universität
innsbruck

Automation of Rewriting
— for Fun in Research and Profit in Teaching

Sarah Winkler

8th International Workshop on Theorem Proving Components for Educational Software
25 August 2019, Natal

Term Rewriting

- automatic analysis of TRSs constitutes theorem proving
- rewriting is at heart of equational reasoning

Term Rewriting

- automatic analysis of TRSs constitutes theorem proving
- rewriting is at heart of equational reasoning

Teaching Term Rewriting

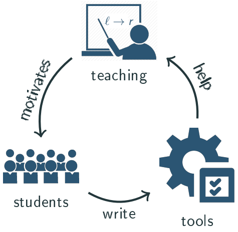
- annual course at University of Innsbruck taught by Aart Middeldorp
- since 2006: International Summer School on Rewriting

Term Rewriting

- automatic analysis of TRSs constitutes theorem proving
- rewriting is at heart of equational reasoning

Teaching Term Rewriting

- annual course at University of Innsbruck taught by Aart Middeldorp
- since 2006: International Summer School on Rewriting

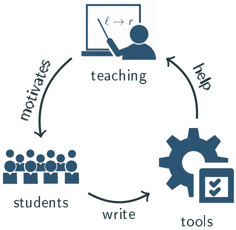


Term Rewriting

- automatic analysis of TRSs constitutes theorem proving
- rewriting is at heart of equational reasoning

Teaching Term Rewriting

- annual course at University of Innsbruck taught by Aart Middeldorp
- since 2006: International Summer School on Rewriting



Rewrite Tools Developed @ Computational Logic Group

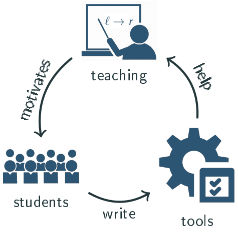
T$_\text{T}$T$_2$, T$_\text{C}$T, CSI, Cat, mkbtt, KBCV, mædmax, FORT, ProTeM, CeTA, ConCon, MiniSmt, AutoStrat, Ctrl

Term Rewriting

- automatic analysis of TRSs constitutes theorem proving
- rewriting is at heart of equational reasoning

Teaching Term Rewriting

- annual course at University of Innsbruck taught by Aart Middeldorp
- since 2006: International Summer School on Rewriting



Rewrite Tools Developed @ Computational Logic Group

$\mathsf{T_TT_2}$, $\mathsf{T_CT}$, CSI, Cat, mkbtt, KBCV, mædmax, FORT, ProTeM, CeTA, ConCon, MiniSmt, AutoStrat, Ctrl

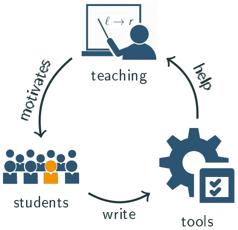
`http://cl-informatik.uibk.ac.at/research/software`

Term Rewriting

- automatic analysis of TRSs constitutes theorem proving
- rewriting is at heart of equational reasoning

Teaching Term Rewriting

- annual course at University of Innsbruck taught by Aart Middeldorp
- since 2006: International Summer School on Rewriting



Rewrite Tools Developed @ Computational Logic Group

$\mathsf{T_TT_2}$, $\mathsf{T_CT}$, CSI, Cat, mkbtt, KBCV, mædmax, FORT, ProTeM, CeTA, ConCon, MiniSmt, AutoStrat, Ctrl

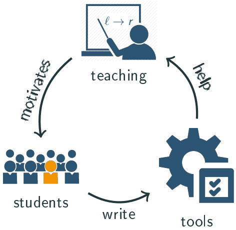
`http://cl-informatik.uibk.ac.at/research/software`

Term Rewriting

- automatic analysis of TRSs constitutes theorem proving
- rewriting is at heart of equational reasoning

Teaching Term Rewriting

- annual course at University of Innsbruck taught by Aart Middeldorp
- since 2006: International Summer School on Rewriting



Rewrite Tools Developed @ Computational Logic Group

TTT2, TCT, CSI, Cat, mkbtt, KBCV, mædmax, FORT, ProTeM, CeTA, ConCon, MiniSmt, AutoStrat, Ctrl

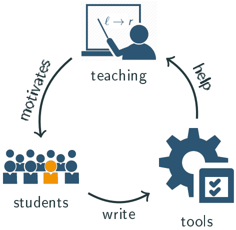
`http://cl-informatik.uibk.ac.at/research/software`

Term Rewriting

- automatic analysis of TRSs constitutes theorem proving
- rewriting is at heart of equational reasoning

Teaching Term Rewriting

- annual course at University of Innsbruck taught by Aart Middeldorp
- since 2006: International Summer School on Rewriting



Rewrite Tools Developed @ Computational Logic Group

TTT2, TCT, CSI, Cat, mkbtt, KBCV, mædmax, FORT, ProTeM, CeTA, ConCon, MiniSmt, AutoStrat, Ctrl

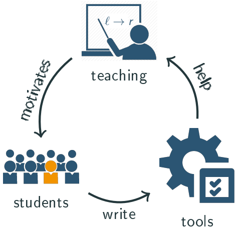
http://cl-informatik.uibk.ac.at/research/software

Term Rewriting

- automatic analysis of TRSs constitutes theorem proving
- rewriting is at heart of equational reasoning

Teaching Term Rewriting

- annual course at University of Innsbruck taught by Aart Middeldorp
- since 2006: International Summer School on Rewriting



Rewrite Tools Developed @ Computational Logic Group

TTT2, TCT, CSI, Cat, mkbtt, KBCV, mædmax, FORT, ProTeM, CeTA, ConCon, MiniSmt, AutoStrat, Ctrl

http://cl-informatik.uibk.ac.at/research/software

Outline

Teaching Example 1: Cola Gene Puzzle

Genetic engineers want to create cows that produce cola instead of milk. To that end they have to transform the DNA of the milk gene

TAGCTAGCTAGCT

in every fertilized egg into the cola gene

CTGACTGACT

Teaching Example 1: Cola Gene Puzzle

Genetic engineers want to create cows that produce cola instead of milk. To that end they have to transform the DNA of the milk gene

TAGCTAGCTAGCT

in every fertilized egg into the cola gene

CTGACTGACT

Techniques exist to perform the following DNA transformations:

TCAT $\leftrightarrow$ T   GAG $\leftrightarrow$ AG   CTC $\leftrightarrow$ TC   AGTA $\leftrightarrow$ A   TAT $\leftrightarrow$ CT

Teaching Example 1: Cola Gene Puzzle

Genetic engineers want to create cows that produce cola instead of milk. To that end they have to transform the DNA of the milk gene

TAGCTAGCTAGCT

in every fertilized egg into the cola gene

CTGACTGACT

Techniques exist to perform the following DNA transformations:

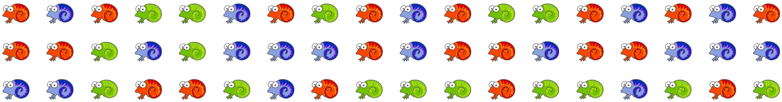
TCAT $\leftrightarrow$ T GAG $\leftrightarrow$ AG CTC $\leftrightarrow$ TC AGTA $\leftrightarrow$ A TAT $\leftrightarrow$ CT

Recently it has been discovered that the mad cow disease is caused by a retrovirus with the following DNA sequence

CTGCTACTGACT

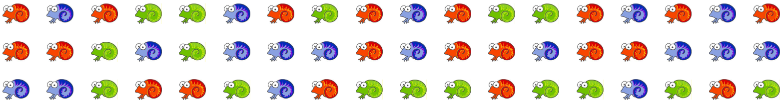
What now, if accidentally cows with this virus are created? According to the engineers there is little risk because this never happened in their experiments, but various action groups demand absolute assurance.

Teaching Example 2: Chameleon Puzzle



A colony of Brazilian chameleons consists of 20 red, 18 blue, and 16 green animals. Whenever two of different color meet, both change to the third color. Some time passes during which no chameleons are born or die nor do any enter or leave the colony.

Teaching Example 2: Chameleon Puzzle



A colony of Brazilian chameleons consists of 20 red, 18 blue, and 16 green animals. Whenever two of different color meet, both change to the third color. Some time passes during which no chameleons are born or die nor do any enter or leave the colony.

Is it possible that all 54 chameleons become the same color?

