Automatic Termination Tools in Knuth-Bendix Completion

Sarah Winkler

Master Thesis
Institute of Computer Science
University of Innsbruck

2008/11/13

Outline

- Preliminaries: Term Rewrite Systems

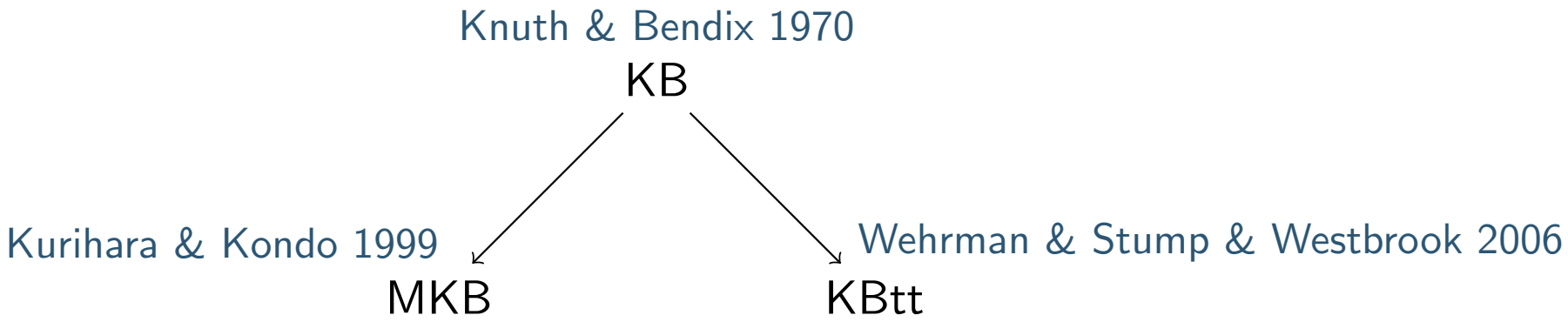
Outline

- Preliminaries: Term Rewrite Systems
- Completion Inference Systems

Knuth & Bendix 1970
KB

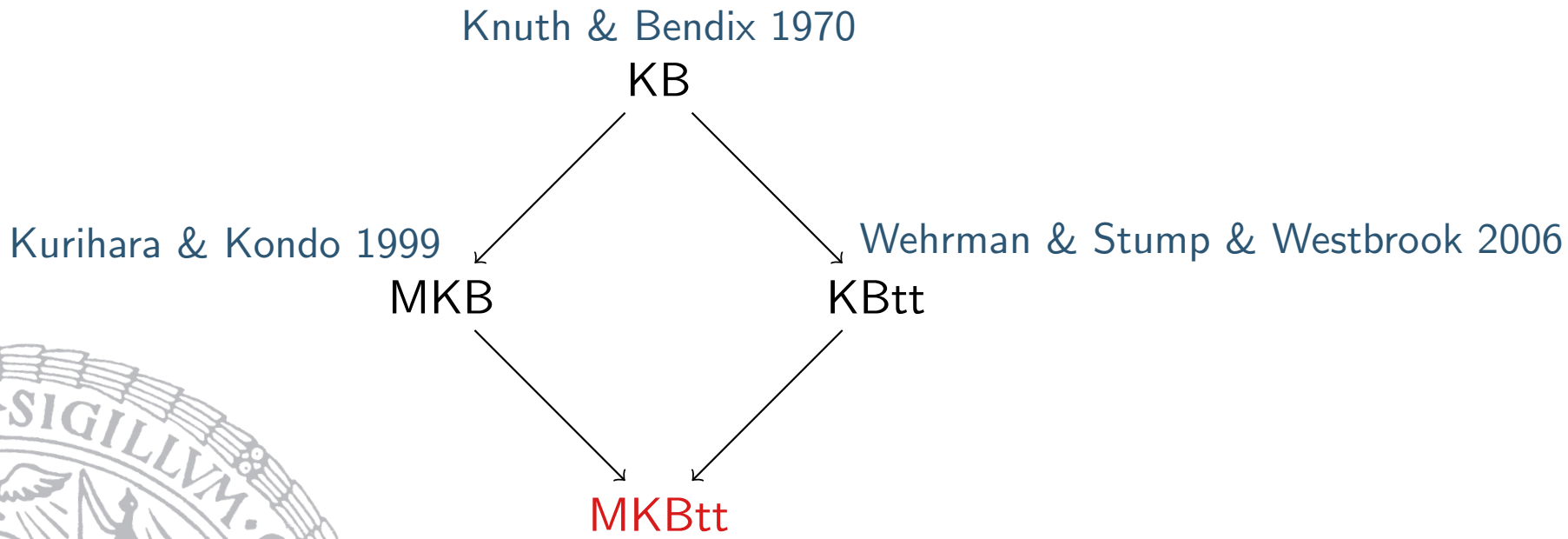
Outline

- Preliminaries: Term Rewrite Systems
- Completion Inference Systems



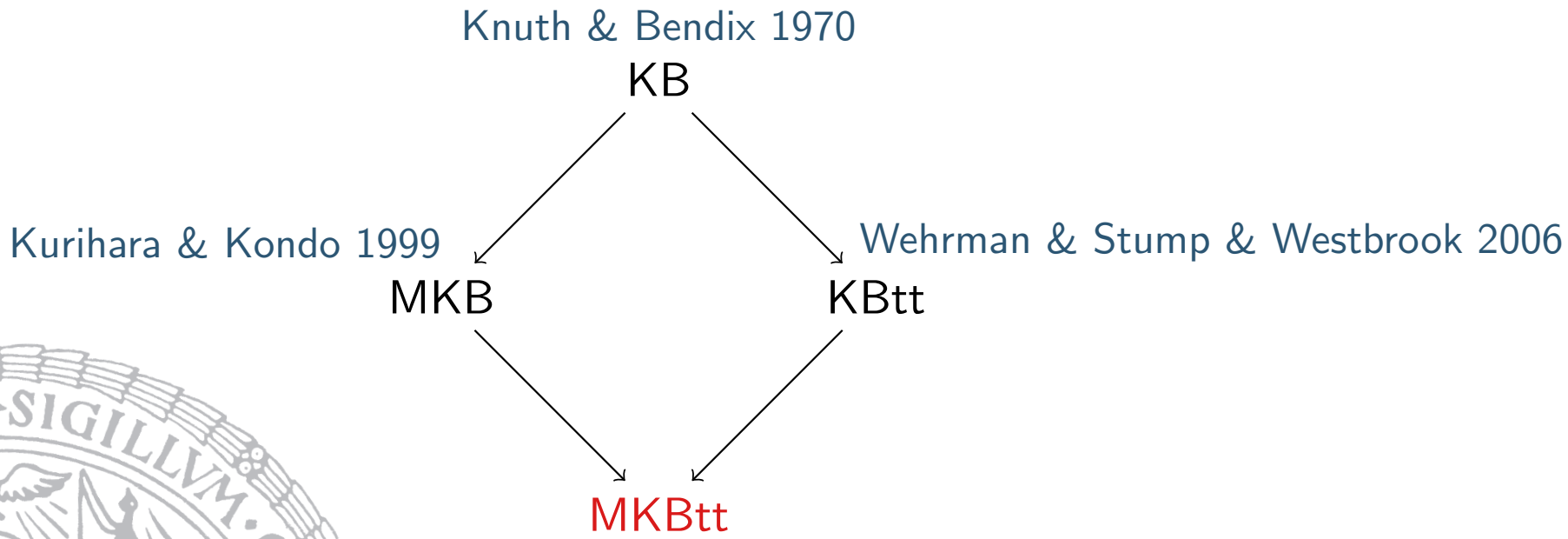
Outline

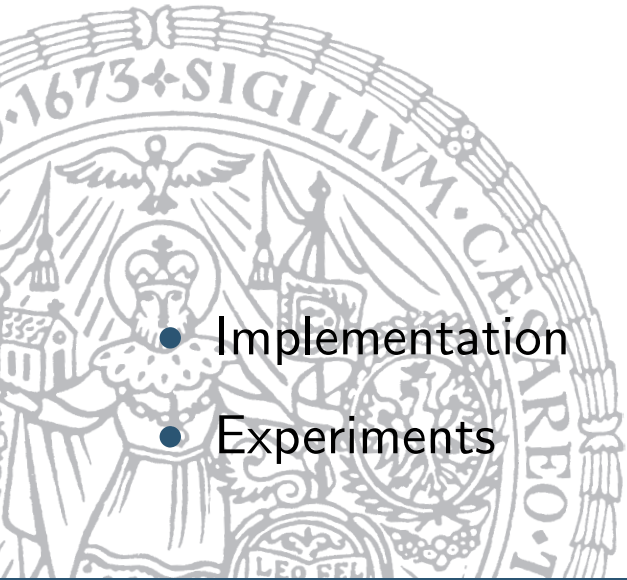
- Preliminaries: Term Rewrite Systems
- Completion Inference Systems



Outline

- Preliminaries: Term Rewrite Systems
- Completion Inference Systems





- Implementation
- Experiments

Example 1: Decimal Addition

$$
\begin{array}{llll}
1{+}1 \to 2 & 1{+}2 \to 3 & \ldots & 1{+}9 \to 1{\cdot}0 \\
2{+}1 \to 3 & 2{+}2 \to 4 & \ldots & 2{+}9 \to 1{\cdot}1 \\
\vdots & \vdots & & \vdots \\
9{+}1 \to 1{\cdot}0 & 9{+}2 \to 1{\cdot}1 & \ldots & 9{+}9 \to 1{\cdot}8
\end{array}
$$

Example 1: Decimal Addition

$$
\begin{array}{llll}
1{+}1 \to 2 & 1{+}2 \to 3 & \ldots & 1{+}9 \to 1{\cdot}0 \\
2{+}1 \to 3 & 2{+}2 \to 4 & \ldots & 2{+}9 \to 1{\cdot}1 \\
\vdots & \vdots & & \vdots \\
9{+}1 \to 1{\cdot}0 & 9{+}2 \to 1{\cdot}1 & \ldots & 9{+}9 \to 1{\cdot}8 \\
\\
0{+}x \to x & x{+}0 \to x & & 0{\cdot}x \to x
\end{array}
$$

Example 1: Decimal Addition

$$
\begin{array}{lll}
1{+}1 \to 2 & 1{+}2 \to 3 & \ldots \quad 1{+}9 \to 1{\cdot}0 \\
2{+}1 \to 3 & 2{+}2 \to 4 & \ldots \quad 2{+}9 \to 1{\cdot}1 \\
\vdots & \vdots & \vdots \\
9{+}1 \to 1{\cdot}0 & 9{+}2 \to 1{\cdot}1 & \ldots \quad 9{+}9 \to 1{\cdot}8
\end{array}
$$

$$
\begin{array}{lll}
0{+}x \to x & x{+}0 \to x & 0{\cdot}x \to x \\
x{\cdot}(y{\cdot}z) \to (x{+}y){\cdot}z & x{+}(y{\cdot}z) \to y{\cdot}(x{+}z) & (x{\cdot}y){+}z \to x{\cdot}(y{+}z)
\end{array}
$$

Example 1: Decimal Addition

$$\begin{array}{lll} 1{+}1 \to 2 & 1{+}2 \to 3 & \ldots \quad 1{+}9 \to 1{\cdot}0 \\ 2{+}1 \to 3 & 2{+}2 \to 4 & \ldots \quad 2{+}9 \to 1{\cdot}1 \\ \vdots & \vdots & \vdots \\ 9{+}1 \to 1{\cdot}0 & 9{+}2 \to 1{\cdot}1 & \ldots \quad 9{+}9 \to 1{\cdot}8 \end{array}$$

$$\begin{array}{lll} 0{+}x \to x & x{+}0 \to x & 0{\cdot}x \to x \\ x{\cdot}(y{\cdot}z) \to (x{+}y){\cdot}z & x{+}(y{\cdot}z) \to y{\cdot}(x{+}z) & (x{\cdot}y){+}z \to x{\cdot}(y{+}z) \end{array}$$

Compute

$$\underline{(3{\cdot}3){+}9} \to$$

Example 1: Decimal Addition

$$
\begin{array}{lll}
1{+}1 \to 2 & 1{+}2 \to 3 & \ldots \quad 1{+}9 \to 1{\cdot}0 \\
2{+}1 \to 3 & 2{+}2 \to 4 & \ldots \quad 2{+}9 \to 1{\cdot}1 \\
\vdots & \vdots & \vdots \\
9{+}1 \to 1{\cdot}0 & 9{+}2 \to 1{\cdot}1 & \ldots \quad 9{+}9 \to 1{\cdot}8
\end{array}
$$

$$
\begin{array}{lll}
0{+}x \to x & x{+}0 \to x & 0{\cdot}x \to x \\
x{\cdot}(y{\cdot}z) \to (x{+}y){\cdot}z & x{+}(y{\cdot}z) \to y{\cdot}(x{+}z) & (x{\cdot}y){+}z \to x{\cdot}(y{+}z)
\end{array}
$$

Compute

$$\underline{(3{\cdot}3){+}9} \to 3{\cdot}(\underline{3{+}9}) \to$$

Example 1: Decimal Addition

$$\begin{array}{lll} 1{+}1 \to 2 & 1{+}2 \to 3 & \ldots \quad 1{+}9 \to 1{\cdot}0 \\ 2{+}1 \to 3 & 2{+}2 \to 4 & \ldots \quad 2{+}9 \to 1{\cdot}1 \\ \vdots & \vdots & \vdots \\ 9{+}1 \to 1{\cdot}0 & 9{+}2 \to 1{\cdot}1 & \ldots \quad 9{+}9 \to 1{\cdot}8 \end{array}$$

$$\begin{array}{lll} 0{+}x \to x & x{+}0 \to x & 0{\cdot}x \to x \\ x{\cdot}(y{\cdot}z) \to (x{+}y){\cdot}z & x{+}(y{\cdot}z) \to y{\cdot}(x{+}z) & (x{\cdot}y){+}z \to x{\cdot}(y{+}z) \end{array}$$

Compute

$$\underline{(3{\cdot}3){+}9} \to 3{\cdot}(\underline{3{+}9}) \to \underline{3{\cdot}(1{\cdot}2)} \to$$

Example 1: Decimal Addition

$$\begin{array}{lll} 1{+}1 \to 2 & 1{+}2 \to 3 & \ldots \quad 1{+}9 \to 1{\cdot}0 \\ 2{+}1 \to 3 & 2{+}2 \to 4 & \ldots \quad 2{+}9 \to 1{\cdot}1 \\ \vdots & \vdots & \vdots \\ 9{+}1 \to 1{\cdot}0 & 9{+}2 \to 1{\cdot}1 & \ldots \quad 9{+}9 \to 1{\cdot}8 \end{array}$$

$$\begin{array}{lll} 0{+}x \to x & x{+}0 \to x & 0{\cdot}x \to x \\ x{\cdot}(y{\cdot}z) \to (x{+}y){\cdot}z & x{+}(y{\cdot}z) \to y{\cdot}(x{+}z) & (x{\cdot}y){+}z \to x{\cdot}(y{+}z) \end{array}$$

Compute

$$\underline{(3{\cdot}3){+}9} \to 3{\cdot}(\underline{3{+}9}) \to \underline{3{\cdot}(1{\cdot}2)} \to \underline{(3{+}1){\cdot}2} \to$$

Example 1: Decimal Addition

$$\begin{array}{lll} 1{+}1 \to 2 & 1{+}2 \to 3 & \ldots \quad 1{+}9 \to 1{\cdot}0 \\ 2{+}1 \to 3 & 2{+}2 \to 4 & \ldots \quad 2{+}9 \to 1{\cdot}1 \\ \vdots & \vdots & \vdots \\ 9{+}1 \to 1{\cdot}0 & 9{+}2 \to 1{\cdot}1 & \ldots \quad 9{+}9 \to 1{\cdot}8 \end{array}$$

$$\begin{array}{lll} 0{+}x \to x & x{+}0 \to x & 0{\cdot}x \to x \\ x{\cdot}(y{\cdot}z) \to (x{+}y){\cdot}z & x{+}(y{\cdot}z) \to y{\cdot}(x{+}z) & (x{\cdot}y){+}z \to x{\cdot}(y{+}z) \end{array}$$

Compute

$$\underline{(3{\cdot}3){+}9} \to 3{\cdot}(\underline{3{+}9}) \to \underline{3{\cdot}(1{\cdot}2)} \to \underline{(3{+}1){\cdot}2} \to 4{\cdot}2$$

Example 1: Decimal Addition

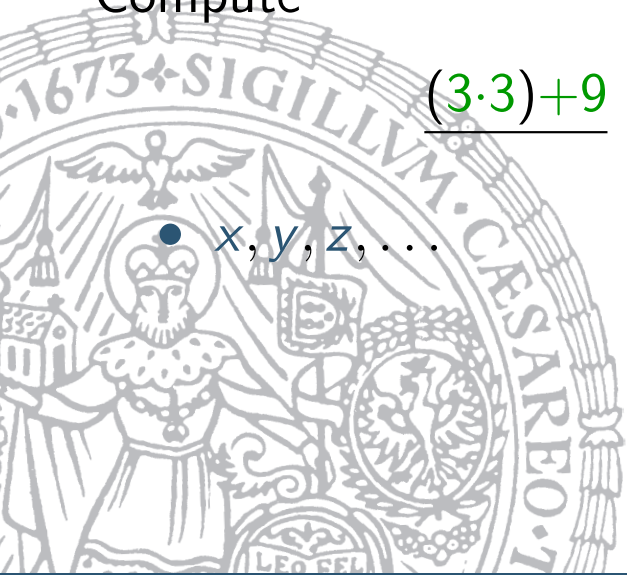
$$\begin{array}{lll} 1{+}1 \to 2 & 1{+}2 \to 3 & \ldots \quad 1{+}9 \to 1{\cdot}0 \\ 2{+}1 \to 3 & 2{+}2 \to 4 & \ldots \quad 2{+}9 \to 1{\cdot}1 \\ \vdots & \vdots & \vdots \\ 9{+}1 \to 1{\cdot}0 & 9{+}2 \to 1{\cdot}1 & \ldots \quad 9{+}9 \to 1{\cdot}8 \end{array}$$

$$\begin{array}{lll} 0{+}x \to x & x{+}0 \to x & 0{\cdot}x \to x \\ x{\cdot}(y{\cdot}z) \to (x{+}y){\cdot}z & x{+}(y{\cdot}z) \to y{\cdot}(x{+}z) & (x{\cdot}y){+}z \to x{\cdot}(y{+}z) \end{array}$$

Compute

$$\underline{(3{\cdot}3){+}9} \to 3{\cdot}(\underline{3{+}9}) \to \underline{3{\cdot}(1{\cdot}2)} \to \underline{(3{+}1){\cdot}2} \to 4{\cdot}2$$

