$

These formulas are small, and easily converted to CNF by the Tseitin transformation.

# Finishing up the SAT encoding

We also require:

$$\langle \ell \succ_{\text{LPO}} r \rangle$$

since the topmost inequality should hold.

The full formula has:

- $|\ell| * |r| * 4 + 1$ formulas per rule
- $\mathcal{O}(|\Sigma|^3)$ formulas to define $\triangleright$

These formulas are small, and easily converted to CNF by the Tseitin transformation.

A satisfying assignment immediately allows us to read off the proof!

# Reading off the proof

$$
\begin{array}{rcl}
\underline{\langle \mathtt{f}(\mathtt{g}(x)) \succ_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x))\rangle} & \rightarrow & \underline{\langle \mathtt{f}(\mathtt{g}(x)) \succ^{\textsf{sub}}_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x))\rangle} \vee \\
 & & \langle \mathtt{f}(\mathtt{g}(x)) \succ^{\textsf{copy}}_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x))\rangle \vee \\
 & & \langle \mathtt{f}(\mathtt{g}(x)) \succ^{\textsf{lex}}_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x))\rangle \vee \\
\underline{\langle \mathtt{f}(\mathtt{g}(x)) \succ^{\textsf{sub}}_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x))\rangle} & \rightarrow & \langle \underline{\mathtt{g}(x) \succ_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x))\rangle} \\
\langle \mathtt{f}(\mathtt{g}(x)) \succ^{\textsf{copy}}_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x))\rangle & \rightarrow & \langle \mathtt{f} \rhd \mathtt{h}\rangle \wedge \mathtt{f}(\mathtt{g}(x)) \succ_{\mathrm{LPO}} \mathtt{f}(x)\rangle \\
\neg\langle \mathtt{f}(\mathtt{g}(x)) \succ^{\textsf{lex}}_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x))\rangle & & \\
\underline{\langle \mathtt{g}(x) \succ_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x))\rangle} & \rightarrow & \langle \mathtt{g}(x) \succ^{\textsf{sub}}_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x))\rangle \vee \\
 & & \underline{\langle \mathtt{g}(x) \succ^{\textsf{copy}}_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x))\rangle} \vee \\
 & & \langle \mathtt{g}(x) \succ^{\textsf{lex}}_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x))\rangle \\
\underline{\langle \mathtt{g}(x) \succ^{\textsf{copy}}_{\mathrm{LPO}} \mathtt{h}(\mathtt{f}(x))\rangle} & \rightarrow & \underline{\langle \mathtt{g} \rhd \mathtt{h}\rangle} \wedge \underline{\langle \mathtt{g}(x) \succ_{\mathrm{LPO}} \mathtt{f}(x)\rangle} \\
\underline{\langle \mathtt{g}(x) \succ_{\mathrm{LPO}} \mathtt{f}(x)\rangle} & \rightarrow & \langle \mathtt{g}(x) \succ^{\textsf{sub}}_{\mathrm{LPO}} \mathtt{f}(x)\rangle \vee \underline{\langle \mathtt{g}(x) \succ^{\textsf{copy}}_{\mathrm{LPO}} \mathtt{f}(x)\rangle} \vee \\
\underline{\langle \mathtt{g}(x) \succ^{\textsf{copy}}_{\mathrm{LPO}} \mathtt{f}(x)\rangle} & \rightarrow & \underline{\langle \mathtt{g} \rhd \mathtt{f}\rangle} \wedge \underline{\mathtt{g}(x) \succ_{\mathrm{LPO}} x\rangle} \\
\underline{\langle \mathtt{g}(x) \succ_{\mathrm{LPO}} x\rangle} & \rightarrow & \underline{\langle \mathtt{g}(x) \succ^{\textsf{sub}}_{\mathrm{LPO}} x\rangle} \vee \langle \mathtt{g}(x) \succ^{\textsf{copy}}_{\mathrm{LPO}} x\rangle \vee \langle \mathtt{g}(x) \succ \\
\langle \mathtt{g}(x) \succ^{\textsf{sub}}_{\mathrm{LPO}} x\rangle & \rightarrow & \top
\end{array}
$$

# Variants of LPO

Many recursive path orderings exist:

# Variants of LPO

Many recursive path orderings exist:

- for **(lex)**, compare $[s_1, \ldots, s_n]$ and $[t_1, \ldots, t_n]$ differently:

# Variants of LPO

Many recursive path orderings exist:

- for **(lex)**, compare $[s_1, \ldots, s_n]$ and $[t_1, \ldots, t_n]$ differently:
  - placewise comparison

# Variants of LPO

Many recursive path orderings exist:

- for **(lex)**, compare $[s_1, \ldots, s_n]$ and $[t_1, \ldots, t_n]$ differently:
  - placewise comparison
  - the multiset comparison (see the assignment)

# Variants of LPO

Many recursive path orderings exist:

- for **(lex)**, compare $[s_1, \ldots, s_n]$ and $[t_1, \ldots, t_n]$ differently:
  - placewise comparison
  - the multiset comparison (see the assignment)
  - either lexicographic or multiset, depending on the function symbol

# Variants of LPO

Many recursive path orderings exist:

- for **(lex)**, compare $[s_1, \ldots, s_n]$ and $[t_1, \ldots, t_n]$ differently:
  - placewise comparison
  - the multiset comparison (see the assignment)
  - either lexicographic or multiset, depending on the function symbol
- using a **quasi-ordering** for the symbol comparison $\rhd$

# Variants of LPO

Many recursive path orderings exist:

- for **(lex)**, compare $[s_1, \ldots, s_n]$ and $[t_1, \ldots, t_n]$ differently:
  - placewise comparison
  - the multiset comparison (see the assignment)
  - either lexicographic or multiset, depending on the function symbol
- using a **quasi-ordering** for the symbol comparison $\rhd$
- if $\mathsf{c}$ is the smallest symbol in $\rhd$ and has arity $0$, letting $x \succeq_{\mathrm{LPO}} \mathsf{c}$ also for variables

# Subterm property

The lexicographic path ordering (and all variations) has the following property:

**Theorem**

If $s$ is a proper subterm of $t$, then $t \succ_{\mathrm{LPO}} s$.

# Subterm property

The lexicographic path ordering (and all variations) has the following property:

**Theorem**

If $s$ is a proper subterm of $t$, then $t \succ_{\mathrm{LPO}} s$.

**Proof.** This easily follows from rule **(sub)**. $\square$

# Quiz

1. What is the difference between weak and strong normalisation?
2. What is the difference between local confluence and general confluence?
3. Use the lexicographic path ordering (by hand) to prove termination of:

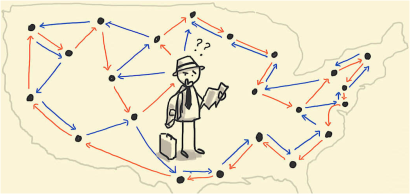
$$\begin{aligned} \mathtt{f}(\mathtt{g}(x), \mathtt{g}(\mathtt{b})) &\Rightarrow \mathtt{f}(x, x) \\ \mathtt{g}(\mathtt{a}) &\Rightarrow \mathtt{b} \\ \mathtt{b} &\Rightarrow \mathtt{a} \end{aligned}$$

4. What properties should a relation $\succ$ satisfy to be a reduction order?

# Some Advice for the Practical assignment

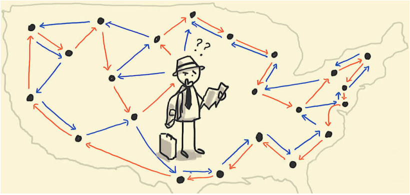
# Some Advice for the Practical assignment

Suppose: you have to find a solution for the Traveling Salesman Problem.



# Some Advice for the Practical assignment

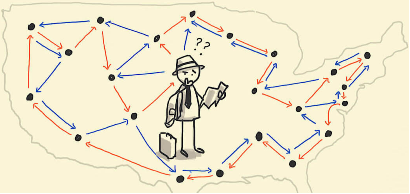
Suppose: you have to find a solution for the Traveling Salesman Problem.




Problem: find **the shortest** route visiting a number of cities

# Some Advice for the Practical assignment

Suppose: you have to find a solution for the Traveling Salesman Problem.




Problem: find **the shortest** route visiting a number of cities

Solution:

- encode, for given $N$, the problem "find a route $\leq N$" into SMT
- use binary search to find the smallest $N$ for which this is satisfiable