Teaching Example 3: Simple Game

- two-player game where state is sequence of black and white stones

Teaching Example 3: Simple Game

- two-player game where state is sequence of black and white stones
- allowed moves are

$$\bullet\ \bullet \rightarrow \circ \qquad \circ\ \circ \rightarrow \circ \qquad \bullet\ \circ \rightarrow \bullet \qquad \circ\ \bullet \rightarrow \bullet$$

Teaching Example 3: Simple Game

- two-player game where state is sequence of black and white stones
- allowed moves are

$$\bullet\ \bullet \to \circ \qquad \circ\ \circ \to \circ \qquad \bullet\ \circ \to \bullet \qquad \circ\ \bullet \to \bullet$$

- player who puts last white wins

Teaching Example 3: Simple Game

- two-player game where state is sequence of black and white stones
- allowed moves are

$$\bullet\ \bullet \to \circ \qquad \circ\ \circ \to \circ \qquad \bullet\ \circ \to \bullet \qquad \circ\ \bullet \to \bullet$$

- player who puts last white wins
- initial state

$$\bullet\ \bullet\ \circ\ \bullet\ \circ\ \bullet\ \bullet\ \circ\ \circ\ \bullet\ \circ\ \circ\ \bullet\ \bullet\ \circ$$

Teaching Example 3: Simple Game

- two-player game where state is sequence of black and white stones
- allowed moves are

● ● → ○ ○ ○ → ○ ● ○ → ● ○ ● → ●

- player who puts last white wins
- initial state

● ● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○

player 1 ○ ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○

Teaching Example 3: Simple Game

- two-player game where state is sequence of black and white stones
- allowed moves are

● ● → ○ ○ ○ → ○ ● ○ → ● ○ ● → ●

- player who puts last white wins
- initial state

● ● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○

player 1 ○ ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○

Teaching Example 3: Simple Game

- two-player game where state is sequence of black and white stones
- allowed moves are

● ● $\rightarrow$ ○   ○ ○ $\rightarrow$ ○   ● ○ $\rightarrow$ ●   ○ ● $\rightarrow$ ●

- player who puts last white wins
- initial state

	● ● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 1	○ ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 2	● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ○

Teaching Example 3: Simple Game

- two-player game where state is sequence of black and white stones
- allowed moves are

● ● → ○   ○ ○ → ○   ● ○ → ●   ○ ● → ●

- player who puts last white wins
- initial state

	● ● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 1	○ ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 2	● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ○

Teaching Example 3: Simple Game

- two-player game where state is sequence of black and white stones
- allowed moves are

$$\bullet\ \bullet \to \circ \qquad \circ\ \circ \to \circ \qquad \bullet\ \circ \to \bullet \qquad \circ\ \bullet \to \bullet$$

- player who puts last white wins
- initial state

	● ● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 1	○ ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 2	● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ○
player 1	● ○ ● ○ ● ● ○ ● ○ ○ ● ○

Teaching Example 3: Simple Game

- two-player game where state is sequence of black and white stones
- allowed moves are

● ● → ○ ○ ○ → ○ ● ○ → ● ○ ● → ●

- player who puts last white wins
- initial state

	● ● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 1	○ ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 2	● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ○
player 1	● ○ ● ○ ● ● ○ ● ○ ○ ● ○

Teaching Example 3: Simple Game

- two-player game where state is sequence of black and white stones
- allowed moves are

● ● → ○ ○ ○ → ○ ● ○ → ● ○ ● → ●

- player who puts last white wins
- initial state

	● ● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 1	○ ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 2	● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ○
player 1	● ○ ● ○ ● ● ○ ● ○ ○ ● ○
player 2	● ○ ● ○ ● ● ○ ● ○ ● ○

Teaching Example 3: Simple Game

- two-player game where state is sequence of black and white stones
- allowed moves are

● ● → ○   ○ ○ → ○   ● ○ → ●   ○ ● → ●

- player who puts last white wins
- initial state

	● ● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 1	○ ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 2	● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ○
player 1	● ○ ● ○ ● ● ○ ● ○ ○ ● ○
player 2	● ○ ● ○ ● ● ○ ● ○ ● ○

Teaching Example 3: Simple Game

- two-player game where state is sequence of black and white stones
- allowed moves are

● ● → ○ ○ ○ → ○ ● ○ → ● ○ ● → ●

- player who puts last white wins
- initial state

	● ● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 1	○ ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 2	● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ○
player 1	● ○ ● ○ ● ● ○ ● ○ ○ ● ○
player 2	● ○ ● ○ ● ● ○ ● ○ ● ○
player 1	● ● ○ ● ● ○ ● ○ ● ○

Teaching Example 3: Simple Game

- two-player game where state is sequence of black and white stones
- allowed moves are

● ● → ○ ○ ○ → ○ ● ○ → ● ○ ● → ●

- player who puts last white wins
- initial state

	● ● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 1	○ ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 2	● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ○
player 1	● ○ ● ○ ● ● ○ ● ○ ○ ● ○
player 2	● ○ ● ○ ● ● ○ ● ○ ● ○
player 1	● ● ○ ● ● ○ ● ○ ● ○
player 2	● ● ○ ● ● ○ ● ○ ●

Teaching Example 3: Simple Game

- two-player game where state is sequence of black and white stones
- allowed moves are

$\bullet\ \bullet \rightarrow \circ \qquad \circ\ \circ \rightarrow \circ \qquad \bullet\ \circ \rightarrow \bullet \qquad \circ\ \bullet \rightarrow \bullet$

- player who puts last white wins
- initial state

	● ● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 1	○ ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 2	● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ○
player 1	● ○ ● ○ ● ● ○ ● ○ ○ ● ○
player 2	● ○ ● ○ ● ● ○ ● ○ ● ○
player 1	● ● ○ ● ● ○ ● ○ ● ○
player 2	● ● ○ ● ● ○ ● ○ ●
player 1	● ● ○ ● ● ○ ● ●

Teaching Example 3: Simple Game

- two-player game where state is sequence of black and white stones
- allowed moves are

● ● → ○   ○ ○ → ○   ● ○ → ●   ○ ● → ●

- player who puts last white wins
- initial state

	● ● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 1	○ ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 2	● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ○
player 1	● ○ ● ○ ● ● ○ ● ○ ○ ● ○
player 2	● ○ ● ○ ● ● ○ ● ○ ● ○
player 1	● ● ○ ● ● ○ ● ○ ● ○
player 2	● ● ○ ● ● ○ ● ○ ●
player 1	● ● ○ ● ● ○ ● ●
player 2	● ● ○ ● ● ○ ○

Teaching Example 3: Simple Game

- two-player game where state is sequence of black and white stones
- allowed moves are

$$\bullet\ \bullet \rightarrow \circ \qquad \circ\ \circ \rightarrow \circ \qquad \bullet\ \circ \rightarrow \bullet \qquad \circ\ \bullet \rightarrow \bullet$$

- player who puts last white wins
- initial state

	● ● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 1	○ ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 2	● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ○
player 1	● ○ ● ○ ● ● ○ ● ○ ○ ● ○
player 2	● ○ ● ○ ● ● ○ ● ○ ● ○
player 1	● ● ○ ● ● ○ ● ○ ● ○
player 2	● ● ○ ● ● ○ ● ○ ●
player 1	● ● ○ ● ● ○ ● ●
player 2	● ● ○ ● ● ○ ○
player 1	● ● ○ ● ● ○

Teaching Example 3: Simple Game

- two-player game where state is sequence of black and white stones
- allowed moves are

● ● → ○   ○ ○ → ○   ● ○ → ●   ○ ● → ●

- player who puts last white wins
- initial state

	● ● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 1	○ ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 2	● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ○
player 1	● ○ ● ○ ● ● ○ ● ○ ○ ● ○
player 2	● ○ ● ○ ● ● ○ ● ○ ● ○
player 1	● ● ○ ● ● ○ ● ○ ● ○
player 2	● ● ○ ● ● ○ ● ○ ●
player 1	● ● ○ ● ● ○ ● ●
player 2	● ● ○ ● ● ○ ○
player 1	● ● ○ ● ● ○
player 2	● ● ○ ● ●

Teaching Example 3: Simple Game

- two-player game where state is sequence of black and white stones
- allowed moves are