- $x, y, z, \ldots$ variables $\mathcal{V}$



Example 1: Decimal Addition

$$\begin{array}{lll} 1{+}1 \to 2 & 1{+}2 \to 3 & \ldots \quad 1{+}9 \to 1{\cdot}0 \\ 2{+}1 \to 3 & 2{+}2 \to 4 & \ldots \quad 2{+}9 \to 1{\cdot}1 \\ \vdots & \vdots & \vdots \\ 9{+}1 \to 1{\cdot}0 & 9{+}2 \to 1{\cdot}1 & \ldots \quad 9{+}9 \to 1{\cdot}8 \end{array}$$

$$\begin{array}{lll} 0{+}x \to x & x{+}0 \to x & 0{\cdot}x \to x \\ x{\cdot}(y{\cdot}z) \to (x{+}y){\cdot}z & x{+}(y{\cdot}z) \to y{\cdot}(x{+}z) & (x{\cdot}y){+}z \to x{\cdot}(y{+}z) \end{array}$$

Compute

$$\underline{(3{\cdot}3){+}9} \to 3{\cdot}(\underline{3{+}9}) \to \underline{3{\cdot}(1{\cdot}2)} \to \underline{(3{+}1){\cdot}2} \to 4{\cdot}2$$

- $x, y, z, \ldots$
- $+, \cdot, 1, 2, \ldots, 9$

variables $\mathcal{V}$
function symbols $\mathcal{F}$

Example 1: Decimal Addition

$$\begin{array}{lll} 1{+}1 \to 2 & 1{+}2 \to 3 & \ldots \quad 1{+}9 \to 1{\cdot}0 \\ 2{+}1 \to 3 & 2{+}2 \to 4 & \ldots \quad 2{+}9 \to 1{\cdot}1 \\ \vdots & \vdots & \vdots \\ 9{+}1 \to 1{\cdot}0 & 9{+}2 \to 1{\cdot}1 & \ldots \quad 9{+}9 \to 1{\cdot}8 \end{array}$$

$$\begin{array}{lll} 0{+}x \to x & x{+}0 \to x & 0{\cdot}x \to x \\ x{\cdot}(y{\cdot}z) \to (x{+}y){\cdot}z & x{+}(y{\cdot}z) \to y{\cdot}(x{+}z) & (x{\cdot}y){+}z \to x{\cdot}(y{+}z) \end{array}$$

Compute

$$\underline{(3{\cdot}3){+}9} \to 3{\cdot}(\underline{3{+}9}) \to \underline{3{\cdot}(1{\cdot}2)} \to \underline{(3{+}1){\cdot}2} \to 4{\cdot}2$$

- $x, y, z, \ldots$
- $+, \cdot, 1, 2, \ldots, 9$
- 3, $(3{\cdot}3){+}9$, $y{\cdot}z$, x, $x{\cdot}5$, $\ldots$

variables $\mathcal{V}$
function symbols $\mathcal{F}$
terms $\mathcal{T}(\mathcal{F}, \mathcal{V})$

Example 1: Decimal Addition

$$\begin{array}{lll} 1{+}1 \to 2 & 1{+}2 \to 3 & \dots \quad 1{+}9 \to 1{\cdot}0 \\ 2{+}1 \to 3 & 2{+}2 \to 4 & \dots \quad 2{+}9 \to 1{\cdot}1 \\ \vdots & \vdots & \vdots \\ 9{+}1 \to 1{\cdot}0 & 9{+}2 \to 1{\cdot}1 & \dots \quad 9{+}9 \to 1{\cdot}8 \end{array}$$

$$\begin{array}{lll} 0{+}x \to x & x{+}0 \to x & 0{\cdot}x \to x \\ x{\cdot}(y{\cdot}z) \to (x{+}y){\cdot}z & x{+}(y{\cdot}z) \to y{\cdot}(x{+}z) & (x{\cdot}y){+}z \to x{\cdot}(y{+}z) \end{array}$$

Compute

$$\underline{(3{\cdot}3){+}9} \to 3{\cdot}(\underline{3{+}9}) \to \underline{3{\cdot}(1{\cdot}2)} \to \underline{(3{+}1){\cdot}2} \to 4{\cdot}2$$

- $x, y, z, \dots$
- $+, \cdot, 1, 2, \dots, 9$
- 3, $(3{\cdot}3){+}9$, $y{\cdot}z$, x, $x{\cdot}5$, $\dots$
- $5{+}8 \to 1{\cdot}3$

variables $\mathcal{V}$
function symbols $\mathcal{F}$
terms $\mathcal{T}(\mathcal{F}, \mathcal{V})$
rewrite rule

Example 1: Decimal Addition

$$\begin{array}{lll} 1{+}1 \to 2 & 1{+}2 \to 3 & \dots \quad 1{+}9 \to 1{\cdot}0 \\ 2{+}1 \to 3 & 2{+}2 \to 4 & \dots \quad 2{+}9 \to 1{\cdot}1 \\ \vdots & \vdots & \vdots \\ 9{+}1 \to 1{\cdot}0 & 9{+}2 \to 1{\cdot}1 & \dots \quad 9{+}9 \to 1{\cdot}8 \end{array}$$

$$\begin{array}{lll} 0{+}x \to x & x{+}0 \to x & 0{\cdot}x \to x \\ x{\cdot}(y{\cdot}z) \to (x{+}y){\cdot}z & x{+}(y{\cdot}z) \to y{\cdot}(x{+}z) & (x{\cdot}y){+}z \to x{\cdot}(y{+}z) \end{array}$$

Compute

$$\underline{(3{\cdot}3){+}9} \to 3{\cdot}(\underline{3{+}9}) \to \underline{3{\cdot}(1{\cdot}2)} \to \underline{(3{+}1){\cdot}2} \to 4{\cdot}2$$

- $x, y, z, \dots$
- $+, \cdot, 1, 2, \dots, 9$
- 3, $(3{\cdot}3){+}9$, $y{\cdot}z$, x, $x{\cdot}5$, $\dots$
- $5{+}8 \to 1{\cdot}3$
- $(3{+}0){\cdot}y \to_{x+0 \to x} 3{\cdot}y$

variables $\mathcal{V}$
function symbols $\mathcal{F}$
terms $\mathcal{T}(\mathcal{F}, \mathcal{V})$
rewrite rule
rewrite step

Example 1: Decimal Addition

$$\begin{array}{lll} 1{+}1 \to 2 & 1{+}2 \to 3 & \dots \quad 1{+}9 \to 1{\cdot}0 \\ 2{+}1 \to 3 & 2{+}2 \to 4 & \dots \quad 2{+}9 \to 1{\cdot}1 \\ \vdots & \vdots & \vdots \\ 9{+}1 \to 1{\cdot}0 & 9{+}2 \to 1{\cdot}1 & \dots \quad 9{+}9 \to 1{\cdot}8 \end{array}$$

$$\begin{array}{lll} 0{+}x \to x & x{+}0 \to x & 0{\cdot}x \to x \\ x{\cdot}(y{\cdot}z) \to (x{+}y){\cdot}z & x{+}(y{\cdot}z) \to y{\cdot}(x{+}z) & (x{\cdot}y){+}z \to x{\cdot}(y{+}z) \end{array}$$

Compute $(3{\cdot}3){+}9$

rewrite sequence

$$\underline{(3{\cdot}3){+}9} \to 3{\cdot}(\underline{3{+}9}) \to \underline{3{\cdot}(1{\cdot}2)} \to \underline{(3{+}1){\cdot}2} \to 4{\cdot}2$$

- $x, y, z, \dots$
- $+, \cdot, 1, 2, \dots, 9$
- 3, $(3{\cdot}3){+}9$, $y{\cdot}z$, x, $x{\cdot}5$, $\dots$
- $5{+}8 \to 1{\cdot}3$
- $(3{+}0){\cdot}y \to_{x+0 \to x} 3{\cdot}y$

variables $\mathcal{V}$
function symbols $\mathcal{F}$
terms $\mathcal{T}(\mathcal{F}, \mathcal{V})$
rewrite rule
rewrite step

Example 1: Decimal Addition

term rewrite system $\mathcal{R}$

$$\begin{array}{lll} 1{+}1 \to 2 & 1{+}2 \to 3 & \dots \quad 1{+}9 \to 1{\cdot}0 \\ 2{+}1 \to 3 & 2{+}2 \to 4 & \dots \quad 2{+}9 \to 1{\cdot}1 \\ \vdots & \vdots & \vdots \\ 9{+}1 \to 1{\cdot}0 & 9{+}2 \to 1{\cdot}1 & \dots \quad 9{+}9 \to 1{\cdot}8 \end{array}$$

$$0{+}x \to x \qquad x{+}0 \to x \qquad 0{\cdot}x \to x$$
$$x{\cdot}(y{\cdot}z) \to (x{+}y){\cdot}z \qquad x{+}(y{\cdot}z) \to y{\cdot}(x{+}z) \qquad (x{\cdot}y){+}z \to x{\cdot}(y{+}z)$$

Compute

rewrite sequence

$$\underline{(3{\cdot}3){+}9} \to 3{\cdot}(\underline{3{+}9}) \to \underline{3{\cdot}(1{\cdot}2)} \to \underline{(3{+}1){\cdot}2} \to 4{\cdot}2$$

- $x, y, z, \dots$
- $+, \cdot, 1, 2, \dots, 9$
- 3, $(3{\cdot}3){+}9$, $y{\cdot}z$, x, $x{\cdot}5$, $\dots$
- $5{+}8 \to 1{\cdot}3$
- $(3{+}0){\cdot}y \to_{x+0\to x} 3{\cdot}y$

variables $\mathcal{V}$
function symbols $\mathcal{F}$
terms $\mathcal{T}(\mathcal{F}, \mathcal{V})$
rewrite rule
rewrite step

Example 2: List Reversal

$$\mathcal{R} = \left\{ \begin{array}{c} [] \mathbin{@} x \to x \\ (x : y) \mathbin{@} z \to x : (y \mathbin{@} z) \end{array} \right. \qquad \begin{array}{r} rev([]) \to [] \\ rev(x : y) \to rev(y) \mathbin{@} (x : []) \\ rev(rev(x)) \to x \end{array}$$

Example 2: List Reversal

$$\mathcal{R} = \left\{ \begin{array}{rl} [] \mathbin{@} x & \to x \\ (x : y) \mathbin{@} z & \to x : (y \mathbin{@} z) \end{array} \right. \qquad \begin{array}{rl} rev([]) & \to [] \\ rev(x : y) & \to rev(y) \mathbin{@} (x : []) \\ rev(rev(x)) & \to x \end{array}$$

Definition

TRS $\mathcal{R}$ terminates if $\exists$ no infinite sequence $t_1 \to_{\mathcal{R}} t_2 \to_{\mathcal{R}} t_3 \to_{\mathcal{R}} \ldots$

Example 2: List Reversal

$$\mathcal{R} = \left\{ \begin{array}{rl} [] \mathbin{@} x & \to x \\ (x : y) \mathbin{@} z & \to x : (y \mathbin{@} z) \end{array} \right. \qquad \begin{array}{rl} rev([]) & \to [] \\ rev(x : y) & \to rev(y) \mathbin{@} (x : []) \\ rev(rev(x)) & \to x \end{array}$$

Definition

undecidable

TRS $\mathcal{R}$ terminates if $\exists$ no infinite sequence $t_1 \to_{\mathcal{R}} t_2 \to_{\mathcal{R}} t_3 \to_{\mathcal{R}} \ldots$

Example 2: List Reversal

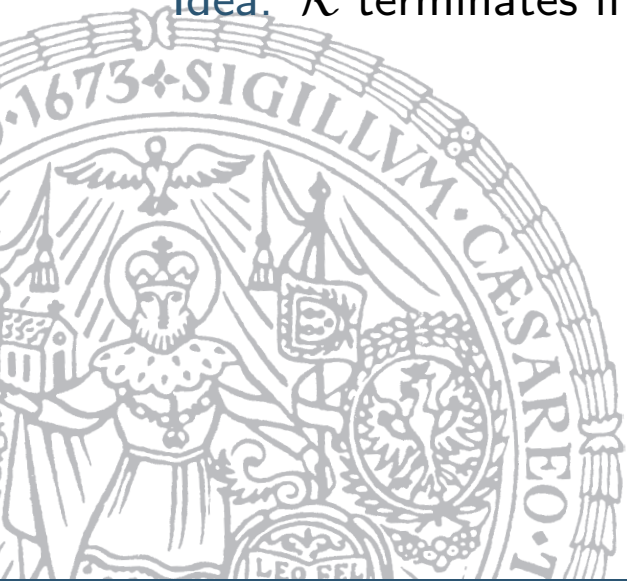
$$\mathcal{R} = \left\{ \begin{array}{rl} [] \mathbin{@} x & \to x \\ (x : y) \mathbin{@} z & \to x : (y \mathbin{@} z) \end{array} \right. \qquad \begin{array}{rl} rev([]) & \to [] \\ rev(x : y) & \to rev(y) \mathbin{@} (x : []) \\ rev(rev(x)) & \to x \end{array}$$

Definition

undecidable

TRS $\mathcal{R}$ terminates if $\exists$ no infinite sequence $t_1 \to_{\mathcal{R}} t_2 \to_{\mathcal{R}} t_3 \to_{\mathcal{R}} \ldots$

- partial methods exist
  Idea: $\mathcal{R}$ terminates if $\exists$ well-founded order $\succ$ such that $s \succ t$ whenever $s \to_{\mathcal{R}} t$



Example 2: List Reversal

$$\mathcal{R} = \left\{ \begin{array}{rl} [] \mathbin{@} x & \to x \\ (x : y) \mathbin{@} z & \to x : (y \mathbin{@} z) \end{array} \right. \qquad \begin{array}{rl} rev([]) & \to [] \\ rev(x : y) & \to rev(y) \mathbin{@} (x : []) \\ rev(rev(x)) & \to x \end{array}$$

Definition

undecidable

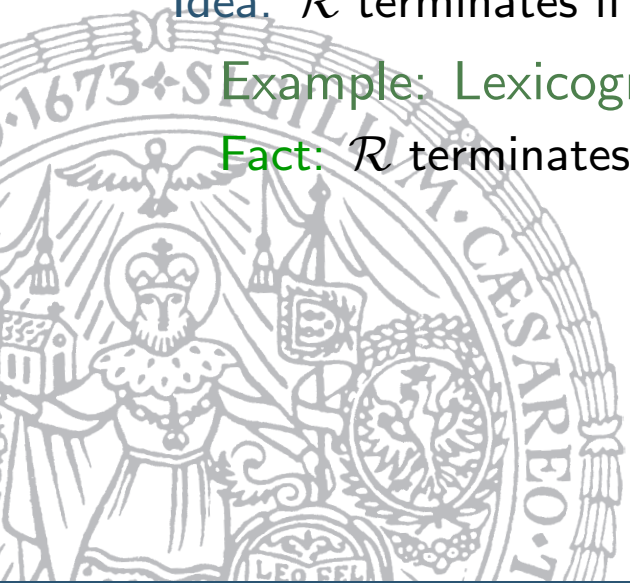
TRS $\mathcal{R}$ terminates if $\exists$ no infinite sequence $t_1 \to_{\mathcal{R}} t_2 \to_{\mathcal{R}} t_3 \to_{\mathcal{R}} \ldots$

- partial methods exist

  Idea: $\mathcal{R}$ terminates if $\exists$ well-founded order $\succ$ such that $s \succ t$ whenever $s \to_{\mathcal{R}} t$

  Example: Lexicographic Path Order — reduction order

  Fact: $\mathcal{R}$ terminates if $l \succ_{lpo} r$ for all rules $l \to r$ in $\mathcal{R}$.



Example 2: List Reversal

$rev > @ > : > []$

$$\mathcal{R} = \left\{ \begin{array}{rl} [] \mathbin{@} x & \to x \\ (x : y) \mathbin{@} z & \to x : (y \mathbin{@} z) \end{array} \right. \qquad \begin{array}{rl} rev([]) & \to [] \\ rev(x : y) & \to rev(y) \mathbin{@} (x : []) \\ rev(rev(x)) & \to x \end{array}$$

Definition

undecidable

TRS $\mathcal{R}$ terminates if $\exists$ no infinite sequence $t_1 \to_{\mathcal{R}} t_2 \to_{\mathcal{R}} t_3 \to_{\mathcal{R}} \ldots$

- partial methods exist

  Idea: $\mathcal{R}$ terminates if $\exists$ well-founded order $\succ$ such that $s \succ t$ whenever $s \to_{\mathcal{R}} t$

  Example: Lexicographic Path Order — reduction order

  Fact: $\mathcal{R}$ terminates if $l \succ_{lpo} r$ for all rules $l \to r$ in $\mathcal{R}$.

  for $>$ ordering on function symbols, $s \succ_{lpo} t$ if $s = f(s_1, \ldots, s_n)$ and

  - $s_i \succeq_{lpo} t$ for some i, or
  - $t = g(t_1, \ldots, t_m)$, $f > g$ and $s \succ_{lpo} t_i$ for $1 \leq i \leq m$, or
  - $t = f(t_1, \ldots, t_n)$, $s \succ_{lpo} t_i$ for $1 \leq i \leq n$ and $(s_1, \ldots, s_n) \succ_{lpo}^{lex} (t_1, \ldots, t_n)$

Example 2: List Reversal

$rev > @ > : > []$

$$\mathcal{R} = \left\{ \begin{array}{rcl} [] \mathbin{@} x & \to & x \\ (x : y) \mathbin{@} z & \to & x : (y \mathbin{@} z) \end{array} \right. \quad \checkmark \qquad \left\{ \begin{array}{rcl} rev([]) & \to & [] \\ rev(x : y) & \to & rev(y) \mathbin{@} (x : []) \\ rev(rev(x)) & \to & x \end{array} \right. \quad \checkmark \quad \checkmark$$

Definition

undecidable

TRS $\mathcal{R}$ terminates if $\exists$ no infinite sequence $t_1 \to_{\mathcal{R}} t_2 \to_{\mathcal{R}} t_3 \to_{\mathcal{R}} \ldots$

- partial methods exist

  Idea: $\mathcal{R}$ terminates if $\exists$ well-founded order $\succ$ such that $s \succ t$ whenever $s \to_{\mathcal{R}} t$

  Example: Lexicographic Path Order — reduction order

  Fact: $\mathcal{R}$ terminates if $l \succ_{lpo} r$ for all rules $l \to r$ in $\mathcal{R}$.

  for $>$ ordering on function symbols, $s \succ_{lpo} t$ if $s = f(s_1, \ldots, s_n)$ and

  - $s_i \succeq_{lpo} t$ for some i, or
  - $t = g(t_1, \ldots, t_m)$, $f > g$ and $s \succ_{lpo} t_i$ for $1 \leq i \leq m$, or
  - $t = f(t_1, \ldots, t_n)$, $s \succ_{lpo} t_i$ for $1 \leq i \leq n$ and $(s_1, \ldots, s_n) \succ_{lpo}^{lex} (t_1, \ldots, t_n)$