● ● → ○   ○ ○ → ○   ● ○ → ●   ○ ● → ●

- player who puts last white wins
- initial state

	● ● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 1	○ ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 2	● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ○
player 1	● ○ ● ○ ● ● ○ ● ○ ○ ● ○
player 2	● ○ ● ○ ● ● ○ ● ○ ● ○
player 1	● ● ○ ● ● ○ ● ○ ● ○
player 2	● ● ○ ● ● ○ ● ○ ●
player 1	● ● ○ ● ● ○ ● ●
player 2	● ● ○ ● ● ○ ○
player 1	● ● ○ ● ● ○
player 2	● ● ○ ● ●
player 1	● ● ○ ○

Teaching Example 3: Simple Game

- two-player game where state is sequence of black and white stones
- allowed moves are

$$\bullet\ \bullet \rightarrow \circ \qquad \circ\ \circ \rightarrow \circ \qquad \bullet\ \circ \rightarrow \bullet \qquad \circ\ \bullet \rightarrow \bullet$$

- player who puts last white wins
- initial state

	● ● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 1	○ ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 2	● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ○
player 1	● ○ ● ○ ● ● ○ ● ○ ○ ● ○
player 2	● ○ ● ○ ● ● ○ ● ○ ● ○
player 1	● ● ○ ● ● ○ ● ○ ● ○
player 2	● ● ○ ● ● ○ ● ○ ●
player 1	● ● ○ ● ● ○ ● ●
player 2	● ● ○ ● ● ○ ○
player 1	● ● ○ ● ● ○
player 2	● ● ○ ● ●
player 1	● ● ○ ○
player 2	● ● ○

Teaching Example 3: Simple Game

- two-player game where state is sequence of black and white stones
- allowed moves are

● ● → ○ ○ ○ → ○ ● ○ → ● ○ ● → ●

- player who puts last white wins
- initial state

	● ● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 1	○ ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 2	● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ○
player 1	● ○ ● ○ ● ● ○ ● ○ ○ ● ○
player 2	● ○ ● ○ ● ● ○ ● ○ ● ○
player 1	● ● ○ ● ● ○ ● ○ ● ○
player 2	● ● ○ ● ● ○ ● ○ ●
player 1	● ● ○ ● ● ○ ● ●
player 2	● ● ○ ● ● ○ ○
player 1	● ● ○ ● ● ○
player 2	● ● ○ ● ●
player 1	● ● ○ ○
player 2	● ● ○
player 1	● ●

Teaching Example 3: Simple Game

- two-player game where state is sequence of black and white stones
- allowed moves are

● ● → ○ ○ ○ → ○ ● ○ → ● ○ ● → ●

- player who puts last white wins
- initial state

	● ● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 1	○ ○ ● ○ ● ● ○ ○ ● ○ ○ ● ● ○
player 2	● ○ ● ○ ● ● ○ ○ ● ○ ○ ● ○
player 1	● ○ ● ○ ● ● ○ ● ○ ○ ● ○
player 2	● ○ ● ○ ● ● ○ ● ○ ● ○
player 1	● ● ○ ● ● ○ ● ○ ● ○
player 2	● ● ○ ● ● ○ ● ○ ●
player 1	● ● ○ ● ● ○ ● ●
player 2	● ● ○ ● ● ○ ○
player 1	● ● ○ ● ● ○
player 2	● ● ○ ● ●
player 1	● ● ○ ○
player 2	● ● ○
player 1	● ●
player 2	○

Teaching Example 3: Simple Game

- two-player game where state is sequence of black and white stones
- allowed moves are

$$\bullet\ \bullet \rightarrow \circ \qquad \circ\ \circ \rightarrow \circ \qquad \bullet\ \circ \rightarrow \bullet \qquad \circ\ \bullet \rightarrow \bullet$$

- player who puts last white wins
- initial state

$$\bullet\ \bullet\ \circ\ \bullet\ \circ\ \bullet\ \bullet\ \circ\ \circ\ \bullet\ \circ\ \circ\ \bullet\ \bullet\ \circ$$

- questions
  - does the game terminate?

Teaching Example 3: Simple Game

- two-player game where state is sequence of black and white stones
- allowed moves are

$$\bullet\ \bullet \to \circ \qquad \circ\ \circ \to \circ \qquad \bullet\ \circ \to \bullet \qquad \circ\ \bullet \to \bullet$$

- player who puts last white wins
- initial state

$$\bullet\ \bullet\ \circ\ \bullet\ \circ\ \bullet\ \bullet\ \circ\ \circ\ \bullet\ \circ\ \circ\ \bullet\ \bullet\ \circ$$

- questions
  - does the game terminate?
  - which strategies are winning strategies for player 2?

Example 4: Sieve of Eratosthenes

TRS $\mathcal{R}$ models sieve of Eratosthenes to enumerate prime numbers:

$$\begin{aligned}
\mathsf{primes} &\to \mathsf{sieve}(\mathsf{from}(\mathsf{s}(\mathsf{s}(0)))) & \mathsf{sieve}(0 : y) &\to \mathsf{sieve}(y) \\
\mathsf{from}(x) &\to x : \mathsf{from}(\mathsf{s}(x)) & \mathsf{sieve}(\mathsf{s}(x) : y) &\to \mathsf{s}(x) : \mathsf{sieve}(\mathsf{filter}(x, y, x)) \\
\mathsf{hd}(x : y) &\to x & \mathsf{filter}(0, y : z, w) &\to 0 : \mathsf{filter}(w, z, w) \\
\mathsf{tl}(x : y) &\to y & \mathsf{filter}(\mathsf{s}(x), y : z, w) &\to y : \mathsf{filter}(x, z, w)
\end{aligned}$$

Example 4: Sieve of Eratosthenes

TRS $\mathcal{R}$ models sieve of Eratosthenes to enumerate prime numbers:

$$
\begin{aligned}
\mathsf{primes} &\to \mathsf{sieve}(\mathsf{from}(\mathsf{s}(\mathsf{s}(\mathsf{0})))) & \mathsf{sieve}(\mathsf{0} : y) &\to \mathsf{sieve}(y) \\
\mathsf{from}(x) &\to x : \mathsf{from}(\mathsf{s}(x)) & \mathsf{sieve}(\mathsf{s}(x) : y) &\to \mathsf{s}(x) : \mathsf{sieve}(\mathsf{filter}(x, y, x)) \\
\mathsf{hd}(x : y) &\to x & \mathsf{filter}(\mathsf{0}, y : z, w) &\to \mathsf{0} : \mathsf{filter}(w, z, w) \\
\mathsf{tl}(x : y) &\to y & \mathsf{filter}(\mathsf{s}(x), y : z, w) &\to y : \mathsf{filter}(x, z, w)
\end{aligned}
$$

Questions About (Functional) Programs

- is the given program terminating?

Example 4: Sieve of Eratosthenes

TRS $\mathcal{R}$ models sieve of Eratosthenes to enumerate prime numbers:

$$\begin{aligned}
\mathsf{primes} &\to \mathsf{sieve}(\mathsf{from}(\mathsf{s}(\mathsf{s}(0)))) & \mathsf{sieve}(0 : y) &\to \mathsf{sieve}(y)\\
\mathsf{from}(x) &\to x : \mathsf{from}(\mathsf{s}(x)) & \mathsf{sieve}(\mathsf{s}(x) : y) &\to \mathsf{s}(x) : \mathsf{sieve}(\mathsf{filter}(x, y, x))\\
\mathsf{hd}(x : y) &\to x & \mathsf{filter}(0, y : z, w) &\to 0 : \mathsf{filter}(w, z, w)\\
\mathsf{tl}(x : y) &\to y & \mathsf{filter}(\mathsf{s}(x), y : z, w) &\to y : \mathsf{filter}(x, z, w)
\end{aligned}$$

Questions About (Functional) Programs

- is the given program terminating?
- are results, if existent, unique?

Example 4: Sieve of Eratosthenes

TRS $\mathcal{R}$ models sieve of Eratosthenes to enumerate prime numbers:

$$
\begin{aligned}
\mathsf{primes} &\to \mathsf{sieve}(\mathsf{from}(\mathsf{s}(\mathsf{s}(0)))) & \mathsf{sieve}(0 : y) &\to \mathsf{sieve}(y) \\
\mathsf{from}(x) &\to x : \mathsf{from}(\mathsf{s}(x)) & \mathsf{sieve}(\mathsf{s}(x) : y) &\to \mathsf{s}(x) : \mathsf{sieve}(\mathsf{filter}(x, y, x)) \\
\mathsf{hd}(x : y) &\to x & \mathsf{filter}(0, y : z, w) &\to 0 : \mathsf{filter}(w, z, w) \\
\mathsf{tl}(x : y) &\to y & \mathsf{filter}(\mathsf{s}(x), y : z, w) &\to y : \mathsf{filter}(x, z, w)
\end{aligned}
$$

Questions About (Functional) Programs

- is the given program terminating?
- are results, if existent, unique?
- what is the program's computational complexity, if it terminates?