Example 2: List Reversal

$rev > @ > : > []$

$$\mathcal{R} = \left\{ \begin{array}{rcl} [] \mathbin{@} x & \to & x \\ (x : y) \mathbin{@} z & \to & x : (y \mathbin{@} z) \end{array} \right. \quad \checkmark \qquad \left\{ \begin{array}{rcl} rev([]) & \to & [] \\ rev(x : y) & \to & rev(y) \mathbin{@} (x : []) \\ rev(rev(x)) & \to & x \end{array} \right. \quad \checkmark \quad \checkmark$$

Definition

undecidable

TRS $\mathcal{R}$ terminates if $\exists$ no infinite sequence $t_1 \to_{\mathcal{R}} t_2 \to_{\mathcal{R}} t_3 \to_{\mathcal{R}} \ldots$

- partial methods exist

  Idea: $\mathcal{R}$ terminates if $\exists$ well-founded order $\succ$ such that $s \succ t$ whenever $s \to_{\mathcal{R}} t$

  Example: Lexicographic Path Order — reduction order

  Fact: $\mathcal{R}$ terminates if $l \succ_{lpo} r$ for all rules $l \to r$ in $\mathcal{R}$.

  for $>$ ordering on function symbols, $s \succ_{lpo} t$ if $s = f(s_1, \ldots, s_n)$ and

  - $s_i \succeq_{lpo} t$ for some i, or
  - $t = g(t_1, \ldots, t_m)$, $f > g$ and $s \succ_{lpo} t_i$ for $1 \leq i \leq m$, or
  - $t = f(t_1, \ldots, t_n)$, $s \succ_{lpo} t_i$ for $1 \leq i \leq n$ and $(s_1, \ldots, s_n) \succ_{lpo}^{lex} (t_1, \ldots, t_n)$

Example 2: List Reversal

$rev > @ > : > []$

$$\mathcal{R} = \left\{ \begin{array}{rcl} [] \mathbin{@} x & \to & x \quad \checkmark \\ (x : y) \mathbin{@} z & \to & x : (y \mathbin{@} z) \quad \checkmark \end{array} \right. \qquad \left\{ \begin{array}{rcl} rev([]) & \to & [] \quad \checkmark \\ rev(x : y) & \to & rev(y) \mathbin{@} (x : []) \\ rev(rev(x)) & \to & x \quad \checkmark \end{array} \right.$$

Definition

undecidable

TRS $\mathcal{R}$ terminates if $\exists$ no infinite sequence $t_1 \to_{\mathcal{R}} t_2 \to_{\mathcal{R}} t_3 \to_{\mathcal{R}} \ldots$

- partial methods exist

  Idea: $\mathcal{R}$ terminates if $\exists$ well-founded order $\succ$ such that $s \succ t$ whenever $s \to_{\mathcal{R}} t$

  Example: Lexicographic Path Order — reduction order

  Fact: $\mathcal{R}$ terminates if $l \succ_{lpo} r$ for all rules $l \to r$ in $\mathcal{R}$.

  for $>$ ordering on function symbols, $s \succ_{lpo} t$ if $s = f(s_1, \ldots, s_n)$ and

  - $s_i \succeq_{lpo} t$ for some i, or
  - $t = g(t_1, \ldots, t_m)$, $f > g$ and $s \succ_{lpo} t_i$ for $1 \leq i \leq m$, or
  - $t = f(t_1, \ldots, t_n)$, $s \succ_{lpo} t_i$ for $1 \leq i \leq n$ and $(s_1, \ldots, s_n) \succ_{lpo}^{lex} (t_1, \ldots, t_n)$

Example 2: List Reversal

$rev > @ > : > []$

$$\mathcal{R} = \left\{ \begin{array}{rcl} [] \mathbin{@} x & \to & x \quad \checkmark \\ (x : y) \mathbin{@} z & \to & x : (y \mathbin{@} z) \quad \checkmark \end{array} \right. \qquad \left\{ \begin{array}{rcl} rev([]) & \to & [] \quad \checkmark \\ rev(x : y) & \to & rev(y) \mathbin{@} (x : []) \quad \checkmark \\ rev(rev(x)) & \to & x \quad \checkmark \end{array} \right.$$

Definition

undecidable

TRS $\mathcal{R}$ terminates if $\exists$ no infinite sequence $t_1 \to_\mathcal{R} t_2 \to_\mathcal{R} t_3 \to_\mathcal{R} \ldots$

- partial methods exist

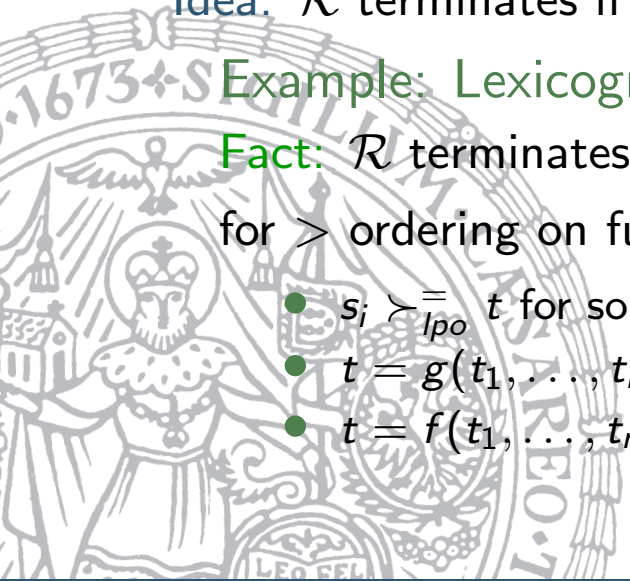
  Idea: $\mathcal{R}$ terminates if $\exists$ well-founded order $\succ$ such that $s \succ t$ whenever $s \to_\mathcal{R} t$

  Example: Lexicographic Path Order — reduction order

  Fact: $\mathcal{R}$ terminates if $l \succ_{lpo} r$ for all rules $l \to r$ in $\mathcal{R}$.

  for $>$ ordering on function symbols, $s \succ_{lpo} t$ if $s = f(s_1, \ldots, s_n)$ and

  - $s_i \succeq_{lpo} t$ for some i, or
  - $t = g(t_1, \ldots, t_m)$, $f > g$ and $s \succ_{lpo} t_i$ for $1 \le i \le m$, or
  - $t = f(t_1, \ldots, t_n)$, $s \succ_{lpo} t_i$ for $1 \le i \le n$ and $(s_1, \ldots, s_n) \succ_{lpo}^{lex} (t_1, \ldots, t_n)$



Example 2: List Reversal

$rev > @ > : > []$

$$\mathcal{R} = \left\{ \begin{array}{rcl} [] \mathbin{@} x & \to & x \quad \checkmark \\ (x : y) \mathbin{@} z & \to & x : (y \mathbin{@} z) \quad \checkmark \end{array} \right. \qquad \left\{ \begin{array}{rcl} rev([]) & \to & [] \quad \checkmark \\ rev(x : y) & \to & rev(y) \mathbin{@} (x : []) \quad \checkmark \\ rev(rev(x)) & \to & x \quad \checkmark \end{array} \right.$$

Definition

undecidable

TRS $\mathcal{R}$ terminates if $\exists$ no infinite sequence $t_1 \to_\mathcal{R} t_2 \to_\mathcal{R} t_3 \to_\mathcal{R} \ldots$

- partial methods exist
  Idea: $\mathcal{R}$ terminates if $\exists$ well-founded order $\succ$ such that $s \succ t$ whenever $s \to_\mathcal{R} t$

  Example: Lexicographic Path Order — reduction order

  Fact: $\mathcal{R}$ terminates if $l \succ_{lpo} r$ for all rules $l \to r$ in $\mathcal{R}$.

  for $>$ ordering on function symbols, $s \succ_{lpo} t$ if $s = f(s_1, \ldots, s_n)$ and
  - $s_i \succeq_{lpo} t$ for some i, or
  - $t = g(t_1, \ldots, t_m)$, $f > g$ and $s \succ_{lpo} t_i$ for $1 \le i \le m$, or
  - $t = f(t_1, \ldots, t_n)$, $s \succ_{lpo} t_i$ for $1 \le i \le n$ and $(s_1, \ldots, s_n) \succ_{lpo}^{lex} (t_1, \ldots, t_n)$
- automatic termination tools: AProVE, Jambox, TORPA, NTI, T_TT_2, ...

Example 3: Insertion Sort

$$\mathcal{R} = \left\{ \begin{array}{rcl} sort([]) & \to & [] \\ sort(x{:}y) & \to & insert(x, sort(y)) \\ insert(x, []) & \to & x{:}[] \\ insert(x, y{:}z) & \to & c(x, y{:}z, x, y) \end{array} \right. \qquad \left\{ \begin{array}{rcl} c(x, y{:}z, w, 0) & \to & x{:}(y{:}z) \\ c(x, y{:}z, 0, w) & \to & y{:}insert(x, z) \\ c(x, y{:}z, s(v), s(w)) & \to & c(x, y{:}z, v, w) \end{array} \right.$$

Example 3: Insertion Sort

$$\mathcal{R} = \left\{ \begin{array}{rl} sort([]) \to [] & \qquad c(x, y{:}z, w, 0) \to x{:}(y{:}z) \\ sort(x{:}y) \to insert(x, sort(y)) & \qquad c(x, y{:}z, 0, w) \to y{:}insert(x, z) \\ insert(x, []) \to x{:}[] & \qquad c(x, y{:}z, s(v), s(w)) \to c(x, y{:}z, v, w) \\ insert(x, y{:}z) \to c(x, y{:}z, x, y) & \end{array} \right.$$

Definition

0 or more steps using $\mathcal{R}$

TRS $\mathcal{R}$ is confluent if $\forall\, s \leftarrow^*_{\mathcal{R}} u \to^*_{\mathcal{R}} t \quad \exists v \quad s \to^*_{\mathcal{R}} v \leftarrow^*_{\mathcal{R}} t$

Example 3: Insertion Sort

$$\mathcal{R} = \left\{ \begin{array}{rl} sort([]) & \to [] \\ sort(x{:}y) & \to insert(x, sort(y)) \\ insert(x, []) & \to x{:}[] \\ insert(x, y{:}z) & \to c(x, y{:}z, x, y) \end{array} \right. \quad \left. \begin{array}{rl} c(x, y{:}z, w, 0) & \to x{:}(y{:}z) \\ c(x, y{:}z, 0, w) & \to y{:}insert(x, z) \\ c(x, y{:}z, s(v), s(w)) & \to c(x, y{:}z, v, w) \end{array} \right.$$

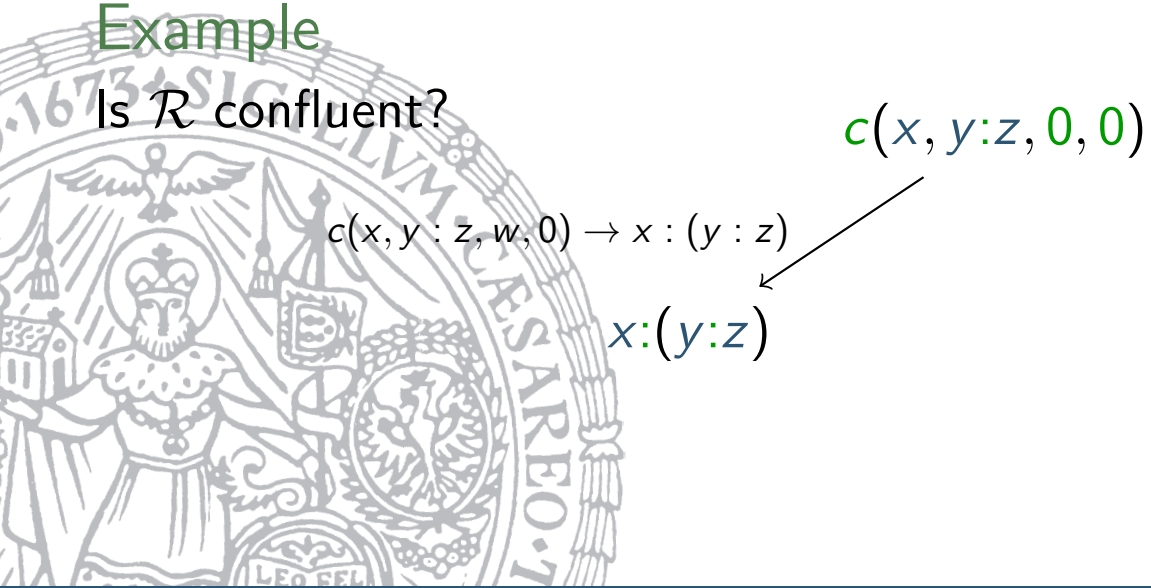
Definition

0 or more steps using $\mathcal{R}$

TRS $\mathcal{R}$ is confluent if $\forall\, s \leftarrow^*_{\mathcal{R}} u \rightarrow^*_{\mathcal{R}} t \quad \exists v \quad s \rightarrow^*_{\mathcal{R}} v \leftarrow^*_{\mathcal{R}} t$

Example

Is $\mathcal{R}$ confluent?



Example 3: Insertion Sort

$$\mathcal{R} = \left\{ \begin{array}{rl} sort([]) & \to [] \\ sort(x{:}y) & \to insert(x, sort(y)) \\ insert(x, []) & \to x{:}[] \\ insert(x, y{:}z) & \to c(x, y{:}z, x, y) \end{array} \right. \quad \left. \begin{array}{rl} c(x, y{:}z, w, 0) & \to x{:}(y{:}z) \\ c(x, y{:}z, 0, w) & \to y{:}insert(x, z) \\ c(x, y{:}z, s(v), s(w)) & \to c(x, y{:}z, v, w) \end{array} \right.$$

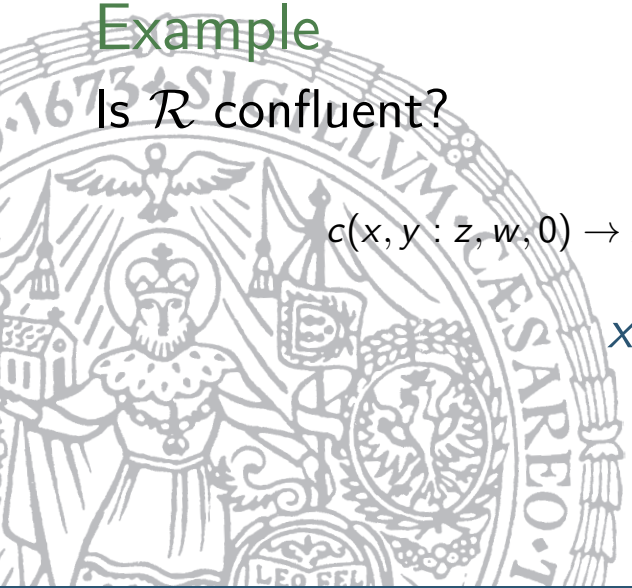
Definition

0 or more steps using $\mathcal{R}$

TRS $\mathcal{R}$ is confluent if $\forall\, s \leftarrow^*_{\mathcal{R}} u \rightarrow^*_{\mathcal{R}} t \quad \exists v \quad s \rightarrow^*_{\mathcal{R}} v \leftarrow^*_{\mathcal{R}} t$

Example

Is $\mathcal{R}$ confluent?



$c(x, y:z, 0, 0)$

$c(x, y:z, w, 0) \to x:(y:z)$ $\qquad$ $c(x, y:z, 0, w) \to y:insert(x, z)$

$x:(y:z)$ $\qquad$ $y:insert(x, z)$

Example 3: Insertion Sort

$$\mathcal{R} = \left\{ \begin{array}{rl} sort([]) & \to [] \\ sort(x{:}y) & \to insert(x, sort(y)) \\ insert(x, []) & \to x{:}[] \\ insert(x, y{:}z) & \to c(x, y{:}z, x, y) \end{array} \right. \qquad \begin{array}{rl} c(x, y{:}z, w, 0) & \to x{:}(y{:}z) \\ c(x, y{:}z, 0, w) & \to y{:}insert(x, z) \\ c(x, y{:}z, s(v), s(w)) & \to c(x, y{:}z, v, w) \end{array}$$

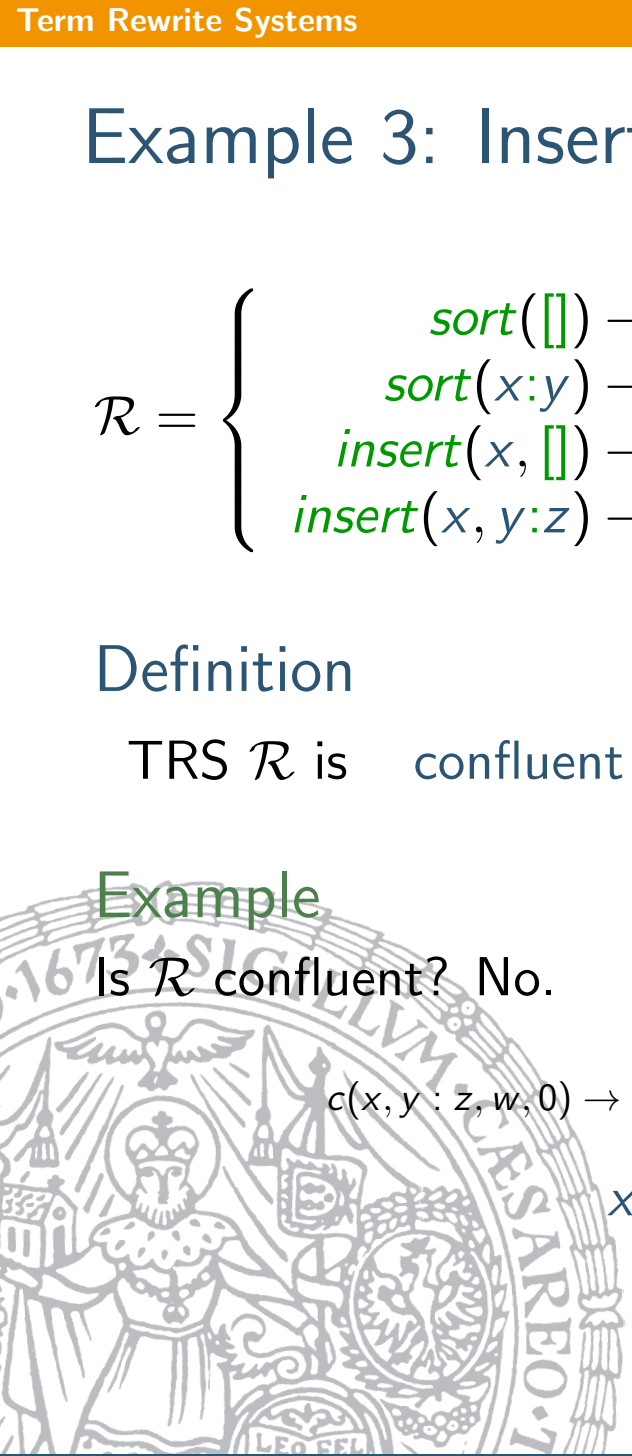
Definition

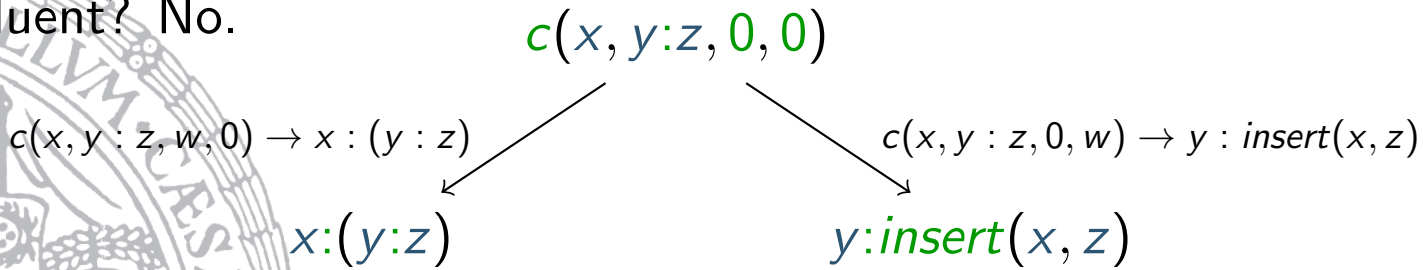
TRS $\mathcal{R}$ is confluent if $\forall\, s \leftarrow^*_{\mathcal{R}} u \rightarrow^*_{\mathcal{R}} t \quad \exists v \quad s \rightarrow^*_{\mathcal{R}} v \leftarrow^*_{\mathcal{R}} t$

0 or more steps using $\mathcal{R}$

Example

Is $\mathcal{R}$ confluent? No.





Example 3: Insertion Sort

$$\mathcal{R} = \left\{ \begin{array}{rl} sort([]) & \to [] \\ sort(x{:}y) & \to insert(x, sort(y)) \\ insert(x, []) & \to x{:}[] \\ insert(x, y{:}z) & \to c(x, y{:}z, x, y) \end{array} \right. \qquad \begin{array}{rl} c(x, y{:}z, w, 0) & \to x{:}(y{:}z) \\ c(x, y{:}z, 0, w) & \to y{:}insert(x, z) \\ c(x, y{:}z, s(v), s(w)) & \to c(x, y{:}z, v, w) \end{array}$$

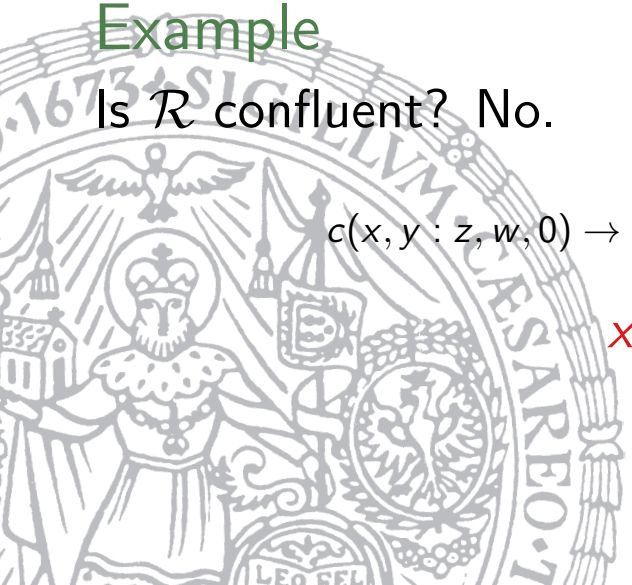
Definition

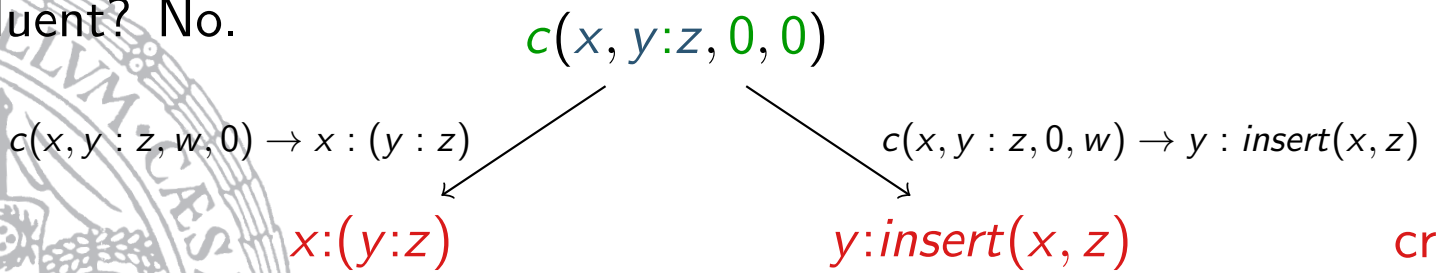
TRS $\mathcal{R}$ is confluent if $\forall\, s \leftarrow^*_{\mathcal{R}} u \rightarrow^*_{\mathcal{R}} t \quad \exists v \quad s \rightarrow^*_{\mathcal{R}} v \leftarrow^*_{\mathcal{R}} t$

0 or more steps using $\mathcal{R}$

Example

Is $\mathcal{R}$ confluent? No.





critical pair

Example 3: Insertion Sort

$$\mathcal{R} = \left\{ \begin{array}{rl} sort([]) & \to [] \\ sort(x{:}y) & \to insert(x, sort(y)) \\ insert(x, []) & \to x{:}[] \\ insert(x, y{:}z) & \to c(x, y{:}z, x, y) \end{array} \right. \qquad \begin{array}{rl} c(x, y{:}z, w, 0) & \to x{:}(y{:}z) \\ c(x, y{:}z, 0, w) & \to y{:}insert(x, z) \\ c(x, y{:}z, s(v), s(w)) & \to c(x, y{:}z, v, w) \end{array}$$

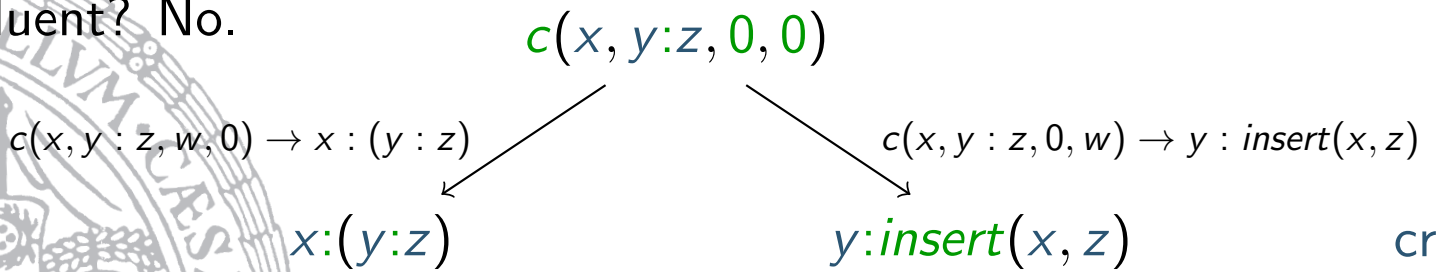
Definition

TRS $\mathcal{R}$ is confluent if $\forall\, s \leftarrow^*_{\mathcal{R}} u \rightarrow^*_{\mathcal{R}} t \quad \exists v \quad s \rightarrow^*_{\mathcal{R}} v \leftarrow^*_{\mathcal{R}} t$

(0 or more steps using $\mathcal{R}$)

Example

Is $\mathcal{R}$ confluent? No.



critical pair

Fact

terminating TRS is confluent if and only if all critical pairs are joinable

Outline

Knuth-Bendix Completion

$$\underset{\text{reduction ordering}}{\succ} + \underset{\text{equations}}{\mathcal{E}} \longrightarrow_{KB} \underset{\text{rewrite system}}{\mathcal{R}}$$

$\mathcal{R}$ is confluent, terminating, reduced and $\approx_{\mathcal{E}} = \leftrightarrow^*_{\mathcal{R}}$

$\leftrightarrow^*_{\mathcal{R}} = \rightarrow^*_{\mathcal{R}} \cup \leftarrow^*_{\mathcal{R}}$

Knuth-Bendix Completion

$$\underset{\text{reduction ordering}}{\succ} + \underset{\text{equations}}{\mathcal{E}} \longrightarrow_{KB} \underset{\text{rewrite system}}{\mathcal{R}}$$

$\mathcal{R}$ is confluent, terminating, reduced and $\approx_{\mathcal{E}} = \leftrightarrow^*_{\mathcal{R}}$

$\leftrightarrow^*_{\mathcal{R}} = \rightarrow^*_{\mathcal{R}} \cup \leftarrow^*_{\mathcal{R}}$

Example (Group Theory)

$$\mathcal{E} \qquad \begin{aligned} \mathsf{e} \cdot x &\approx x \\ x^{-} \cdot x &\approx \mathsf{e} \\ (x \cdot y) \cdot z &\approx x \cdot (y \cdot z) \end{aligned}$$

Knuth-Bendix Completion

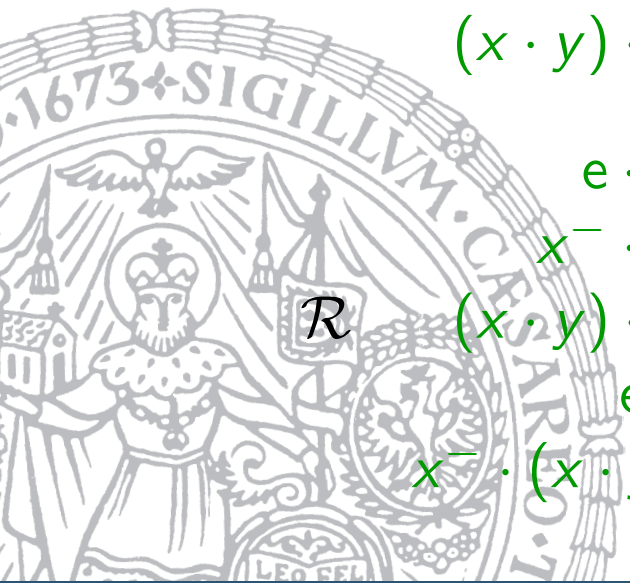
$$\underset{\text{reduction ordering}}{\succ} + \underset{\text{equations}}{\mathcal{E}} \longrightarrow_{KB} \underset{\text{rewrite system}}{\mathcal{R}}$$

$\mathcal{R}$ is confluent, terminating, reduced and $\approx_{\mathcal{E}} = \leftrightarrow^*_{\mathcal{R}}$

$\leftrightarrow^*_{\mathcal{R}} = \rightarrow^*_{\mathcal{R}} \cup \leftarrow^*_{\mathcal{R}}$

Example (Group Theory)

$$\mathcal{E}\quad \begin{aligned} \mathsf{e} \cdot x &\approx x \\ x^- \cdot x &\approx \mathsf{e} \\ (x \cdot y) \cdot z &\approx x \cdot (y \cdot z) \end{aligned}$$



$$\mathcal{R}\quad \begin{aligned} \mathsf{e} \cdot x &\to x & x \cdot \mathsf{e} &\to x \\ x^- \cdot x &\to \mathsf{e} & x \cdot x^- &\to \mathsf{e} \\ (x \cdot y) \cdot z &\to x \cdot (y \cdot z) & x^{--} &\to x \\ \mathsf{e}^- &\to \mathsf{e} & (x \cdot y)^- &\to y^- \cdot x^- \\ x^- \cdot (x \cdot y) &\to y & x \cdot (x^- \cdot y) &\to y \end{aligned}$$

Knuth-Bendix Completion

$$\underset{\text{reduction ordering}}{\succ} + \underset{\text{equations}}{\mathcal{E}} \longrightarrow_{KB} \underset{\text{rewrite system}}{\mathcal{R}}$$

$\mathcal{R}$ is confluent, terminating, reduced and $\approx_{\mathcal{E}} = \leftrightarrow^*_{\mathcal{R}}$

$\leftrightarrow^*_{\mathcal{R}} = \rightarrow^*_{\mathcal{R}} \cup \leftarrow^*_{\mathcal{R}}$

Example (Group Theory)

$$(x^- \cdot x)^- \cdot (e \cdot (y \cdot e)) \stackrel{?}{\approx} y \cdot e$$

liza.jpg

$$\mathcal{E} \quad \begin{aligned} e \cdot x &\approx x \\ x^- \cdot x &\approx e \\ (x \cdot y) \cdot z &\approx x \cdot (y \cdot z) \end{aligned}$$

$$\mathcal{R} \quad \begin{aligned} e \cdot x &\to x & x \cdot e &\to x \\ x^- \cdot x &\to e & x \cdot x^- &\to e \\ (x \cdot y) \cdot z &\to x \cdot (y \cdot z) & x^{--} &\to x \\ e^- &\to e & (x \cdot y)^- &\to y^- \cdot x^- \\ x^- \cdot (x \cdot y) &\to y & x \cdot (x^- \cdot y) &\to y \end{aligned}$$

Knuth-Bendix Completion

$$\underset{\text{reduction ordering}}{\succ} + \underset{\text{equations}}{\mathcal{E}} \longrightarrow_{KB} \underset{\text{rewrite system}}{\mathcal{R}}$$

$\mathcal{R}$ is confluent, terminating, reduced and $\approx_{\mathcal{E}} = \leftrightarrow^*_{\mathcal{R}}$

$\leftrightarrow^*_{\mathcal{R}} = \rightarrow^*_{\mathcal{R}} \cup \leftarrow^*_{\mathcal{R}}$

Example (Group Theory)

$$y \leftarrow^!_{\mathcal{R}} (x^- \cdot x)^- \cdot (\mathsf{e} \cdot (y \cdot \mathsf{e})) \approx y \cdot \mathsf{e} \rightarrow^!_{\mathcal{R}} y$$

liza.jpg

$$\mathcal{E} \quad \begin{aligned} \mathsf{e} \cdot x &\approx x \\ x^- \cdot x &\approx \mathsf{e} \\ (x \cdot y) \cdot z &\approx x \cdot (y \cdot z) \end{aligned}$$

$$\mathcal{R} \quad \begin{aligned} \mathsf{e} \cdot x &\to x & x \cdot \mathsf{e} &\to x \\ x^- \cdot x &\to \mathsf{e} & x \cdot x^- &\to \mathsf{e} \\ (x \cdot y) \cdot z &\to x \cdot (y \cdot z) & x^{--} &\to x \\ \mathsf{e}^- &\to \mathsf{e} & (x \cdot y)^- &\to y^- \cdot x^- \\ x^- \cdot (x \cdot y) &\to y & x \cdot (x^- \cdot y) &\to y \end{aligned}$$

KB: Standard Completion

Definition (KB)

$\mathcal{E}$: set of equations $\mathcal{R}$: set of rewrite rules $\succ$: reduction order

inference system KB consists of six rules:

KB: Standard Completion

Definition (KB)

$\mathcal{E}$: set of equations $\mathcal{R}$: set of rewrite rules $\succ$: reduction order

inference system KB consists of six rules:

orient $\dfrac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{}$

if $s \succ t$

KB: Standard Completion

Definition (KB)

$\mathcal{E}$: set of equations $\mathcal{R}$: set of rewrite rules $\succ$: reduction order

inference system KB consists of six rules:

orient
$$\frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{\mathcal{E}, \mathcal{R} \cup \{s \to t\}} \quad \text{if } s \succ t$$

KB: Standard Completion

Definition (KB)

$\mathcal{E}$: set of equations $\mathcal{R}$: set of rewrite rules $\succ$: reduction order

inference system KB consists of six rules:

orient $$\frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{\mathcal{E}, \mathcal{R} \cup \{s \to t\}} \quad \text{if } s \succ t$$

deduce $$\frac{\mathcal{E}, \mathcal{R}}{} \quad \text{if } s \leftarrow_{\mathcal{R}} u \to_{\mathcal{R}} t$$

KB: Standard Completion

Definition (KB)

$\mathcal{E}$: set of equations $\mathcal{R}$: set of rewrite rules $\succ$: reduction order

inference system KB consists of six rules:

orient $$\frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{\mathcal{E}, \mathcal{R} \cup \{s \to t\}} \text{ if } s \succ t$$

deduce $$\frac{\mathcal{E}, \mathcal{R}}{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}} \text{ if } s \leftarrow_{\mathcal{R}} u \to_{\mathcal{R}} t$$

KB: Standard Completion

Definition (KB)

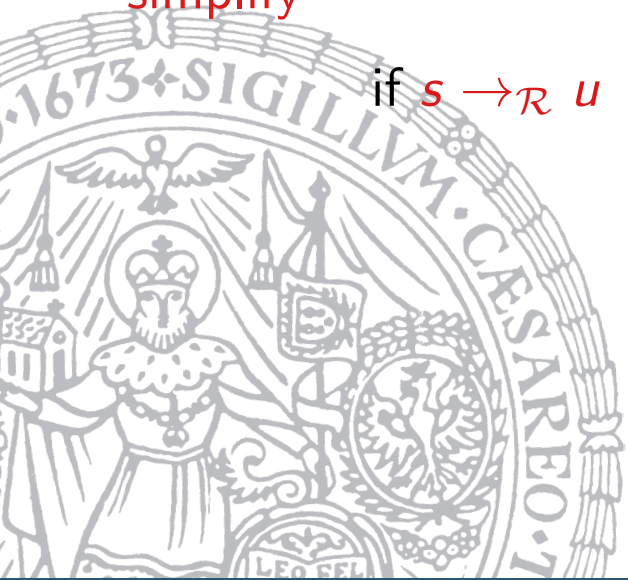
$\mathcal{E}$: set of equations $\mathcal{R}$: set of rewrite rules $\succ$: reduction order

inference system KB consists of six rules:

orient
$$\frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{\mathcal{E}, \mathcal{R} \cup \{s \to t\}} \quad \text{if } s \succ t$$

deduce
$$\frac{\mathcal{E}, \mathcal{R}}{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}} \quad \text{if } s \leftarrow_{\mathcal{R}} u \to_{\mathcal{R}} t$$

simplify
$$\frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{\quad} \quad \text{if } s \to_{\mathcal{R}} u$$



KB: Standard Completion

Definition (KB)

$\mathcal{E}$: set of equations $\mathcal{R}$: set of rewrite rules $\succ$: reduction order

inference system KB consists of six rules:

orient
$$\frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{\mathcal{E}, \mathcal{R} \cup \{s \to t\}} \quad \text{if } s \succ t$$

deduce
$$\frac{\mathcal{E}, \mathcal{R}}{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}} \quad \text{if } s \leftarrow_{\mathcal{R}} u \to_{\mathcal{R}} t$$

simplify
$$\frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{\mathcal{E} \cup \{u \approx t\}, \mathcal{R}} \quad \text{if } s \to_{\mathcal{R}} u$$