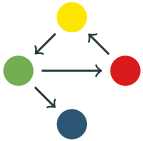
Outline

Abstract Rewriting

- abstract rewrite system is carrier set with binary relation

Example

rewrite system $\mathcal{R}$

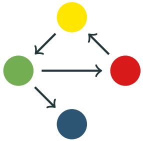


Abstract Rewriting

- abstract rewrite system is carrier set with binary relation

Example

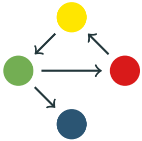
rewrite system $\mathcal{R}$



- rewrite step:

Abstract Rewriting

- abstract rewrite system is carrier set with binary relation



Example

rewrite system $\mathcal{R}$

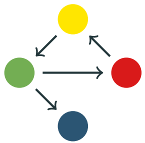
- rewrite sequence:

Abstract Rewriting

- abstract rewrite system is carrier set with binary relation

Example

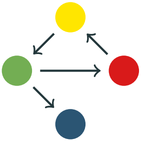
rewrite system $\mathcal{R}$



- rewrite sequence:

Abstract Rewriting

- abstract rewrite system is carrier set with binary relation
- . . . is terminating if there are no infinite rewrite sequences



Example

rewrite system $\mathcal{R}$

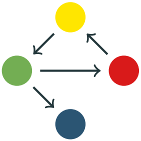
- rewrite sequence:

Abstract Rewriting

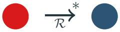
- abstract rewrite system is carrier set with binary relation
- …is terminating if there are no infinite rewrite sequences

Example

rewrite system $\mathcal{R}$



- rewrite sequence:



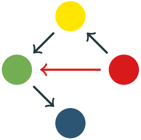
- $\mathcal{R}$ is not terminating:

Abstract Rewriting

- abstract rewrite system is carrier set with binary relation
- ... is terminating if there are no infinite rewrite sequences

Example

rewrite system $\mathcal{R}$



- rewrite sequence:

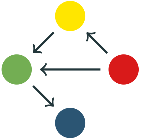
- $\mathcal{R}$ is terminating

Abstract Rewriting

- abstract rewrite system is carrier set with binary relation
- ... is terminating if there are no infinite rewrite sequences
- ... is confluent if ${}_{\mathcal{R}}^{*}\!\leftarrow \cdot \rightarrow_{\mathcal{R}}^{*} \subseteq \rightarrow_{\mathcal{R}}^{*} \cdot {}_{\mathcal{R}}^{*}\!\leftarrow$

Example

rewrite system $\mathcal{R}$



- rewrite sequence:

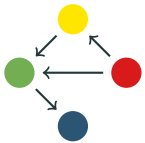
- $\mathcal{R}$ is terminating

Abstract Rewriting

- abstract rewrite system is carrier set with binary relation
- ... is terminating if there are no infinite rewrite sequences
- ... is confluent if ${}_{\mathcal{R}}^{*}\!\leftarrow \cdot \to_{\mathcal{R}}^{*} \subseteq \downarrow_{\mathcal{R}}$

Example

rewrite system $\mathcal{R}$



- rewrite sequence:

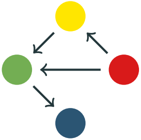
- $\mathcal{R}$ is terminating

Abstract Rewriting

- abstract rewrite system is carrier set with binary relation
- ... is terminating if there are no infinite rewrite sequences
- ... is confluent if ${}_{\mathcal{R}}^{*}\!\!\leftarrow \cdot \rightarrow_{\mathcal{R}}^{*} \subseteq \downarrow_{\mathcal{R}}$

Example

rewrite system $\mathcal{R}$



- rewrite sequence:

- $\mathcal{R}$ is terminating
- $\mathcal{R}$ is confluent

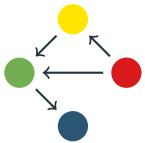
  e.g.  and 

Abstract Rewriting

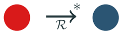
- abstract rewrite system is carrier set with binary relation
- ... is terminating if there are no infinite rewrite sequences
- ... is confluent if ${}_{\mathcal{R}}^{*}\!\leftarrow \cdot \rightarrow_{\mathcal{R}}^{*} \subseteq \downarrow_{\mathcal{R}}$

Example

rewrite system $\mathcal{R}$



- rewrite sequence:



- $\mathcal{R}$ is terminating
- $\mathcal{R}$ is confluent

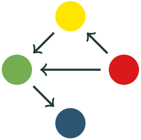
  e.g. $\rightarrow_{\mathcal{R}}^{*}$ and $\rightarrow_{\mathcal{R}}^{*}$ but also $\rightarrow_{\mathcal{R}}^{*}$ and $\rightarrow_{\mathcal{R}}^{*}$

Abstract Rewriting

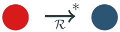
- abstract rewrite system is carrier set with binary relation
- ... is terminating if there are no infinite rewrite sequences
- ... is confluent if ${}_{\mathcal{R}}^{*}\!\leftarrow \cdot \rightarrow_{\mathcal{R}}^{*} \subseteq \downarrow_{\mathcal{R}}$
- ... is complete if terminating and confluent

Example

rewrite system $\mathcal{R}$



- rewrite sequence:



- $\mathcal{R}$ is terminating
- $\mathcal{R}$ is confluent

  e.g. $\to_{\mathcal{R}}^{*}$ and $\to_{\mathcal{R}}^{*}$ but also $\to_{\mathcal{R}}^{*}$ and $\to_{\mathcal{R}}^{*}$

Term Rewriting

- assume term structure on objects to rewrite

Term Rewriting

- assume term structure on objects to rewrite

Example

- function symbols 0, s, $+ \ldots$

Term Rewriting

- assume term structure on objects to rewrite

Example

- function symbols 0, s, $+$ ... and variables: x, y, z, ...

Term Rewriting

- assume term structure on objects to rewrite

Example

- function symbols 0, s, $+$ … and variables: x, y, z, …
- terms $\quad 0 \quad \mathsf{s}(0) \quad \mathsf{s}(\mathsf{s}(0)) \quad \mathsf{s}(\mathsf{s}(\mathsf{s}(0)))$

Term Rewriting

- assume term structure on objects to rewrite

Example

- function symbols 0, s, $+$ ... and variables: x, y, z, ...
- terms $\quad 0 \quad \mathsf{s}(0) \quad \mathsf{s}(\mathsf{s}(0)) \quad \mathsf{s}(\mathsf{s}(\mathsf{s}(0))) \quad x \quad \mathsf{s}(\mathsf{s}(x)) \quad 0 + y$

Term Rewriting

- assume term structure on objects to rewrite
- rewrite rule is pair of terms $\ell \to r$
- term rewrite system (TRS) is set of rewrite rules

Example

- function symbols 0, s, $+$ . . . and variables: x, y, z, . . .
- terms $\quad 0 \quad \mathsf{s}(0) \quad \mathsf{s}(\mathsf{s}(0)) \quad \mathsf{s}(\mathsf{s}(\mathsf{s}(0))) \quad x \quad \mathsf{s}(\mathsf{s}(x)) \quad 0 + y$

Term Rewriting

- assume term structure on objects to rewrite
- rewrite rule is pair of terms $\ell \to r$
- term rewrite system (TRS) is set of rewrite rules

Example

- function symbols 0, s, $+ \ldots$ and variables: x, y, z, $\ldots$
- terms $\quad 0 \quad \mathsf{s}(0) \quad \mathsf{s}(\mathsf{s}(0)) \quad \mathsf{s}(\mathsf{s}(\mathsf{s}(0))) \quad x \quad \mathsf{s}(\mathsf{s}(x)) \quad 0 + y$
- rewrite rules $\mathcal{R}$:

$$0 + x \to x \qquad\qquad \mathsf{s}(x) + y \to \mathsf{s}(x + y)$$

Term Rewriting

- assume term structure on objects to rewrite
- rewrite rule is pair of terms $\ell \to r$
- term rewrite system (TRS) is set of rewrite rules
- rewrite step applies rewrite rule using substitution and in context