KB: Standard Completion

Definition (KB)

$\mathcal{E}$: set of equations $\mathcal{R}$: set of rewrite rules $\succ$: reduction order

inference system KB consists of six rules:

orient
$$\frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{\mathcal{E}, \mathcal{R} \cup \{s \to t\}} \quad \text{if } s \succ t$$

deduce
$$\frac{\mathcal{E}, \mathcal{R}}{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}} \quad \text{if } s \leftarrow_{\mathcal{R}} u \to_{\mathcal{R}} t$$

simplify
$$\frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{\mathcal{E} \cup \{u \approx t\}, \mathcal{R}} \quad \text{if } s \to_{\mathcal{R}} u$$

compose
$$\frac{\mathcal{E}, \mathcal{R} \cup \{s \to t\}}{} \quad \text{if } t \to_{\mathcal{R}} u$$

KB: Standard Completion

Definition (KB)

$\mathcal{E}$: set of equations $\mathcal{R}$: set of rewrite rules $\succ$: reduction order

inference system KB consists of six rules:

orient
$$\frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{\mathcal{E}, \mathcal{R} \cup \{s \to t\}} \quad \text{if } s \succ t$$

deduce
$$\frac{\mathcal{E}, \mathcal{R}}{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}} \quad \text{if } s \leftarrow_{\mathcal{R}} u \to_{\mathcal{R}} t$$

simplify
$$\frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{\mathcal{E} \cup \{u \approx t\}, \mathcal{R}} \quad \text{if } s \to_{\mathcal{R}} u$$

compose
$$\frac{\mathcal{E}, \mathcal{R} \cup \{s \to t\}}{\mathcal{E}, \mathcal{R} \cup \{s \to u\}} \quad \text{if } t \to_{\mathcal{R}} u$$

KB: Standard Completion

Definition (KB)

$\mathcal{E}$: set of equations $\mathcal{R}$: set of rewrite rules $\succ$: reduction order

inference system KB consists of six rules:

orient
$$\frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{\mathcal{E}, \mathcal{R} \cup \{s \to t\}} \quad \text{if } s \succ t$$

deduce
$$\frac{\mathcal{E}, \mathcal{R}}{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}} \quad \text{if } s \leftarrow_{\mathcal{R}} u \to_{\mathcal{R}} t$$

simplify
$$\frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{\mathcal{E} \cup \{u \approx t\}, \mathcal{R}} \quad \text{if } s \to_{\mathcal{R}} u$$

compose
$$\frac{\mathcal{E}, \mathcal{R} \cup \{s \to t\}}{\mathcal{E}, \mathcal{R} \cup \{s \to u\}} \quad \text{if } t \to_{\mathcal{R}} u$$

collapse
$$\frac{}{\mathcal{E}, \mathcal{R} \cup \{s \to t\}} \quad \text{if } s \to u$$

KB: Standard Completion

Definition (KB)

$\mathcal{E}$: set of equations $\mathcal{R}$: set of rewrite rules $\succ$: reduction order

inference system KB consists of six rules:

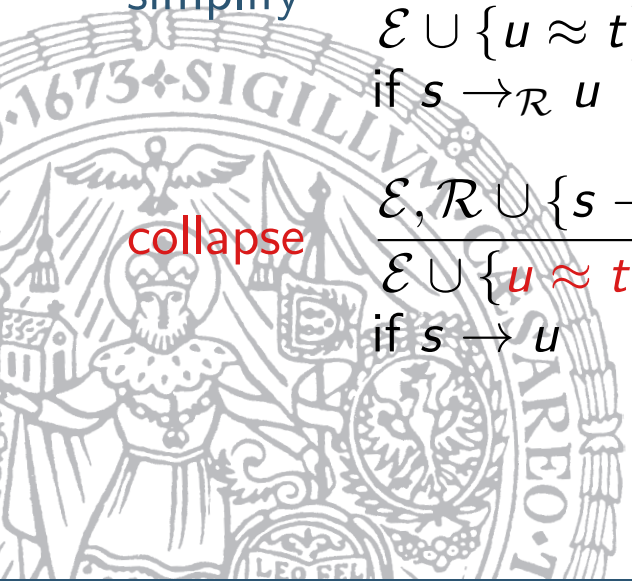
orient
$$\frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{\mathcal{E}, \mathcal{R} \cup \{s \to t\}} \quad \text{if } s \succ t$$

deduce
$$\frac{\mathcal{E}, \mathcal{R}}{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}} \quad \text{if } s \leftarrow_{\mathcal{R}} u \to_{\mathcal{R}} t$$

simplify
$$\frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{\mathcal{E} \cup \{u \approx t\}, \mathcal{R}} \quad \text{if } s \to_{\mathcal{R}} u$$

compose
$$\frac{\mathcal{E}, \mathcal{R} \cup \{s \to t\}}{\mathcal{E}, \mathcal{R} \cup \{s \to u\}} \quad \text{if } t \to_{\mathcal{R}} u$$

collapse
$$\frac{\mathcal{E}, \mathcal{R} \cup \{s \to t\}}{\mathcal{E} \cup \{u \approx t\}, \mathcal{R}} \quad \text{if } s \to u$$



KB: Standard Completion

Definition (KB)

$\mathcal{E}$: set of equations $\mathcal{R}$: set of rewrite rules $\succ$: reduction order

inference system KB consists of six rules:

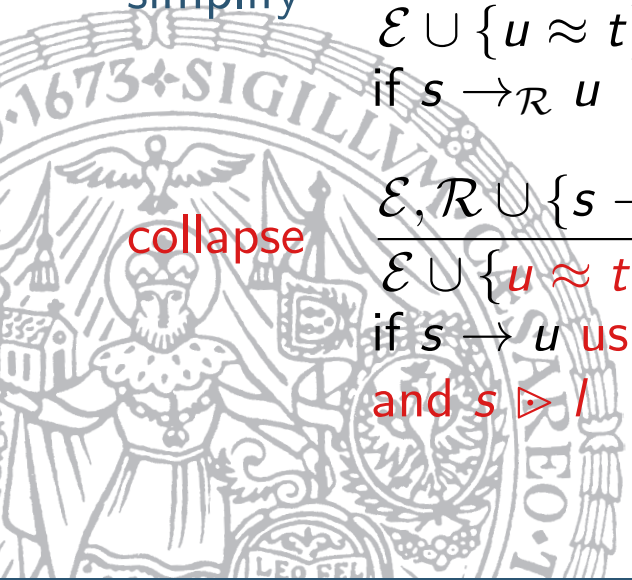
orient
$$\frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{\mathcal{E}, \mathcal{R} \cup \{s \to t\}} \quad \text{if } s \succ t$$

deduce
$$\frac{\mathcal{E}, \mathcal{R}}{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}} \quad \text{if } s \leftarrow_{\mathcal{R}} u \to_{\mathcal{R}} t$$

simplify
$$\frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{\mathcal{E} \cup \{u \approx t\}, \mathcal{R}} \quad \text{if } s \to_{\mathcal{R}} u$$

compose
$$\frac{\mathcal{E}, \mathcal{R} \cup \{s \to t\}}{\mathcal{E}, \mathcal{R} \cup \{s \to u\}} \quad \text{if } t \to_{\mathcal{R}} u$$

collapse
$$\frac{\mathcal{E}, \mathcal{R} \cup \{s \to t\}}{\mathcal{E} \cup \{u \approx t\}, \mathcal{R}} \quad \text{if } s \to u \text{ using } l \to r \text{ and } s \rhd l$$



KB: Standard Completion

Definition (KB)

$\mathcal{E}$: set of equations $\mathcal{R}$: set of rewrite rules $\succ$: reduction order

inference system KB consists of six rules:

orient
$$\frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{\mathcal{E}, \mathcal{R} \cup \{s \to t\}} \quad \text{if } s \succ t$$

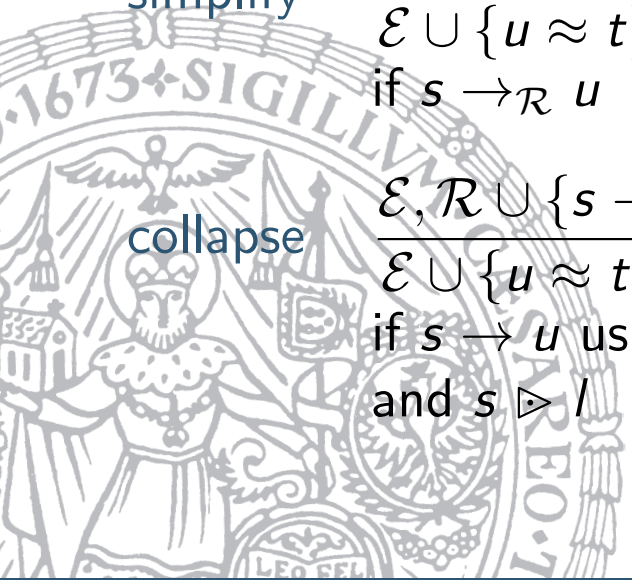
deduce
$$\frac{\mathcal{E}, \mathcal{R}}{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}} \quad \text{if } s \leftarrow_{\mathcal{R}} u \to_{\mathcal{R}} t$$

simplify
$$\frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{\mathcal{E} \cup \{u \approx t\}, \mathcal{R}} \quad \text{if } s \to_{\mathcal{R}} u$$

compose
$$\frac{\mathcal{E}, \mathcal{R} \cup \{s \to t\}}{\mathcal{E}, \mathcal{R} \cup \{s \to u\}} \quad \text{if } t \to_{\mathcal{R}} u$$

collapse
$$\frac{\mathcal{E}, \mathcal{R} \cup \{s \to t\}}{\mathcal{E} \cup \{u \approx t\}, \mathcal{R}} \quad \text{if } s \to u \text{ using } l \to r \text{ and } s \rhd l$$

delete
$$\frac{\mathcal{E} \cup \{s \approx s\}, \mathcal{R}}{}$$



KB: Standard Completion

Definition (KB)

$\mathcal{E}$: set of equations $\mathcal{R}$: set of rewrite rules $\succ$: reduction order

inference system KB consists of six rules:

orient
$$\frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{\mathcal{E}, \mathcal{R} \cup \{s \to t\}} \quad \text{if } s \succ t$$

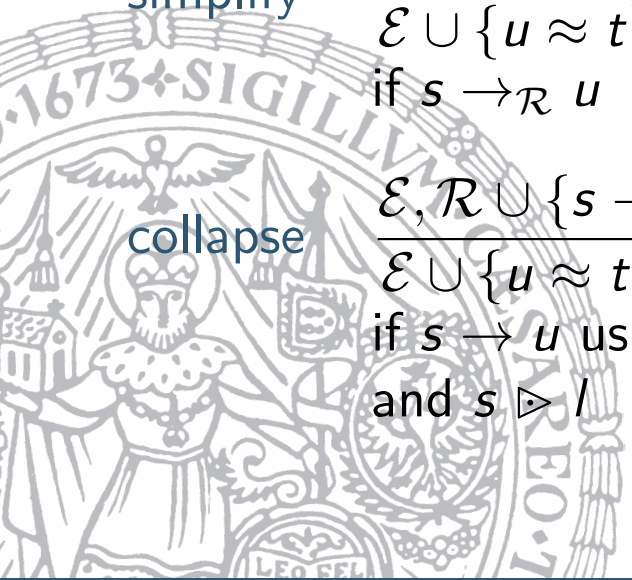
deduce
$$\frac{\mathcal{E}, \mathcal{R}}{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}} \quad \text{if } s \leftarrow_{\mathcal{R}} u \to_{\mathcal{R}} t$$

simplify
$$\frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{\mathcal{E} \cup \{u \approx t\}, \mathcal{R}} \quad \text{if } s \to_{\mathcal{R}} u$$

compose
$$\frac{\mathcal{E}, \mathcal{R} \cup \{s \to t\}}{\mathcal{E}, \mathcal{R} \cup \{s \to u\}} \quad \text{if } t \to_{\mathcal{R}} u$$

collapse
$$\frac{\mathcal{E}, \mathcal{R} \cup \{s \to t\}}{\mathcal{E} \cup \{u \approx t\}, \mathcal{R}} \quad \text{if } s \to u \text{ using } l \to r \text{ and } s \rhd l$$

delete
$$\frac{\mathcal{E} \cup \{s \approx s\}, \mathcal{R}}{\mathcal{E}, \mathcal{R}}$$



KB: Standard Completion

Definition (KB)

$\mathcal{E}$: set of equations $\mathcal{R}$: set of rewrite rules

crucial for success

$\succ$: reduction order

inference system KB consists of six rules:

orient
$$\frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{\mathcal{E}, \mathcal{R} \cup \{s \to t\}} \quad \text{if } s \succ t$$

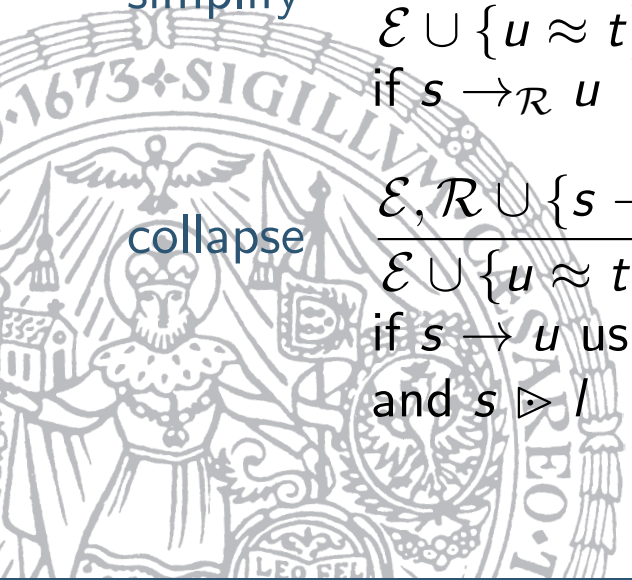
deduce
$$\frac{\mathcal{E}, \mathcal{R}}{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}} \quad \text{if } s \leftarrow_{\mathcal{R}} u \to_{\mathcal{R}} t$$

simplify
$$\frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{\mathcal{E} \cup \{u \approx t\}, \mathcal{R}} \quad \text{if } s \to_{\mathcal{R}} u$$

compose
$$\frac{\mathcal{E}, \mathcal{R} \cup \{s \to t\}}{\mathcal{E}, \mathcal{R} \cup \{s \to u\}} \quad \text{if } t \to_{\mathcal{R}} u$$

collapse
$$\frac{\mathcal{E}, \mathcal{R} \cup \{s \to t\}}{\mathcal{E} \cup \{u \approx t\}, \mathcal{R}} \quad \text{if } s \to u \text{ using } l \to r \text{ and } s \rhd l$$

delete
$$\frac{\mathcal{E} \cup \{s \approx s\}, \mathcal{R}}{\mathcal{E}, \mathcal{R}}$$



Definition

possibly infinite deduction

$$\mathcal{S}: \ \mathcal{E}_0, \mathcal{R}_0 \vdash_{\mathsf{KB}} \mathcal{E}_1, \mathcal{R}_1 \vdash_{\mathsf{KB}} \mathcal{E}_2, \mathcal{R}_2 \vdash_{\mathsf{KB}} \cdots$$

with $\mathcal{E}_0 = \mathcal{E}$ and $\mathcal{R}_0 = \varnothing$

Definition

possibly infinite deduction

$$\mathcal{S}\colon\ \mathcal{E}_0, \mathcal{R}_0 \vdash_{\mathsf{KB}} \mathcal{E}_1, \mathcal{R}_1 \vdash_{\mathsf{KB}} \mathcal{E}_2, \mathcal{R}_2 \vdash_{\mathsf{KB}} \cdots$$

with $\mathcal{E}_0 = \mathcal{E}$ and $\mathcal{R}_0 = \varnothing$

- $\mathcal{E}_\omega$ is set of persistent equations

Definition

possibly infinite deduction

$$\mathcal{S}: \ \mathcal{E}_0, \mathcal{R}_0 \vdash_{\mathsf{KB}} \mathcal{E}_1, \mathcal{R}_1 \vdash_{\mathsf{KB}} \mathcal{E}_2, \mathcal{R}_2 \vdash_{\mathsf{KB}} \cdots$$

with $\mathcal{E}_0 = \mathcal{E}$ and $\mathcal{R}_0 = \varnothing$

- $\mathcal{E}_\omega$ is set of persistent equations: $\mathcal{E}_\omega = \bigcup_{i \geqslant 0} \bigcap_{j \geqslant i} \mathcal{E}_j$

Definition

possibly infinite deduction

$$\mathcal{S}:\ \mathcal{E}_0, \mathcal{R}_0 \vdash_{\mathsf{KB}} \mathcal{E}_1, \mathcal{R}_1 \vdash_{\mathsf{KB}} \mathcal{E}_2, \mathcal{R}_2 \vdash_{\mathsf{KB}} \cdots$$

with $\mathcal{E}_0 = \mathcal{E}$ and $\mathcal{R}_0 = \varnothing$

- $\mathcal{E}_\omega$ is set of persistent equations: $\mathcal{E}_\omega = \bigcup_{i \geqslant 0} \bigcap_{j \geqslant i} \mathcal{E}_j$
- $\mathcal{R}_\omega$ is set of persistent rules

Definition

possibly infinite deduction

$$\mathcal{S}\colon\ \mathcal{E}_0, \mathcal{R}_0 \vdash_{\mathsf{KB}} \mathcal{E}_1, \mathcal{R}_1 \vdash_{\mathsf{KB}} \mathcal{E}_2, \mathcal{R}_2 \vdash_{\mathsf{KB}} \cdots$$

with $\mathcal{E}_0 = \mathcal{E}$ and $\mathcal{R}_0 = \varnothing$

- $\mathcal{E}_\omega$ is set of persistent equations: $\mathcal{E}_\omega = \bigcup_{i \geqslant 0} \bigcap_{j \geqslant i} \mathcal{E}_j$
- $\mathcal{R}_\omega$ is set of persistent rules
- $\mathcal{S}$ succeeds if $\mathcal{E}_\omega = \varnothing$ and $\mathcal{R}_\omega$ is confluent and terminating