Example

- function symbols 0, s, $+$ ... and variables: x, y, z, ...
- terms $\quad 0 \quad \mathsf{s}(0) \quad \mathsf{s}(\mathsf{s}(0)) \quad \mathsf{s}(\mathsf{s}(\mathsf{s}(0))) \quad x \quad \mathsf{s}(\mathsf{s}(x)) \quad 0 + y$
- rewrite rules $\mathcal{R}$:

$$0 + x \to x \qquad\qquad \mathsf{s}(x) + y \to \mathsf{s}(x + y)$$

Term Rewriting

- assume term structure on objects to rewrite
- rewrite rule is pair of terms $\ell \to r$
- term rewrite system (TRS) is set of rewrite rules
- rewrite step applies rewrite rule using substitution and in context

Example

- function symbols 0, s, $+$ ... and variables: x, y, z, ...
- terms $\quad 0 \quad \mathsf{s}(0) \quad \mathsf{s}(\mathsf{s}(0)) \quad \mathsf{s}(\mathsf{s}(\mathsf{s}(0))) \quad x \quad \mathsf{s}(\mathsf{s}(x)) \quad 0 + y$
- rewrite rules $\mathcal{R}$:

$$0 + x \to x \qquad\qquad \boxed{\mathsf{s}(x) + y} \to \mathsf{s}(x + y)$$

- rewrite step

$$\boxed{\mathsf{s}(\mathsf{s}(0)) + \mathsf{s}(\mathsf{s}(\mathsf{s}(0)))}$$

Term Rewriting

- assume term structure on objects to rewrite
- rewrite rule is pair of terms $\ell \to r$
- term rewrite system (TRS) is set of rewrite rules
- rewrite step applies rewrite rule using substitution and in context

Example

- function symbols 0, s, $+$ . . . and variables: x, y, z, . . .
- terms 0 $\mathsf{s}(0)$ $\mathsf{s}(\mathsf{s}(0))$ $\mathsf{s}(\mathsf{s}(\mathsf{s}(0)))$ x $\mathsf{s}(\mathsf{s}(x))$ $0 + y$
- rewrite rules $\mathcal{R}$:

$$0 + x \to x \qquad \mathsf{s}(x) + y \to \boxed{\mathsf{s}(x + y)}$$

- rewrite step

$$\mathsf{s}(\mathsf{s}(0)) + \mathsf{s}(\mathsf{s}(\mathsf{s}(0))) \to_{\mathcal{R}} \boxed{\mathsf{s}(\ \mathsf{s}(0) + \mathsf{s}(\mathsf{s}(\mathsf{s}(0)))\ )}$$

Term Rewriting

- assume term structure on objects to rewrite
- rewrite rule is pair of terms $\ell \to r$
- term rewrite system (TRS) is set of rewrite rules
- rewrite step applies rewrite rule using substitution and in context

Example

- function symbols 0, s, $+$ ... and variables: x, y, z, ...
- terms $\quad 0 \quad \mathsf{s}(0) \quad \mathsf{s}(\mathsf{s}(0)) \quad \mathsf{s}(\mathsf{s}(\mathsf{s}(0))) \quad x \quad \mathsf{s}(\mathsf{s}(x)) \quad 0 + y$
- rewrite rules $\mathcal{R}$:

$$0 + x \to x \qquad\qquad \boxed{\mathsf{s}(x) + y} \to \mathsf{s}(x + y)$$

- rewrite sequence

$$\mathsf{s}(\mathsf{s}(0)) + \mathsf{s}(\mathsf{s}(\mathsf{s}(0))) \to_{\mathcal{R}} \mathsf{s}(\boxed{\mathsf{s}(0) + \mathsf{s}(\mathsf{s}(\mathsf{s}(0)))})$$

Term Rewriting

- assume term structure on objects to rewrite
- rewrite rule is pair of terms $\ell \to r$
- term rewrite system (TRS) is set of rewrite rules
- rewrite step applies rewrite rule using substitution and in context

Example

- function symbols 0, s, $+$ ... and variables: x, y, z, ...
- terms $\quad 0 \quad \mathsf{s}(0) \quad \mathsf{s}(\mathsf{s}(0)) \quad \mathsf{s}(\mathsf{s}(\mathsf{s}(0))) \quad x \quad \mathsf{s}(\mathsf{s}(x)) \quad 0 + y$
- rewrite rules $\mathcal{R}$:

$$0 + x \to x \qquad\qquad \mathsf{s}(x) + y \to \boxed{\mathsf{s}(x + y)}$$

- rewrite sequence

$$\begin{aligned}\mathsf{s}(\mathsf{s}(0)) + \mathsf{s}(\mathsf{s}(\mathsf{s}(0))) &\to_{\mathcal{R}} \mathsf{s}(\,\mathsf{s}(0) + \mathsf{s}(\mathsf{s}(\mathsf{s}(0)))\,) \\ &\to_{\mathcal{R}} \mathsf{s}(\boxed{\mathsf{s}(\,0 + \mathsf{s}(\mathsf{s}(\mathsf{s}(0)))\,)})\end{aligned}$$

Term Rewriting

- assume term structure on objects to rewrite
- rewrite rule is pair of terms $\ell \to r$
- term rewrite system (TRS) is set of rewrite rules
- rewrite step applies rewrite rule using substitution and in context

Example

- function symbols 0, s, $+$ …and variables: x, y, z, …
- terms $\quad 0 \quad \mathsf{s}(0) \quad \mathsf{s}(\mathsf{s}(0)) \quad \mathsf{s}(\mathsf{s}(\mathsf{s}(0))) \quad x \quad \mathsf{s}(\mathsf{s}(x)) \quad 0 + y$
- rewrite rules $\mathcal{R}$:

$$\boxed{0 + x} \to x \qquad\qquad \mathsf{s}(x) + y \to \mathsf{s}(x + y)$$

- rewrite sequence

$$\begin{aligned} \mathsf{s}(\mathsf{s}(0)) + \mathsf{s}(\mathsf{s}(\mathsf{s}(0))) &\to_{\mathcal{R}} \mathsf{s}(\,\mathsf{s}(0) + \mathsf{s}(\mathsf{s}(\mathsf{s}(0)))\,) \\ &\to_{\mathcal{R}} \mathsf{s}(\,\mathsf{s}(\boxed{0 + \mathsf{s}(\mathsf{s}(\mathsf{s}(0)))})\,) \end{aligned}$$

Term Rewriting

- assume term structure on objects to rewrite
- rewrite rule is pair of terms $\ell \to r$
- term rewrite system (TRS) is set of rewrite rules
- rewrite step applies rewrite rule using substitution and in context

Example

- function symbols 0, s, $+$ ... and variables: x, y, z, ...
- terms $\quad 0 \quad \mathsf{s}(0) \quad \mathsf{s}(\mathsf{s}(0)) \quad \mathsf{s}(\mathsf{s}(\mathsf{s}(0))) \quad x \quad \mathsf{s}(\mathsf{s}(x)) \quad 0 + y$
- rewrite rules $\mathcal{R}$:

$$0 + x \to \boxed{x} \qquad\qquad \mathsf{s}(x) + y \to \mathsf{s}(x + y)$$

- rewrite sequence

$$\begin{aligned} \mathsf{s}(\mathsf{s}(0)) + \mathsf{s}(\mathsf{s}(\mathsf{s}(0))) &\to_{\mathcal{R}} \mathsf{s}(\ \mathsf{s}(0) + \mathsf{s}(\mathsf{s}(\mathsf{s}(0)))\ ) \\ &\to_{\mathcal{R}} \mathsf{s}(\ \mathsf{s}(\ 0 + \mathsf{s}(\mathsf{s}(\mathsf{s}(0)))\ )\ ) \\ &\to_{\mathcal{R}} \mathsf{s}(\mathsf{s}(\boxed{\mathsf{s}(\mathsf{s}(\mathsf{s}(0)))})) \end{aligned}$$

Term Rewriting

- assume term structure on objects to rewrite
- rewrite rule is pair of terms $\ell \to r$
- term rewrite system (TRS) is set of rewrite rules
- rewrite step applies rewrite rule using substitution and in context

Example

- function symbols 0, s, $+$ ... and variables: x, y, z, ...
- terms $\quad 0 \quad \mathsf{s}(0) \quad \mathsf{s}(\mathsf{s}(0)) \quad \mathsf{s}(\mathsf{s}(\mathsf{s}(0))) \quad x \quad \mathsf{s}(\mathsf{s}(x)) \quad 0 + y$
- rewrite rules $\mathcal{R}$:

$$0 + x \to x \qquad\qquad \mathsf{s}(x) + y \to \mathsf{s}(x + y)$$

- rewrite sequence

$$\begin{aligned} \mathsf{s}(\mathsf{s}(0)) + \mathsf{s}(\mathsf{s}(\mathsf{s}(0))) &\to_{\mathcal{R}} \mathsf{s}(\,\mathsf{s}(0) + \mathsf{s}(\mathsf{s}(\mathsf{s}(0)))\,) \\ &\to_{\mathcal{R}} \mathsf{s}(\,\mathsf{s}(\,0 + \mathsf{s}(\mathsf{s}(\mathsf{s}(0)))\,) \\ &\to_{\mathcal{R}} \mathsf{s}(\mathsf{s}(\,\mathsf{s}(\mathsf{s}(\mathsf{s}(0)))\,)) \end{aligned}$$

normal form

Term Rewriting

- assume term structure on objects to rewrite
- rewrite rule is pair of terms $\ell \to r$
- term rewrite system (TRS) is set of rewrite rules
- rewrite step applies rewrite rule using substitution and in context

Example

- function symbols 0, s, $+$ ... and variables: x, y, z, ...
- terms $\quad 0 \quad \mathsf{s}(0) \quad \mathsf{s}(\mathsf{s}(0)) \quad \mathsf{s}(\mathsf{s}(\mathsf{s}(0))) \quad x \quad \mathsf{s}(\mathsf{s}(x)) \quad 0 + y$
- rewrite rules $\mathcal{R}$:

$$0 + x \to x \qquad\qquad \mathsf{s}(x) + y \to \mathsf{s}(x + y)$$

- rewrite sequence

$$\begin{aligned} \mathsf{s}(\mathsf{s}(0)) + \mathsf{s}(\mathsf{s}(\mathsf{s}(0))) &\to_{\mathcal{R}} \mathsf{s}(\,\mathsf{s}(0) + \mathsf{s}(\mathsf{s}(\mathsf{s}(0)))\,) \\ &\to_{\mathcal{R}} \mathsf{s}(\,\mathsf{s}(\,0 + \mathsf{s}(\mathsf{s}(\mathsf{s}(0)))\,)\,) \\ &\to_{\mathcal{R}} \mathsf{s}(\mathsf{s}(\,\mathsf{s}(\mathsf{s}(\mathsf{s}(0)))\,)) \end{aligned}$$

Fact

term rewriting is Turing complete model of computation

Properties of Interest

given TRS $\mathcal{R}$,

- is every rewrite sequence terminating? termination

Properties of Interest

given TRS $\mathcal{R}$,

- is every rewrite sequence terminating? termination
- if yes, how long do $\to_{\mathcal{R}}$ sequences get? complexity

Properties of Interest

given TRS $\mathcal{R}$,

- is every rewrite sequence terminating? termination
- if yes, how long do $\to_{\mathcal{R}}$ sequences get? complexity
- is rewriting deterministic? confluence

Properties of Interest

given TRS $\mathcal{R}$,

- is every rewrite sequence terminating? termination
- if yes, how long do $\to_{\mathcal{R}}$ sequences get? complexity
- is rewriting deterministic? confluence
- can we decide the equational theory $\leftrightarrow_{\mathcal{R}}^*$? completion

Properties of Interest

given TRS $\mathcal{R}$,

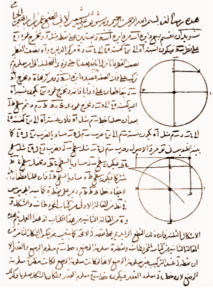
- is every rewrite sequence terminating? termination
- if yes, how long do $\to_{\mathcal{R}}$ sequences get? complexity
- is rewriting deterministic? confluence
- can we decide the equational theory $\leftrightarrow^*_{\mathcal{R}}$? completion
- is the first-order theory of $\to_{\mathcal{R}}$ decidable? decidability

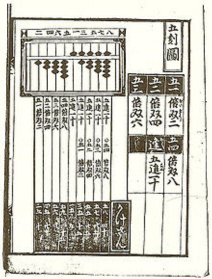
Properties of Interest

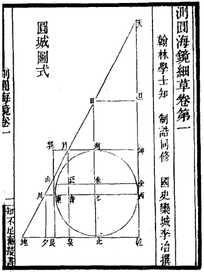
given TRS $\mathcal{R}$,

- is every rewrite sequence terminating? termination
- if yes, how long do $\rightarrow_{\mathcal{R}}$ sequences get? complexity
- is rewriting deterministic? confluence
- can we decide the equational theory $\leftrightarrow^*_{\mathcal{R}}$? completion
- is the first-order theory of $\rightarrow_{\mathcal{R}}$ decidable? decidability
- are two given rewrite sequences equivalent? proof terms

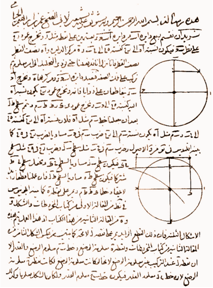
Formal Analysis Technologies

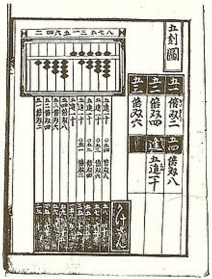


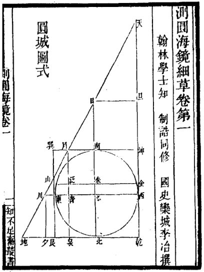




Formal Analysis Technologies







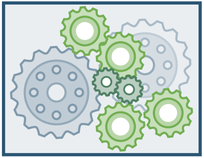
Outline

Termination

T_TT_2: Tyrolean Termination Tool 2

input: TRS $\mathcal{R}$

output: YES + termination proof, or NO + counterexample

$\mathcal{R}$ ⟶  ⟶ YES

⋯⋯⟶ ?

⟶ NO

Korp et al., Tyrolean Termination Tool 2, 2009.

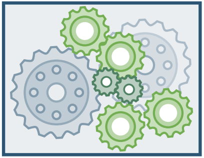
Termination

$\mathsf{T_TT_2}$: Tyrolean Termination Tool 2

input: TRS $\mathcal{R}$

output: YES + termination proof, or NO + counterexample

$\mathcal{R} \longrightarrow$



$\longrightarrow$ YES

$\cdots\cdots\rightarrow$?

$\longrightarrow$ NO

Definition

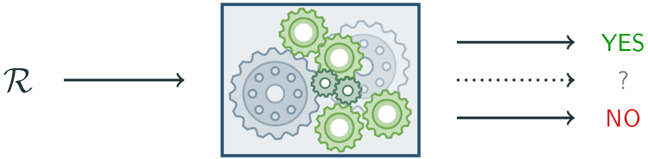
TRS $\mathcal{R}$ is terminating if $\nexists \quad t_1 \to t_2 \to t_3 \to \ldots$

Termination

$\mathsf{T_TT_2}$: Tyrolean Termination Tool 2

input: TRS $\mathcal{R}$

output: YES + termination proof, or NO + counterexample



Example (Addition)

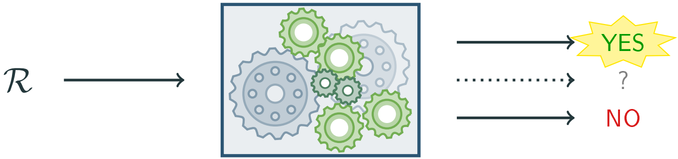
$$0 + x \to x \qquad\qquad \mathsf{s}(x) + y \to \mathsf{s}(x + y)$$

Termination

$\mathsf{T_TT_2}$: Tyrolean Termination Tool 2

input: TRS $\mathcal{R}$

output: YES + termination proof, or NO + counterexample



Example (Addition)

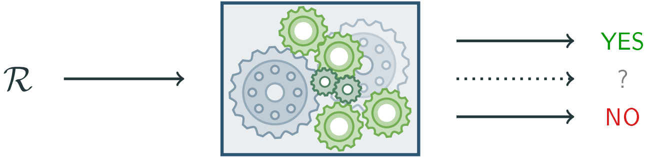
$$0 + x \to x \qquad\qquad \mathsf{s}(x) + y \to \mathsf{s}(x + y)$$