Definition

possibly infinite deduction

$$\mathcal{S}\colon\ \mathcal{E}_0, \mathcal{R}_0 \vdash_{\mathsf{KB}} \mathcal{E}_1, \mathcal{R}_1 \vdash_{\mathsf{KB}} \mathcal{E}_2, \mathcal{R}_2 \vdash_{\mathsf{KB}} \cdots$$

with $\mathcal{E}_0 = \mathcal{E}$ and $\mathcal{R}_0 = \varnothing$

- $\mathcal{E}_\omega$ is set of persistent equations: $\mathcal{E}_\omega = \bigcup_{i \geqslant 0} \bigcap_{j \geqslant i} \mathcal{E}_j$
- $\mathcal{R}_\omega$ is set of persistent rules
- $\mathcal{S}$ succeeds if $\mathcal{E}_\omega = \varnothing$ and $\mathcal{R}_\omega$ is confluent and terminating
- $\mathcal{S}$ fails if $\mathcal{E}_\omega \neq \varnothing$

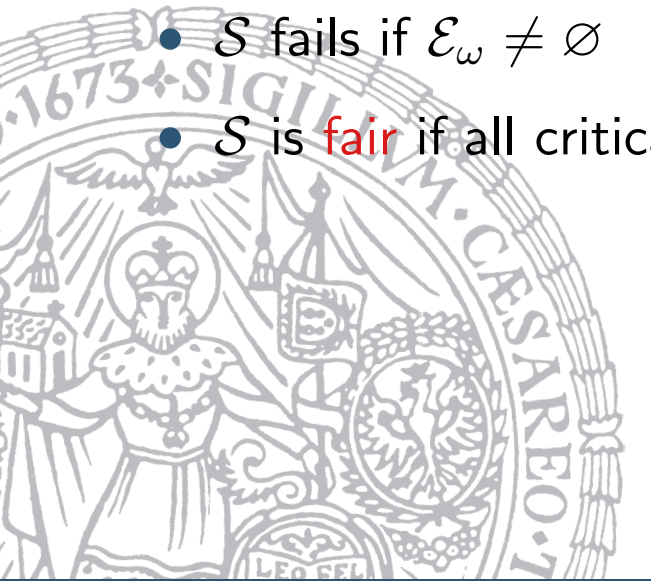
Definition

possibly infinite deduction

$$\mathcal{S}\colon\ \mathcal{E}_0, \mathcal{R}_0 \vdash_{\mathsf{KB}} \mathcal{E}_1, \mathcal{R}_1 \vdash_{\mathsf{KB}} \mathcal{E}_2, \mathcal{R}_2 \vdash_{\mathsf{KB}} \cdots$$

with $\mathcal{E}_0 = \mathcal{E}$ and $\mathcal{R}_0 = \varnothing$

- $\mathcal{E}_\omega$ is set of persistent equations: $\mathcal{E}_\omega = \bigcup_{i \geqslant 0} \bigcap_{j \geqslant i} \mathcal{E}_j$
- $\mathcal{R}_\omega$ is set of persistent rules
- $\mathcal{S}$ succeeds if $\mathcal{E}_\omega = \varnothing$ and $\mathcal{R}_\omega$ is confluent and terminating
- $\mathcal{S}$ fails if $\mathcal{E}_\omega \neq \varnothing$
- $\mathcal{S}$ is fair if all critical pairs of persisting rules are considered by deduce



Definition

possibly infinite deduction

$$\mathcal{S}\colon\ \mathcal{E}_0, \mathcal{R}_0 \vdash_{\mathsf{KB}} \mathcal{E}_1, \mathcal{R}_1 \vdash_{\mathsf{KB}} \mathcal{E}_2, \mathcal{R}_2 \vdash_{\mathsf{KB}} \cdots$$

with $\mathcal{E}_0 = \mathcal{E}$ and $\mathcal{R}_0 = \varnothing$

- $\mathcal{E}_\omega$ is set of persistent equations: $\mathcal{E}_\omega = \bigcup_{i \geqslant 0} \bigcap_{j \geqslant i} \mathcal{E}_j$
- $\mathcal{R}_\omega$ is set of persistent rules
- $\mathcal{S}$ succeeds if $\mathcal{E}_\omega = \varnothing$ and $\mathcal{R}_\omega$ is confluent and terminating
- $\mathcal{S}$ fails if $\mathcal{E}_\omega \neq \varnothing$
- $\mathcal{S}$ is fair if all critical pairs of persisting rules are considered by deduce

Theorem

every nonfailing fair deduction succeeds

Example

$$\begin{aligned} f(f(x)) &\approx g(x) \\ g(a) &\approx b \end{aligned}$$

Example

$$\begin{aligned} f(f(x)) &\approx g(x) \\ g(a) &\approx b \end{aligned}$$

- LPO with precedence $f > g > b > a$

Example

$$\begin{aligned} f(f(x)) &\approx g(x) \\ g(a) &\approx b \end{aligned}$$

- LPO with precedence $f > g > b > a$
- orient: $f(f(x)) >_{lpo} g(x)$

Example

$$\mathsf{f}(\mathsf{f}(x)) \;\to\; \mathsf{g}(x)$$

$$\mathsf{g}(\mathsf{a}) \;\approx\; \mathsf{b}$$

- LPO with precedence $\mathsf{f} > \mathsf{g} > \mathsf{b} > \mathsf{a}$
- orient: $\mathsf{f}(\mathsf{f}(x)) >_{\mathsf{lpo}} \mathsf{g}(x)$

Example

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

$$g(a) \approx b$$

- LPO with precedence $f > g > b > a$
- orient: $g(a) >_{lpo} b$

Example

$$\begin{aligned} f(f(x)) &\to g(x) \\ g(a) &\to b \end{aligned}$$

- LPO with precedence $f > g > b > a$
- orient: $g(a) >_{lpo} b$

Example

$$\begin{aligned} f(f(x)) &\to g(x) \\ g(a) &\to b \end{aligned}$$

- LPO with precedence $f > g > b > a$
- deduce: $f(g(x)) \leftarrow f(f(f(x))) \to g(f(x))$ critical pair

Example

$$\mathsf{f}(\mathsf{g}(x)) \approx \mathsf{g}(\mathsf{f}(x)) \qquad \begin{aligned} \mathsf{f}(\mathsf{f}(x)) &\to \mathsf{g}(x) \\ \mathsf{g}(\mathsf{a}) &\to \mathsf{b} \end{aligned}$$

- LPO with precedence $\mathsf{f} > \mathsf{g} > \mathsf{b} > \mathsf{a}$
- deduce: $\mathsf{f}(\mathsf{g}(x)) \leftarrow \mathsf{f}(\mathsf{f}(\mathsf{f}(x))) \to \mathsf{g}(\mathsf{f}(x))$ critical pair

Example

$$\mathsf{f}(\mathsf{g}(x)) \approx \mathsf{g}(\mathsf{f}(x)) \qquad \begin{aligned} \mathsf{f}(\mathsf{f}(x)) &\to \mathsf{g}(x) \\ \mathsf{g}(\mathsf{a}) &\to \mathsf{b} \end{aligned}$$

- LPO with precedence $\mathsf{f} > \mathsf{g} > \mathsf{b} > \mathsf{a}$
- orient: $\mathsf{f}(\mathsf{g}(x)) >_{\mathsf{lpo}} \mathsf{g}(\mathsf{f}(x))$

Example

$$\begin{aligned} f(f(x)) &\to g(x) \\ g(a) &\to b \\ f(g(x)) &\to g(f(x)) \end{aligned}$$

- LPO with precedence $f > g > b > a$
- orient: $f(g(x)) >_{lpo} g(f(x))$

Example

$$\begin{aligned} f(f(x)) &\to g(x) \\ g(a) &\to b \\ f(g(x)) &\to g(f(x)) \end{aligned}$$

- LPO with precedence $f > g > b > a$
- deduce: $f(g(f(x))) \leftarrow f(f(g(x))) \to g(g(x))$ critical pair

Example

$$\mathsf{f}(\mathsf{g}(\mathsf{f}(x))) \approx \mathsf{g}(\mathsf{g}(x))$$

$$\begin{aligned} \mathsf{f}(\mathsf{f}(x)) &\to \mathsf{g}(x) \\ \mathsf{g}(\mathsf{a}) &\to \mathsf{b} \\ \mathsf{f}(\mathsf{g}(x)) &\to \mathsf{g}(\mathsf{f}(x)) \end{aligned}$$

- LPO with precedence $\mathsf{f} > \mathsf{g} > \mathsf{b} > \mathsf{a}$
- deduce: $\mathsf{f}(\mathsf{b}) \leftarrow \mathsf{f}(\mathsf{g}(\mathsf{a})) \to \mathsf{g}(\mathsf{f}(\mathsf{a}))$ critical pair

Example

$$\begin{aligned} f(g(f(x))) &\approx g(g(x)) \\ f(b) &\approx g(f(a)) \end{aligned}$$

$$\begin{aligned} f(f(x)) &\to g(x) \\ g(a) &\to b \\ f(g(x)) &\to g(f(x)) \end{aligned}$$

- LPO with precedence $f > g > b > a$

Example

$$\begin{aligned} f(g(f(x))) &\approx g(g(x)) \\ f(b) &\approx g(f(a)) \end{aligned}$$

$$\begin{aligned} f(f(x)) &\to g(x) \\ g(a) &\to b \\ f(g(x)) &\to g(f(x)) \end{aligned}$$

- LPO with precedence $f > g > b > a$
- simplify: $f(g(f(x))) \to g(f(f(x)))$

Example

$$\begin{aligned} g(f(f(x))) &\approx g(g(x)) \\ f(b) &\approx g(f(a)) \end{aligned}$$

$$\begin{aligned} f(f(x)) &\to g(x) \\ g(a) &\to b \\ f(g(x)) &\to g(f(x)) \end{aligned}$$

- LPO with precedence $f > g > b > a$
- simplify: $g(f(f(x))) \to g(g(x))$

Example

$$\begin{aligned} g(g(x)) &\approx g(g(x)) \\ f(b) &\approx g(f(a)) \end{aligned}$$

$$\begin{aligned} f(f(x)) &\to g(x) \\ g(a) &\to b \\ f(g(x)) &\to g(f(x)) \end{aligned}$$

- LPO with precedence $f > g > b > a$
- delete: $g(g(x)) = g(g(x))$

Example

$$\mathsf{f}(\mathsf{b}) \approx \mathsf{g}(\mathsf{f}(\mathsf{a}))$$

$$\begin{aligned} \mathsf{f}(\mathsf{f}(x)) &\to \mathsf{g}(x) \\ \mathsf{g}(\mathsf{a}) &\to \mathsf{b} \\ \mathsf{f}(\mathsf{g}(x)) &\to \mathsf{g}(\mathsf{f}(x)) \end{aligned}$$

- LPO with precedence $\mathsf{f} > \mathsf{g} > \mathsf{b} > \mathsf{a}$
- orient: $\mathsf{f}(\mathsf{b}) >_{\mathsf{lpo}} \mathsf{g}(\mathsf{f}(\mathsf{a}))$

Example

$$\begin{aligned} f(f(x)) &\to g(x) \\ g(a) &\to b \\ f(g(x)) &\to g(f(x)) \\ f(b) &\to g(f(a)) \end{aligned}$$

- LPO with precedence $f > g > b > a$
- orient: $f(b) >_{lpo} g(f(a))$

Example

$$\begin{aligned} f(f(x)) &\to g(x) \\ g(a) &\to b \\ f(g(x)) &\to g(f(x)) \\ f(b) &\to g(f(a)) \end{aligned}$$

- LPO with precedence $f > g > b > a$
- deduce: $f(g(f(a))) \leftarrow f(f(b)) \to g(b)$ critical pair

Example

$$f(g(f(a))) \approx g(b)$$

$$\begin{aligned} f(f(x)) &\to g(x) \\ g(a) &\to b \\ f(g(x)) &\to g(f(x)) \\ f(b) &\to g(f(a)) \end{aligned}$$

- LPO with precedence $f > g > b > a$
- deduce: $f(g(f(a))) \leftarrow f(f(b)) \to g(b)$ critical pair

Example

$$g(f(f(a))) \approx g(b)$$

$$\begin{aligned} f(f(x)) &\to g(x) \\ g(a) &\to b \\ f(g(x)) &\to g(f(x)) \\ f(b) &\to g(f(a)) \end{aligned}$$

- LPO with precedence $f > g > b > a$
- simplify: $f(g(f(a))) \to g(f(f(a)))$

Example

$$g(g(a)) \approx g(b)$$

$$\begin{aligned} f(f(x)) &\to g(x) \\ g(a) &\to b \\ f(g(x)) &\to g(f(x)) \\ f(b) &\to g(f(a)) \end{aligned}$$

- LPO with precedence $f > g > b > a$
- simplify: $g(f(f(a))) \to g(g(a))$

Example

$$g(b) \approx g(b)$$

$$\begin{aligned} f(f(x)) &\to g(x) \\ g(a) &\to b \\ f(g(x)) &\to g(f(x)) \\ f(b) &\to g(f(a)) \end{aligned}$$

- LPO with precedence $f > g > b > a$
- simplify: $g(g(a)) \to g(b)$

Example

$$g(b) \approx g(b)$$

$$\begin{aligned} f(f(x)) &\to g(x) \\ g(a) &\to b \\ f(g(x)) &\to g(f(x)) \\ f(b) &\to g(f(a)) \end{aligned}$$

- LPO with precedence $f > g > b > a$
- delete: $g(b) = g(b)$

Example

$$\begin{aligned} f(f(x)) &\to g(x) \\ g(a) &\to b \\ f(g(x)) &\to g(f(x)) \\ f(b) &\to g(f(a)) \end{aligned}$$

- LPO with precedence $f > g > b > a$
- obtain confluent and terminating TRS

MKB: Multicompletion

$$\underbrace{\mathcal{O} = \{\succ_1, \ldots, \succ_n\}}_{\text{reduction orderings}} + \underbrace{\mathcal{E}}_{\text{equations}} \longrightarrow_{MKB} \underbrace{\mathcal{R}, \succ_i}_{\text{rewrite system}}$$

$\mathcal{R}$ is confluent, terminating with $\succ_i$, reduced and $\approx_{\mathcal{E}} = \leftrightarrow^*_{\mathcal{R}}$

MKB: Multicompletion

$$\underbrace{\mathcal{O} = \{\succ_1, \ldots, \succ_n\}}_{\text{reduction orderings}} + \underbrace{\mathcal{E}}_{\text{equations}} \longrightarrow_{MKB} \underbrace{\mathcal{R}, \succ_i}_{\text{rewrite system}}$$

$\mathcal{R}$ is confluent, terminating with $\succ_i$, reduced and $\approx_{\mathcal{E}} = \leftrightarrow^*_{\mathcal{R}}$

Definition (MKB node)

node is tuple $\langle s : t, R_0, R_1, E \rangle$ such that

- data $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{V})$
- labels R_0, R_1, E are disjoint subsets of $\{\succ_1, \ldots, \succ_n\}$

MKB: Multicompletion

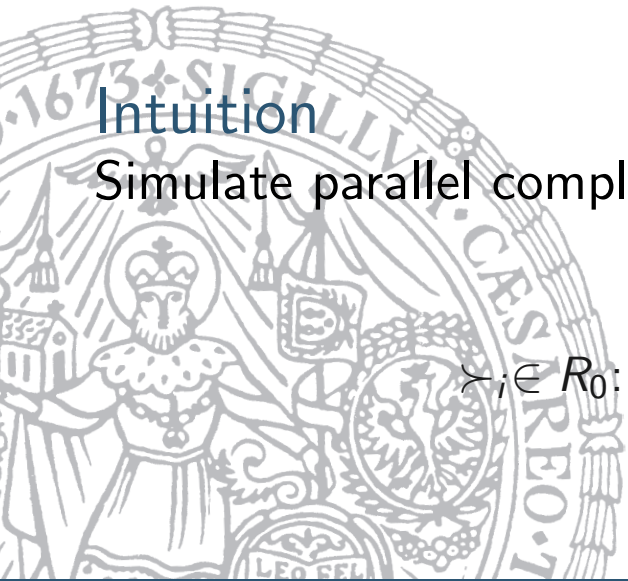
$$\underbrace{\mathcal{O} = \{\succ_1, \ldots, \succ_n\}}_{\text{reduction orderings}} + \underbrace{\mathcal{E}}_{\text{equations}} \longrightarrow_{MKB} \underbrace{\mathcal{R}, \succ_i}_{\text{rewrite system}}$$

$\mathcal{R}$ is confluent, terminating with $\succ_i$, reduced and $\approx_{\mathcal{E}} = \leftrightarrow^*_{\mathcal{R}}$

Definition (MKB node)

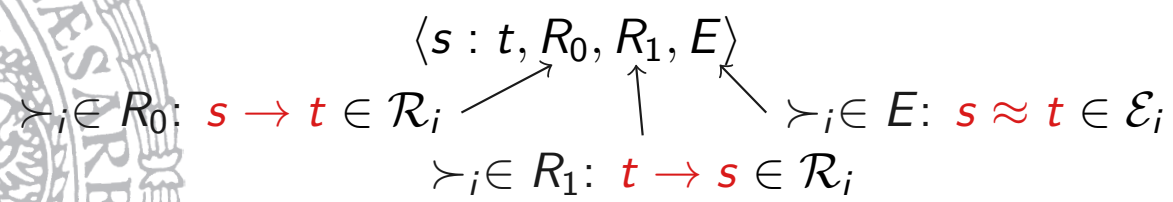
node is tuple $\langle s : t, R_0, R_1, E \rangle$ such that

- data $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{V})$
- labels R_0, R_1, E are disjoint subsets of $\{\succ_1, \ldots, \succ_n\}$



Intuition

Simulate parallel completion sequences using $\succ_i$ in state $\mathcal{E}_i, \mathcal{R}_i$:



MKB: Multicompletion

$$\underbrace{\mathcal{O} = \{\succ_1, \ldots, \succ_n\}}_{\text{reduction orderings}} + \underbrace{\mathcal{E}}_{\text{equations}} \longrightarrow_{MKB} \underbrace{\mathcal{R}, \succ_i}_{\text{rewrite system}}$$

$\mathcal{R}$ is confluent, terminating with $\succ_i$, reduced and $\approx_{\mathcal{E}} = \leftrightarrow^*_{\mathcal{R}}$

Definition (MKB node)

node is tuple $\langle s : t, R_0, R_1, E \rangle$ such that

- data $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{V})$
- labels R_0, R_1, E are disjoint subsets of $\{\succ_1, \ldots, \succ_n\}$

Definition (Orient in MKB)

$$\text{orient} \quad \frac{\mathcal{N} \uplus \{\langle s : t, R_0, R_1, E \rangle\}}{\mathcal{N} \cup \langle s : t, R_0 \cup O, R_1, E \setminus O \rangle} \quad \text{if } s \succ_i t \text{ for all } \succ_i \in O \text{ and } O \neq \varnothing$$

MKB: Multicompletion

$$\underbrace{\mathcal{O} = \{\succ_1, \ldots, \succ_n\}}_{\text{reduction orderings}} + \underbrace{\mathcal{E}}_{\text{equations}} \longrightarrow_{MKB} \underbrace{\mathcal{R}, \succ_i}_{\text{rewrite system}}$$

$\mathcal{R}$ is confluent, terminating with $\succ_i$, reduced and $\approx_{\mathcal{E}} = \leftrightarrow^*_{\mathcal{R}}$

Definition (MKB node)

node is tuple $\langle s : t, R_0, R_1, E \rangle$ such that

- data $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{V})$
- labels R_0, R_1, E are disjoint subsets of $\{\succ_1, \ldots, \succ_n\}$

Definition (Orient in MKB)

$$\text{orient} \quad \frac{\mathcal{N} \uplus \{\langle s : t, R_0, R_1, E \rangle\}}{\mathcal{N} \cup \langle s : t, R_0 \cup O, R_1, E \setminus O \rangle} \quad \text{if } s \succ_i t \text{ for all } \succ_i \in O \text{ and } O \neq \varnothing$$

- inference system: share steps among parallel deductions

KBtt: Using Termination Tools

$$\underset{\text{reduction ordering}}{\succ} + \underset{\text{equations}}{\mathcal{E}} \longrightarrow_{KB} \underset{\text{rewrite system}}{\mathcal{R}}$$

$\mathcal{R}$ is confluent, terminating, reduced and $\approx_{\mathcal{E}} = \leftrightarrow^*_{\mathcal{R}}$

Definition (Orient in KB)

$$\text{orient} \quad \frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}}{\mathcal{E}, \mathcal{R} \cup \{s \to t\}} \quad \text{if } s \succ t$$

KBtt: Using Termination Tools

$$\underset{\text{equations}}{\mathcal{E}} \longrightarrow_{KBtt} \underset{\text{rewrite system}}{\mathcal{R}}$$

$\mathcal{R}$ is confluent, terminating, reduced and $\approx_{\mathcal{E}} = \leftrightarrow^*_{\mathcal{R}}$

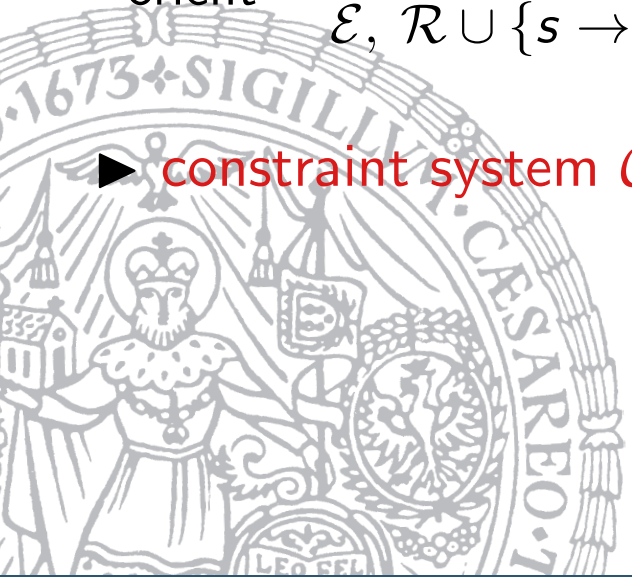
Definition (Orient in KBtt)

$$\text{orient} \quad \frac{\mathcal{E} \cup \{s \approx t\},\ \mathcal{R},\ \mathcal{C}}{\mathcal{E},\ \mathcal{R} \cup \{s \to t\},\ \mathcal{C} \cup \{s \to t\}}$$

check with termination tool

if $\mathcal{C} \cup \{s \to t\}$ terminates

- constraint system $\mathcal{C}$ collects all created rules



KBtt: Using Termination Tools

$$\underset{\text{equations}}{\mathcal{E}} \longrightarrow_{KB} \underset{\text{rewrite system}}{\mathcal{R}}$$

$\mathcal{R}$ is confluent, terminating, reduced and $\approx_{\mathcal{E}} = \leftrightarrow^*_{\mathcal{R}}$

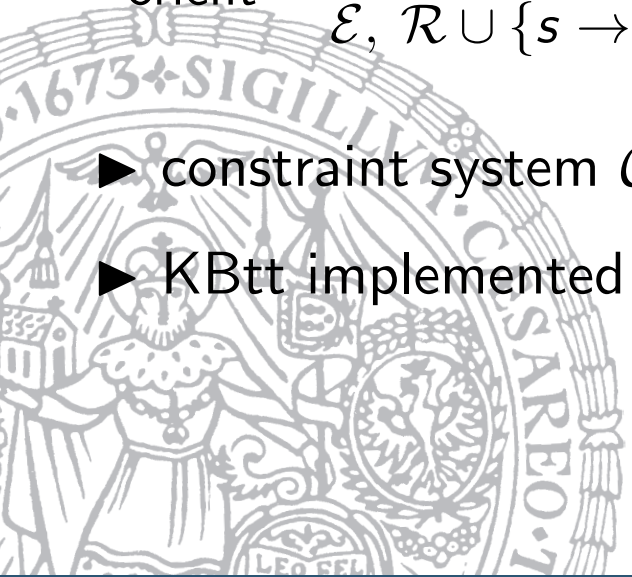
Definition (Orient in KBtt)

$$\text{orient} \quad \frac{\mathcal{E} \cup \{s \approx t\}, \mathcal{R}, \mathcal{C}}{\mathcal{E}, \mathcal{R} \cup \{s \to t\}, \mathcal{C} \cup \{s \to t\}}$$

check with termination tool

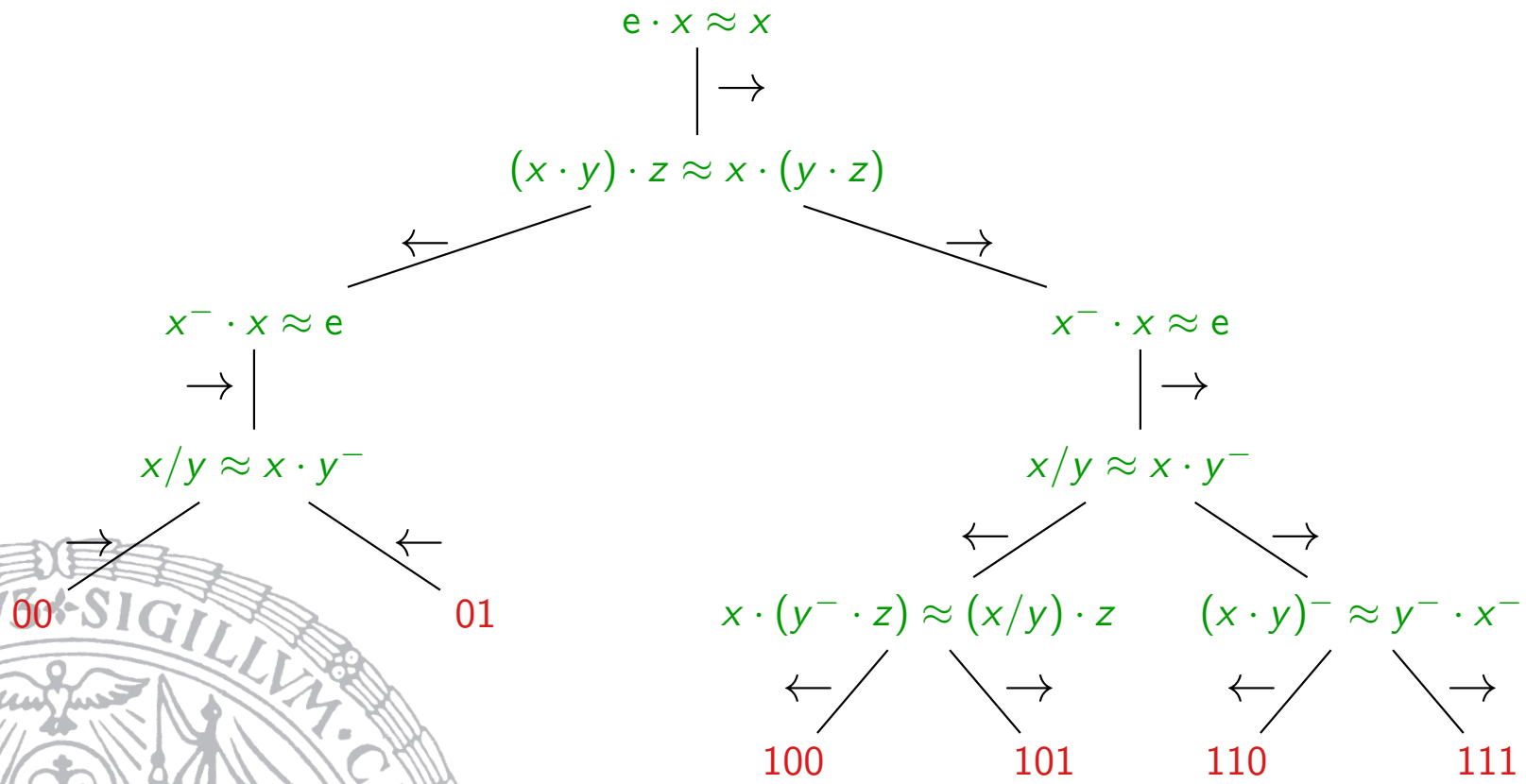
if $\mathcal{C} \cup \{s \to t\}$ terminates

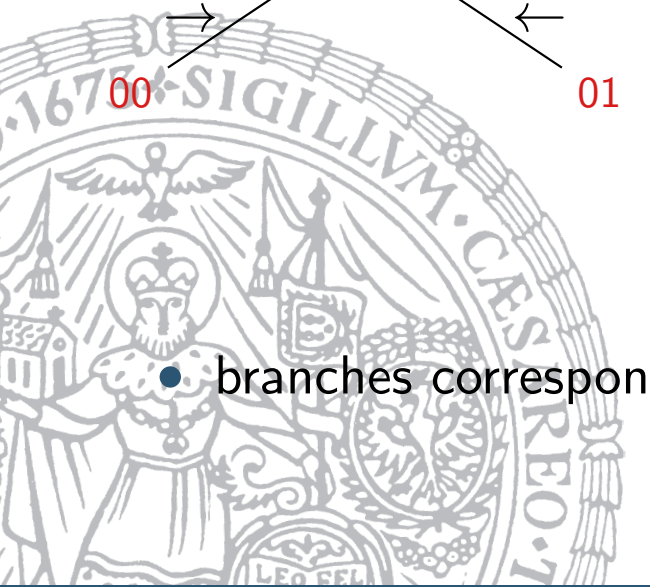
- constraint system $\mathcal{C}$ collects all created rules
- KBtt implemented in completion tool Slothrop



MKBtt: Example

$e \cdot x \approx x \quad (x \cdot y) \cdot z \approx x \cdot (y \cdot z) \quad x^- \cdot x \approx e \quad x/y \approx x \cdot y^- \quad \cdots$

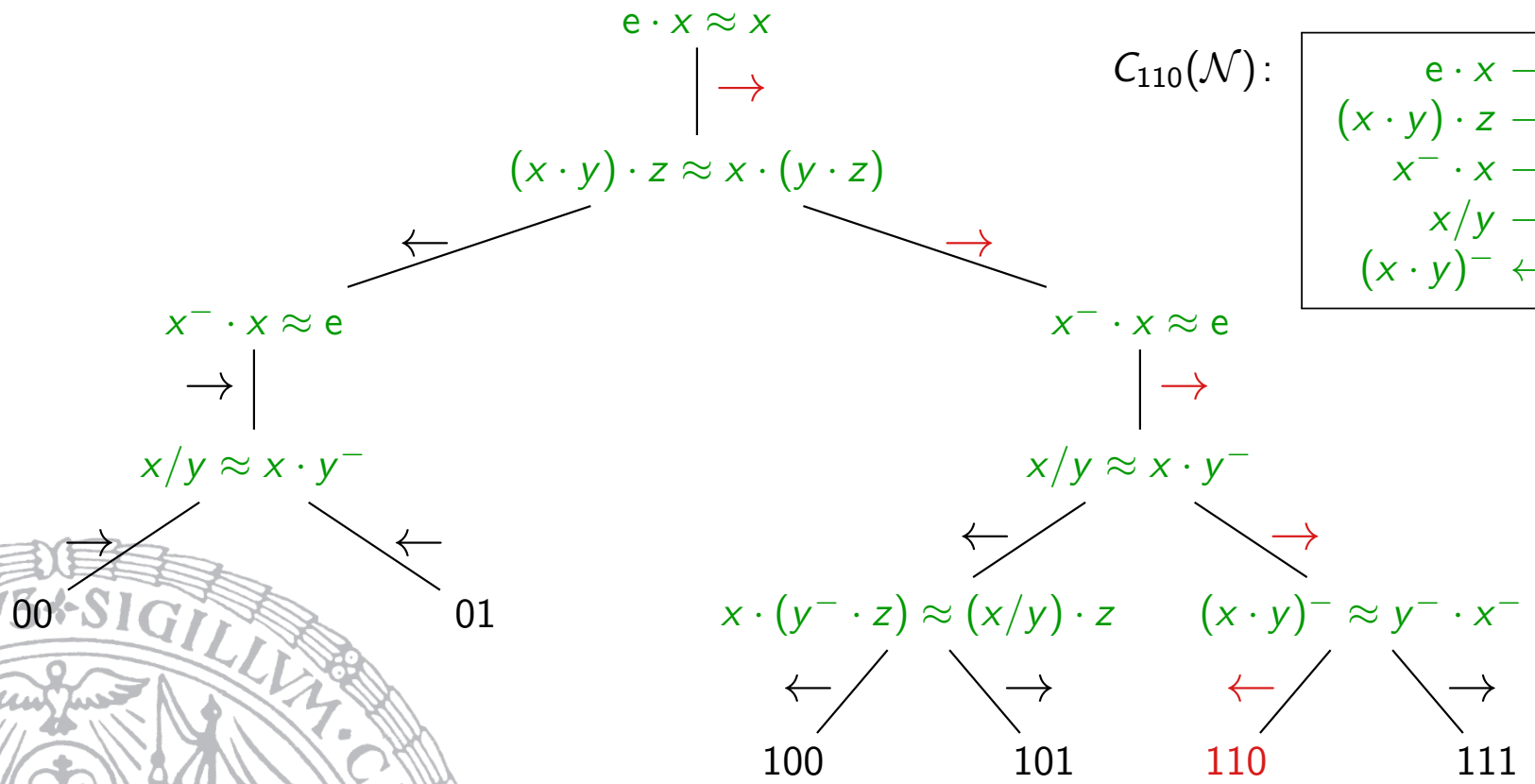




- branches correspond to KBtt processes

MKBtt: Example

$\mathsf{e} \cdot x \approx x \quad (x \cdot y) \cdot z \approx x \cdot (y \cdot z) \quad x^- \cdot x \approx \mathsf{e} \quad x/y \approx x \cdot y^- \quad \cdots$



$\mathcal{C}_{110}(\mathcal{N})$:

$$\begin{aligned} \mathsf{e} \cdot x &\to x \\ (x \cdot y) \cdot z &\to x \cdot (y \cdot z) \\ x^- \cdot x &\to \mathsf{e} \\ x/y &\to x \cdot y^- \\ (x \cdot y)^- &\leftarrow y^- \cdot x^- \end{aligned}$$

- branches correspond to KBtt processes
- orientations on branch p determine rules in $\mathcal{C}_p(\mathcal{N})$

Definition (MKBtt node)

node is tuple $\langle s : t, R_0, R_1, E, C_0, C_1 \rangle$ such that

Definition (MKBtt node)

node is tuple $\langle s : t, R_0, R_1, E, C_0, C_1 \rangle$ such that

- R_0, R_1, E, C_0, C_1 are sets of processes

Definition (MKBtt node)

node is tuple $\langle s : t, R_0, R_1, E, C_0, C_1 \rangle$ such that

- $R_0,\ R_1,\ E,\ C_0,\ C_1$ are sets of processes
- $(R_0 \cup C_0) \cap (R_1 \cup C_1) = \varnothing,\ E \cap (R_0 \cup C_0 \cup R_1 \cup C_1) = \varnothing$

Definition (MKBtt node)

node is tuple $\langle s : t, R_0, R_1, E, C_0, C_1 \rangle$ such that

- $R_0,\ R_1,\ E,\ C_0,\ C_1$ are sets of processes
- $(R_0 \cup C_0) \cap (R_1 \cup C_1) = \varnothing,\ E \cap (R_0 \cup C_0 \cup R_1 \cup C_1) = \varnothing$

Definition

orient

$$\frac{\mathcal{N} \cup \{\langle s : t, R_0, R_1, E, C_0, C_1 \rangle\}}{\{\langle s : t, R_0 \quad , R_1 \quad , \quad , C_0 \quad , C_1 \quad \rangle\}}$$

Definition (MKBtt node)

node is tuple $\langle s : t, R_0, R_1, E, C_0, C_1 \rangle$ such that

- R_0, R_1, E, C_0, C_1 are sets of processes
- $(R_0 \cup C_0) \cap (R_1 \cup C_1) = \varnothing$, $E \cap (R_0 \cup C_0 \cup R_1 \cup C_1) = \varnothing$

Definition

orient

$$\frac{\mathcal{N} \cup \{\langle s : t, R_0, R_1, E, C_0, C_1 \rangle\}}{\{\langle s : t, R_0 \quad , R_1 \quad , \quad , C_0 \quad , C_1 \quad \rangle\}}$$

- where $E_{lr}, E_{rl} \subseteq E$ such that $E_{lr} \cup E_{rl} \neq \varnothing$,

Definition (MKBtt node)

node is tuple $\langle s : t, R_0, R_1, E, C_0, C_1 \rangle$ such that

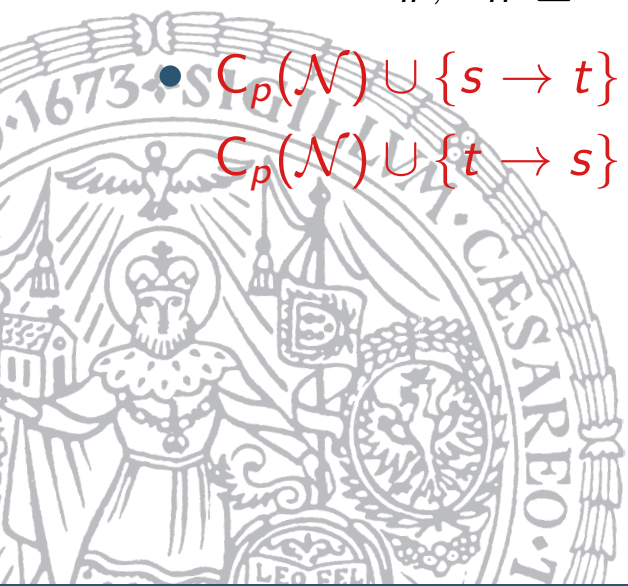
- R_0, R_1, E, C_0, C_1 are sets of processes
- $(R_0 \cup C_0) \cap (R_1 \cup C_1) = \varnothing$, $E \cap (R_0 \cup C_0 \cup R_1 \cup C_1) = \varnothing$