Termination

TTT2: Tyrolean Termination Tool 2

input: TRS $\mathcal{R}$

output: YES + termination proof, or NO + counterexample



Example (Simple Game)

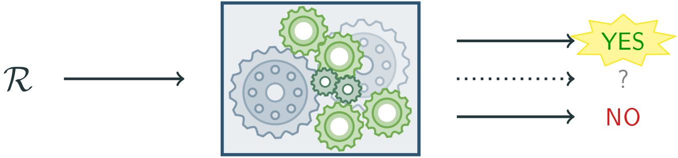
$$\bullet\ \bullet \to \circ \qquad \circ\ \circ \to \circ \qquad \bullet\ \circ \to \bullet \qquad \circ\ \bullet \to \bullet$$

Termination

$\mathsf{T_TT_2}$: Tyrolean Termination Tool 2

input: TRS $\mathcal{R}$

output: YES + termination proof, or NO + counterexample



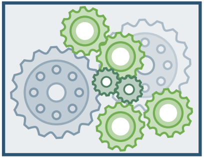
Example (Simple Game)

$\bullet\ \bullet \to \circ \qquad \circ\ \circ \to \circ \qquad \bullet\ \circ \to \bullet \qquad \circ\ \bullet \to \bullet$

TTT2: Tyrolean Termination Tool 2

input: TRS $\mathcal{R}$

output: YES + termination proof, or NO + counterexample

$\mathcal{R}$ $\longrightarrow$  $\longrightarrow$ YES; $\cdots\rightarrow$?; $\longrightarrow$ NO

Example (Sieve of Eratosthenes)

$\mathsf{primes} \to \mathsf{sieve}(\mathsf{from}(\mathsf{s}(\mathsf{s}(0))))$

$\mathsf{from}(x) \to x : \mathsf{from}(\mathsf{s}(x))$

$\mathsf{hd}(x : y) \to x$

$\mathsf{tl}(x : y) \to y$

$\mathsf{sieve}(0 : y) \to \mathsf{sieve}(y)$

$\mathsf{sieve}(\mathsf{s}(x) : y) \to \mathsf{s}(x) : \mathsf{sieve}(\mathsf{filter}(x, y, x))$

$\mathsf{filter}(0, y : z, w) \to 0 : \mathsf{filter}(w, z, w)$

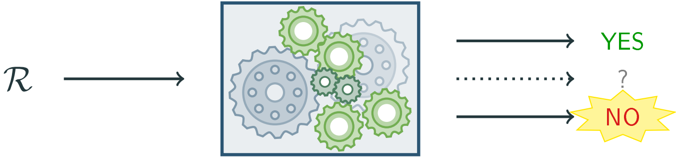
$\mathsf{filter}(\mathsf{s}(x), y : z, w) \to y : \mathsf{filter}(x, z, w)$

Termination

$\mathsf{T_TT_2}$: Tyrolean Termination Tool 2

input: TRS $\mathcal{R}$

output: YES + termination proof, or NO + counterexample



Example (Sieve of Eratosthenes)

$$\mathsf{primes} \to \mathsf{sieve}(\mathsf{from}(\mathsf{s}(\mathsf{s}(0)))) \qquad \mathsf{sieve}(0 : y) \to \mathsf{sieve}(y)$$

$$\mathsf{from}(x) \to x : \mathsf{from}(\mathsf{s}(x)) \qquad \mathsf{sieve}(\mathsf{s}(x) : y) \to \mathsf{s}(x) : \mathsf{sieve}(\mathsf{filter}(x, y, x))$$

$$\mathsf{hd}(x : y) \to x \qquad \mathsf{filter}(0, y : z, w) \to 0 : \mathsf{filter}(w, z, w)$$

$$\mathsf{tl}(x : y) \to y \qquad \mathsf{filter}(\mathsf{s}(x), y : z, w) \to y : \mathsf{filter}(x, z, w)$$

$\mathsf{T_TT_2}$: Techniques

- dependency pair (DP) framework
- dependency graph approximations
- interpretation methods: polynomials, matrices, arctic, ordinals
- reduction orders: lexicographic path order, Knuth-Bendix order, weighted path order
- labeling techniques: semantic labelling, matchbounds
- non-termination: loops and unfoldings
- ...

$\mathsf{T_TT_2}$: Techniques

- dependency pair (DP) framework
- dependency graph approximations
- interpretation methods: polynomials, matrices, arctic, ordinals
- reduction orders: lexicographic path order, Knuth-Bendix order, weighted path order
- labeling techniques: semantic labelling, matchbounds
- non-termination: loops and unfoldings
- ...

Termination Competition

- annual competition
- term rewriting: standard TRS, string rewriting, relative termination, termination modulo, conditional, innermost
- programs: C, logic programming, integer transition systems
- `http://termination-portal.org`

Research Example: Battle of Hercules and Hydra (1)

Hydra is a tree-shaped monster which grows whenever Hercules chops off a head:

If the cut-off head has a grandparent in the tree then the branch from this grandparent multiplies.

Hydra gets more and more angry: the number of copies corresponds to the number of heads already cut off.

Will Hydra ever die, such that Hercules wins?

Research Example: Battle of Hercules and Hydra (2)

process is modelled by TRS $\mathcal{R}$:

$$
\begin{aligned}
\circ(x) &\to \bullet(\square(x)) & \bullet(\square(x)) &\to \square(\bullet(\bullet(x))) & \bullet(x) &\to x \\
\square(\circ(x)) &\to \circ(\square(x)) & \bullet(\mathsf{c}_1(x,y)) &\to \mathsf{c}_1(x,\mathsf{H}(x,y)) \\
\mathsf{H}(0,x) &\to \circ(x) & \bullet(\mathsf{H}(\mathsf{H}(0,y),z)) &\to \mathsf{c}_1(y,z) \\
\mathsf{c}_2(x,y,z) &\to \circ(\mathsf{H}(y,z)) & \bullet(\mathsf{H}(\mathsf{H}(\mathsf{H}(0,x),y),z)) &\to \mathsf{c}_2(x,y,z) \\
\mathsf{c}_1(y,z) &\to \circ(z) & \bullet(\mathsf{c}_2(x,y,z)) &\to \mathsf{c}_2(x,\mathsf{H}(x,y),z)
\end{aligned}
$$

Touzet, Encoding the Hydra Battle as a Rewrite System, 1998.

Research Example: Battle of Hercules and Hydra (2)

process is modelled by TRS $\mathcal{R}$:

$$
\begin{array}{rclrclrcl}
\circ(x) & \to & \bullet(\Box(x)) & \bullet(\Box(x)) & \to & \Box(\bullet(\bullet(x))) & \bullet(x) & \to & x \\
\Box(\circ(x)) & \to & \circ(\Box(x)) & \bullet(\mathsf{c_1}(x,y)) & \to & \mathsf{c_1}(x,\mathsf{H}(x,y)) & & & \\
\mathsf{H}(0,x) & \to & \circ(x) & \bullet(\mathsf{H}(\mathsf{H}(0,y),z)) & \to & \mathsf{c_1}(y,z) & & & \\
\mathsf{c_2}(x,y,z) & \to & \circ(\mathsf{H}(y,z)) & \bullet(\mathsf{H}(\mathsf{H}(\mathsf{H}(0,x),y),z)) & \to & \mathsf{c_2}(x,y,z) & & & \\
\mathsf{c_1}(y,z) & \to & \circ(z) & \bullet(\mathsf{c_2}(x,y,z)) & \to & \mathsf{c_2}(x,\mathsf{H}(x,y),z) & & &
\end{array}
$$

Long-Standing Open Problem

show termination of $\mathcal{R}$ automatically

Research Example: Battle of Hercules and Hydra (2)

process is modelled by TRS $\mathcal{R}$:

$$\begin{aligned}
\circ(x) &\to \bullet(\square(x)) & \bullet(\square(x)) &\to \square(\bullet(\bullet(x))) & \bullet(x) &\to x \\
\square(\circ(x)) &\to \circ(\square(x)) & \bullet(\mathsf{c}_1(x,y)) &\to \mathsf{c}_1(x,\mathsf{H}(x,y)) \\
\mathsf{H}(0,x) &\to \circ(x) & \bullet(\mathsf{H}(\mathsf{H}(0,y),z)) &\to \mathsf{c}_1(y,z) \\
\mathsf{c}_2(x,y,z) &\to \circ(\mathsf{H}(y,z)) & \bullet(\mathsf{H}(\mathsf{H}(\mathsf{H}(0,x),y),z)) &\to \mathsf{c}_2(x,y,z) \\
\mathsf{c}_1(y,z) &\to \circ(z) & \bullet(\mathsf{c}_2(x,y,z)) &\to \mathsf{c}_2(x,\mathsf{H}(x,y),z)
\end{aligned}$$

Long-Standing Open Problem

show termination of $\mathcal{R}$ automatically

Zankl, Winkler, and Middeldorp, **Beyond Polynomials and Peano Arithmetic – Automation of Elementary and Ordinal Interpretations**, 2015.