Definition

orient

$$\frac{\mathcal{N} \cup \{\langle s : t, R_0, R_1, E, C_0, C_1 \rangle\}}{\{\langle s : t, R_0 \quad , R_1 \quad , \quad , C_0 \quad , C_1 \quad \rangle\}}$$

- where $E_{lr}, E_{rl} \subseteq E$ such that $E_{lr} \cup E_{rl} \neq \varnothing$,
- $C_p(\mathcal{N}) \cup \{s \to t\}$ terminates for all $p \in E_{lr}$ and $C_p(\mathcal{N}) \cup \{t \to s\}$ terminates for all $p \in E_{rl}$,



Definition (MKBtt node)

node is tuple $\langle s : t, R_0, R_1, E, C_0, C_1 \rangle$ such that

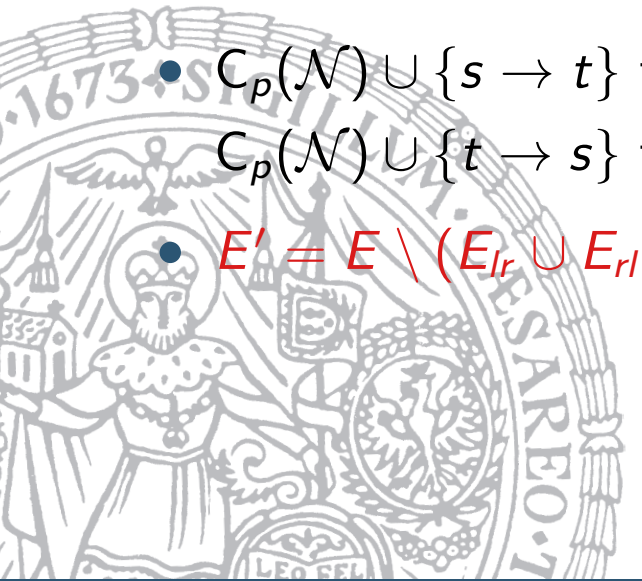
- R_0, R_1, E, C_0, C_1 are sets of processes
- $(R_0 \cup C_0) \cap (R_1 \cup C_1) = \varnothing$, $E \cap (R_0 \cup C_0 \cup R_1 \cup C_1) = \varnothing$

Definition

orient

$$\frac{\mathcal{N} \cup \{\langle s : t, R_0, R_1, E, C_0, C_1 \rangle\}}{\{\langle s : t, R_0 \quad , R_1 \quad , E', C_0 \quad , C_1 \quad \rangle\}}$$

- where $E_{lr}, E_{rl} \subseteq E$ such that $E_{lr} \cup E_{rl} \neq \varnothing$,
- $\mathsf{C}_p(\mathcal{N}) \cup \{s \to t\}$ terminates for all $p \in E_{lr}$ and $\mathsf{C}_p(\mathcal{N}) \cup \{t \to s\}$ terminates for all $p \in E_{rl}$,
- $E' = E \setminus (E_{lr} \cup E_{rl})$,



Definition (MKBtt node)

node is tuple $\langle s : t, R_0, R_1, E, C_0, C_1 \rangle$ such that

- R_0, R_1, E, C_0, C_1 are sets of processes
- $(R_0 \cup C_0) \cap (R_1 \cup C_1) = \varnothing$, $E \cap (R_0 \cup C_0 \cup R_1 \cup C_1) = \varnothing$

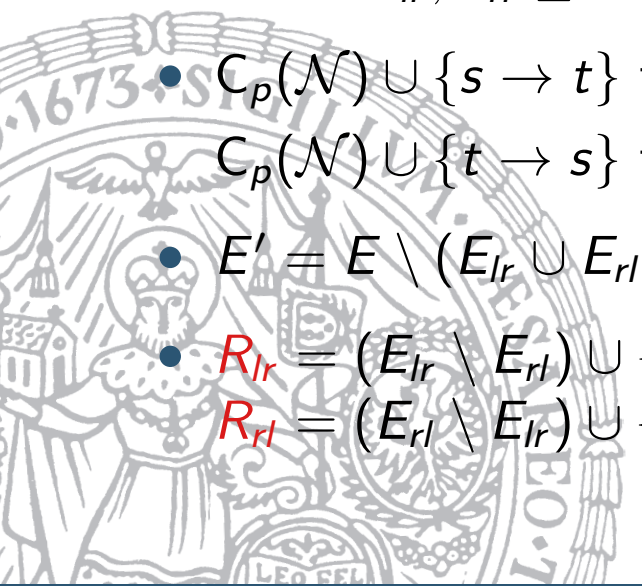
Definition

orient

$$\frac{\mathcal{N} \cup \{\langle s : t, R_0, R_1, E, C_0, C_1 \rangle\}}{\{\langle s : t, R_0 \cup R_{lr}, R_1 \cup R_{rl}, E', C_0 \cup R_{lr}, C_1 \cup R_{rl} \rangle\}}$$

- where $E_{lr}, E_{rl} \subseteq E$ such that $E_{lr} \cup E_{rl} \neq \varnothing$,
- $C_p(\mathcal{N}) \cup \{s \to t\}$ terminates for all $p \in E_{lr}$ and
  $C_p(\mathcal{N}) \cup \{t \to s\}$ terminates for all $p \in E_{rl}$,
- $E' = E \setminus (E_{lr} \cup E_{rl})$,
- $R_{lr} = (E_{lr} \setminus E_{rl}) \cup \{p0 \mid p \in (E_{rl} \cap E_{lr})\}$ and
  $R_{rl} = (E_{rl} \setminus E_{lr}) \cup \{p1 \mid p \in (E_{rl} \cap E_{lr})\}$,

split processes



Definition (MKBtt node)

node is tuple $\langle s : t, R_0, R_1, E, C_0, C_1 \rangle$ such that

- R_0, R_1, E, C_0, C_1 are sets of processes
- $(R_0 \cup C_0) \cap (R_1 \cup C_1) = \varnothing$, $E \cap (R_0 \cup C_0 \cup R_1 \cup C_1) = \varnothing$

Definition

orient

$$\frac{\mathcal{N} \cup \{\langle s : t, R_0, R_1, E, C_0, C_1 \rangle\}}{split_{E_{rl} \cap E_{lr}}(\mathcal{N}) \cup \{\langle s : t, R_0 \cup R_{lr}, R_1 \cup R_{rl}, E', C_0 \cup R_{lr}, C_1 \cup R_{rl} \rangle\}}$$

- where $E_{lr}, E_{rl} \subseteq E$ such that $E_{lr} \cup E_{rl} \neq \varnothing$,
- $C_p(\mathcal{N}) \cup \{s \to t\}$ terminates for all $p \in E_{lr}$ and $C_p(\mathcal{N}) \cup \{t \to s\}$ terminates for all $p \in E_{rl}$,
- $E' = E \setminus (E_{lr} \cup E_{rl})$,
- $R_{lr} = (E_{lr} \setminus E_{rl}) \cup \{p0 \mid p \in (E_{rl} \cap E_{lr})\}$ and $R_{rl} = (E_{rl} \setminus E_{lr}) \cup \{p1 \mid p \in (E_{rl} \cap E_{lr})\}$,
- and $split_P(\mathcal{N})$ replaces every $p \in P$ by $p0$, $p1$.

Outline

- `mkbTT` is written in OCaml ($\approx$ 5000 lines of code)

- `mkbTT` is written in OCaml ($\approx$ 5000 lines of code)
- interfaces arbitrary termination prover, or uses TT_2 internally

- `mkbTT` is written in OCaml ($\approx$ 5000 lines of code)
- interfaces arbitrary termination prover, or uses T$_{\mathsf{T}}$T$_2$ internally
- simple command-line interface

```
mkbtt cge2.trs -t 3600 -T 1 -ct -st -tp ttt2fast
```

- `mkbTT` is written in OCaml ($\approx$ 5000 lines of code)
- interfaces arbitrary termination prover, or uses T_TT_2 internally
- simple command-line interface

```
mkbtt cge2.trs -t 3600 -T 1 -ct -st -tp ttt2fast
```

input system

- `mkbTT` is written in OCaml ($\approx$ 5000 lines of code)
- interfaces arbitrary termination prover, or uses T_TT_2 internally
- simple command-line interface

```
mkbtt cge2.trs -t 3600 -T 1 -ct -st -tp ttt2fast
```

global time limit (in seconds)

- `mkbTT` is written in OCaml ($\approx$ 5000 lines of code)
- interfaces arbitrary termination prover, or uses T$_{\mathsf{T}}$T$_2$ internally
- simple command-line interface

```
mkbtt cge2.trs -t 3600 -T 1 -ct -st -tp ttt2fast
```

time limit (in seconds) for each call to termination prover

- `mkbTT` is written in OCaml ($\approx$ 5000 lines of code)
- interfaces arbitrary termination prover, or uses T$_{\mathsf{T}}$T$_2$ internally
- simple command-line interface

```
mkbtt cge2.trs -t 3600 -T 1 -ct -st -tp ttt2fast
```

termination prover

- `mkbTT` is written in OCaml ($\approx$ 5000 lines of code)
- interfaces arbitrary termination prover, or uses T$_{\mathsf{T}}$T$_2$ internally
- simple command-line interface

```
mkbtt cge2.trs -t 3600 -T 1 -ct -st -tp ttt2fast
```

output statistics

- `mkbTT` is written in OCaml ($\approx$ 5000 lines of code)
- interfaces arbitrary termination prover, or uses T_TT_2 internally
- simple command-line interface

```
mkbtt cge2.trs -t 3600 -T 1 -ct -st -tp ttt2fast
```

output completed system

- `mkbTT` is written in OCaml ($\approx$ 5000 lines of code)
- interfaces arbitrary termination prover, or uses T$_{\mathsf{T}}$T$_2$ internally
- simple command-line interface

```
mkbtt cge2.trs -t 3600 -T 1 -ct -st -tp ttt2fast
```

- `mkbTT` is written in OCaml ($\approx$ 5000 lines of code)
- interfaces arbitrary termination prover, or uses T$_{\mathsf{T}}$T$_2$ internally
- simple command-line interface

```
mkbtt cge2.trs -t 3600 -T 1 -ct -st -tp ttt2fast
```

- http://cl-informatik.uibk.ac.at/mkbtt

Outline

- hardware: Intel® Pentium™ M processor, 2 GHz, 1 GB
- global time limit: 1 hour
- termination time limit: AProVE 5 seconds

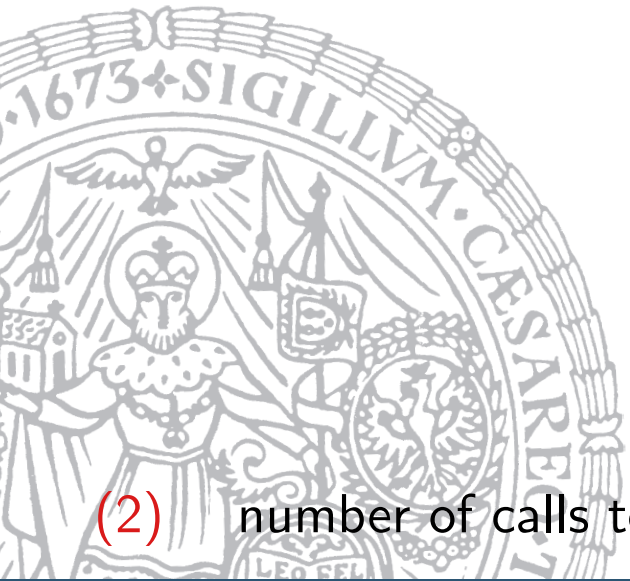
- hardware: Intel® Pentium™ M processor, 2 GHz, 1 GB
- global time limit: 1 hour
- termination time limit: AProVE 5 seconds

	Slothrop		mkbTT		
TRS	(1)	(2)	(1)	(2)	(3)
SK90_3.01	800.37	326	85.30	51	29

(1) total time in seconds

- hardware: Intel® Pentium™ M processor, 2 GHz, 1 GB
- global time limit: 1 hour
- termination time limit: AProVE 5 seconds

TRS	Slothrop (1)	Slothrop (2)	mkbTT (1)	mkbTT (2)	mkbTT (3)
SK90_3.01	800.37	326	85.30	51	29



(2) number of calls to termination prover

- hardware: Intel® Pentium™ M processor, 2 GHz, 1 GB
- global time limit: 1 hour
- termination time limit: AProVE 5 seconds

TRS	Slothrop (1)	Slothrop (2)	mkbTT (1)	mkbTT (2)	mkbTT (3)
SK90_3.01	800.37	326	85.30	51	29

(3) number of iterations

- hardware: Intel® Pentium™ M processor, 2 GHz, 1 GB
- global time limit: 1 hour
- termination time limit: AProVE 5 seconds

TRS	Slothrop (1)	Slothrop (2)	mkbTT (1)	mkbTT (2)	mkbTT (3)
SK90_3.01	800.37	326	85.30	51	29
SK90_3.04	∞		∞		
SK90_3.05	∞		913.77	225	94
SK90_3.19	84.25	43	112.65	65	11
SK90_3.27	∞		∞		
SL-cge2	∞		2793.21	1220	77
SL-cge3	∞		∞		
SL-endo	246.56	101	218.96	135	45
SL-ep	∞		∞		

∞ timeout

- hardware: Intel® Pentium™ M processor, 2 GHz, 1 GB
- global time limit: 1 hour
- termination time limit: AProVE 5 seconds

TRS	Slothrop (1)	Slothrop (2)	mkbTT (1)	mkbTT (2)	mkbTT (3)
SK90_3.01	800.37	326	85.30	51	29
SK90_3.04	∞		∞		
SK90_3.05	∞		913.77	225	94
SK90_3.19	84.25	43	112.65	65	11
SK90_3.27	∞		∞		
SL-cge2	∞		2793.21	1220	77
SL-cge3	∞		∞		
SL-endo	246.56	101	218.96	135	45
SL-ep	∞		∞		
⋮	⋮			⋮	
27 systems	∅ 2061	#12	∅ 1489	#18	

∅ average execution time

- hardware: Intel® Pentium™ M processor, 2 GHz, 1 GB
- global time limit: 1 hour
- termination time limit: AProVE 5 seconds

TRS	Slothrop (1)	Slothrop (2)	mkbTT (1)	mkbTT (2)	mkbTT (3)
SK90_3.01	800.37	326	85.30	51	29
SK90_3.04	∞		∞		
SK90_3.05	∞		913.77	225	94
SK90_3.19	84.25	43	112.65	65	11
SK90_3.27	∞		∞		
SL-cge2	∞		2793.21	1220	77
SL-cge3	∞		∞		
SL-endo	246.56	101	218.96	135	45
SL-ep	∞		∞		
⋮	⋮			⋮	
27 systems	∅ 2061	#12	∅ 1489	#18	

number of successes

- hardware: Intel® Pentium™ M processor, 2 GHz, 1 GB
- global time limit: 1 hour
- termination time limit: AProVE 5 seconds, T_TT_2 1 second

	AProVE					T_TT_2				
	Slothrop		mkbTT			Slothrop		mkbTT		
TRS	(1)	(2)	(1)	(2)	(3)	(1)	(2)	(1)	(2)	(3)
SK90_3.01	800.37	326	85.30	51	29	71.52	304	4.45	51	29
SK90_3.04	∞		∞							
SK90_3.05	∞		913.77	225	94					
SK90_3.19	84.25	43	112.65	65	11					
SK90_3.27	∞		∞							
SL-cge2	∞		2793.21	1220	77					
SL-cge3	∞		∞							
SL-endo	246.56	101	218.96	135	45					
SL-ep	∞		∞							
⋮	⋮			⋮						
27 systems	∅ 2061	#12	∅ 1489	#18						

- hardware: Intel® Pentium™ M processor, 2 GHz, 1 GB
- global time limit: 1 hour
- termination time limit: AProVE 5 seconds, TTT2 1 second

	AProVE					TTT2				
	Slothrop		mkbTT			Slothrop		mkbTT		
TRS	(1)	(2)	(1)	(2)	(3)	(1)	(2)	(1)	(2)	(3)
SK90_3.01	800.37	326	85.30	51	29	71.52	304	4.45	51	29
SK90_3.04	∞		∞			∞		508.24	658	140
SK90_3.05	∞		913.77	225	94	78.48	258	46.45	220	92
SK90_3.19	84.25	43	112.65	65	11	3.39	43	5.01	65	11
SK90_3.27	∞		∞			73.61	70	118.56	143	37
SL-cge2	∞		2793.21	1220	77	665.29	1384	246.64	1072	77
SL-cge3	∞		∞			∞		∞		
SL-endo	246.56	101	218.96	135	45	12.64	105	7.87	135	45
SL-ep	∞		∞			54.47	266	230.06	1101	26
⋮	⋮			⋮		⋮			⋮	
27 systems	∅ 2061	#12	∅ 1489	#18		∅ 2334	#20	∅ 473	#24	

- hardware: Intel® Pentium™ M processor, 2 GHz, 1 GB
- global time limit: 1 hour
- termination time limit: AProVE 5 seconds, T_TT_2 1 second

	AProVE					T_TT_2				
	Slothrop		mkbTT			Slothrop		mkbTT		
TRS	(1)	(2)	(1)	(2)	(3)	(1)	(2)	(1)	(2)	(3)
SK90_3.01	800.37	326	85.30	51	29	71.52	304	4.45	51	29
SK90_3.04	∞		∞			∞		508.24	658	140
SK90_3.05	∞		913.77	225	94	78.48	258	46.45	220	92
SK90_3.19	84.25	43	112.65	65	11	3.39	43	5.01	65	11
SK90_3.27	∞		∞			73.61	70	118.56	143	37
SL-cge2	∞		2793.21	1220	77	665.29	1384	246.64	1072	77
SL-cge3	∞		∞			∞		∞		
SL-endo	246.56	101	218.96	135	45	12.64	105	7.87	135	45
SL-ep	∞		∞			54.47	266	230.06	1101	26
⋮	⋮			⋮		⋮			⋮	
27 systems	∅ 2061	#12	∅ 1489	#18		∅ 2334	#20	∅ 473	#24	

`mkbTT` with $T_TT_2$micro manages to complete CGE_3 in (28 rules, 3277 seconds)

Conclusion

- `mkbTT` is automatic completion tool not requiring user interaction
- MKBtt approach more powerful and more efficient than KBtt
- fast termination provers (and strategies) are important

Conclusion

- `mkbTT` is automatic completion tool not requiring user interaction
- MKBtt approach more powerful and more efficient than KBtt
- fast termination provers (and strategies) are important

Future Work

- improve performance: indexing techniques, tighter coupling with $T_T T_2$
- multi-completion modulo theories (AC)
- ordered multi-completion with termination tools
- multi-completion for theorem proving (CASC – UEQ)