Research Example: Battle of Hercules and Hydra (2)

process is modelled by TRS $\mathcal{R}$:

$$
\begin{array}{rlrlrl}
\circ(x) & \to \bullet(\square(x)) & \bullet(\square(x)) & \to \square(\bullet(\bullet(x))) & \bullet(x) & \to x \\
\square(\circ(x)) & \to \circ(\square(x)) & \bullet(\mathsf{c}_1(x,y)) & \to \mathsf{c}_1(x,\mathsf{H}(x,y)) & & \\
\mathsf{H}(0,x) & \to \circ(x) & \bullet(\mathsf{H}(\mathsf{H}(0,y),z)) & \to \mathsf{c}_1(y,z) & & \\
\mathsf{c}_2(x,y,z) & \to \circ(\mathsf{H}(y,z)) & \bullet(\mathsf{H}(\mathsf{H}(\mathsf{H}(0,x),y),z)) & \to \mathsf{c}_2(x,y,z) & & \\
\mathsf{c}_1(y,z) & \to \circ(z) & \bullet(\mathsf{c}_2(x,y,z)) & \to \mathsf{c}_2(x,\mathsf{H}(x,y),z) & &
\end{array}
$$

Long-Standing Open Problem

show termination of $\mathcal{R}$ automatically

TTT2

Zankl, Winkler, and Middeldorp, **Beyond Polynomials and Peano Arithmetic – Automation of Elementary and Ordinal Interpretations**, 2015.

Ctrl

tool to analyze properties of logically constrained rewrite systems:
allow rewrite rules with side conditions over decidable logic

Kop and Nishida, Constrained Term Rewriting tooL, 2015.

Ctrl

tool to analyze properties of logically constrained rewrite systems:
allow rewrite rules with side conditions over decidable logic

Instcombine Pass in LLVM Middle End

- LLVM provides widely used compilation toolchain

Ctrl

tool to analyze properties of logically constrained rewrite systems:
allow rewrite rules with side conditions over decidable logic

Instcombine Pass in LLVM Middle End

- LLVM provides widely used compilation toolchain
- $>$1000 algebraic simplifications of expressions:
  multiplications to shifts, reordering bitwise operations, ...

Ctrl

tool to analyze properties of logically constrained rewrite systems:
allow rewrite rules with side conditions over decidable logic

Instcombine Pass in LLVM Middle End

- LLVM provides widely used compilation toolchain
- $>$1000 algebraic simplifications of expressions:
  multiplications to shifts, reordering bitwise operations, ...
- optimization set is community maintained, termination is crucial

Ctrl

tool to analyze properties of logically constrained rewrite systems:
allow rewrite rules with side conditions over decidable logic

Instcombine Pass in LLVM Middle End

- LLVM provides widely used compilation toolchain
- >1000 algebraic simplifications of expressions:
  multiplications to shifts, reordering bitwise operations, ...
- optimization set is community maintained, termination is crucial

Research Example: LLVM Expression Simplification

simplification seeking opportunity to replace mul by shift:

$$\mathsf{mul}(\mathsf{sub}(y,x),z) \to \mathsf{mul}(\mathsf{sub}(x,y),\mathsf{abs}(z)) \;\; [z < \mathbf{0}_8 \wedge \mathsf{isPowerOf2}(\mathsf{abs}(z))]$$

Ctrl

tool to analyze properties of logically constrained rewrite systems:
allow rewrite rules with side conditions over decidable logic

Instcombine Pass in LLVM Middle End

- LLVM provides widely used compilation toolchain
- >1000 algebraic simplifications of expressions:
  multiplications to shifts, reordering bitwise operations, ...
- optimization set is community maintained, termination is crucial

Research Example: LLVM Expression Simplification

simplification seeking opportunity to replace mul by shift:

$$\mathsf{mul}(\mathsf{sub}(y, x), z) \to \mathsf{mul}(\mathsf{sub}(x, y), \mathsf{abs}(z)) \;\; [z < \mathbf{0}_8 \wedge \mathsf{isPowerOf2}(\mathsf{abs}(z))]$$

Ctrl can detect loop:

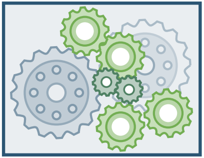
$$\begin{aligned} \mathsf{mul}(\mathsf{sub}(\mathbf{1}_8, \mathbf{1}_8), (-\mathbf{128})_8) &\to_{\mathcal{R}} \mathsf{mul}(\mathsf{sub}(\mathbf{1}_8, \mathbf{1}_8), \mathsf{abs}((-\mathbf{128})_8)) \\ &\to_{\mathsf{calc}} \mathsf{mul}(\mathsf{sub}(\mathbf{1}_8, \mathbf{1}_8), (-\mathbf{128})_8) \end{aligned}$$

Nishida and Winkler, Loop Detection by Logically Constrained Term Rewriting. 2018.

CSI

input: term rewrite system $\mathcal{R}$

output: YES + confluence proof, or NO + counterexample

$\mathcal{R} \longrightarrow$ 

$\longrightarrow$ YES

$\cdots\cdots\rightarrow$?

$\longrightarrow$ NO

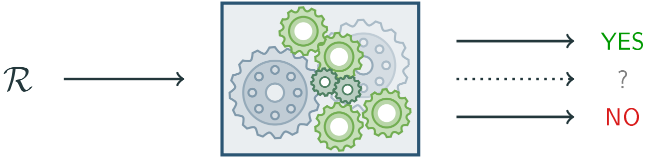
Nagele, Felgenhauer, and Middeldorp, **CSI: New Evidence — A Progress Report**, 2017.

Confluence

CSI

input: term rewrite system $\mathcal{R}$

output: YES + confluence proof, or NO + counterexample



Definition

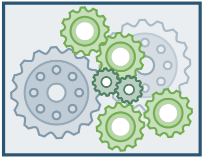
TRS $\mathcal{R}$ is confluent if ${}_{\mathcal{R}}^{*}\!\leftarrow \cdot \to_{\mathcal{R}}^{*} \subseteq \to_{\mathcal{R}}^{*} \cdot {}_{\mathcal{R}}^{*}\!\leftarrow$

CSI

input: term rewrite system $\mathcal{R}$

output: YES + confluence proof, or NO + counterexample

$\mathcal{R} \longrightarrow$



$\longrightarrow$ YES

$\cdots\cdots\rightarrow$?

$\longrightarrow$ NO

Definition

TRS $\mathcal{R}$ is confluent if ${}_{\mathcal{R}}^{*}\!\!\leftarrow \cdot \rightarrow_{\mathcal{R}}^{*} \subseteq \rightarrow_{\mathcal{R}}^{*} \cdot {}_{\mathcal{R}}^{*}\!\!\leftarrow$

$$\begin{tikzcd} & \cdot \arrow[dl, "*"'] \arrow[dr, "*"] & \\ \cdot & & \cdot \end{tikzcd}$$

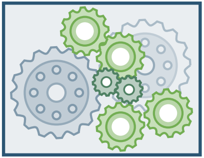
Confluence

CSI

input: term rewrite system $\mathcal{R}$

output: YES + confluence proof, or NO + counterexample

$\mathcal{R} \longrightarrow$



$\longrightarrow$ YES

$\cdots\cdots\rightarrow$?

$\longrightarrow$ NO

Definition

TRS $\mathcal{R}$ is confluent if ${}_{\mathcal{R}}^{*}\!\leftarrow \cdot \to_{\mathcal{R}}^{*} \;\subseteq\; \to_{\mathcal{R}}^{*} \cdot {}_{\mathcal{R}}^{*}\!\leftarrow$

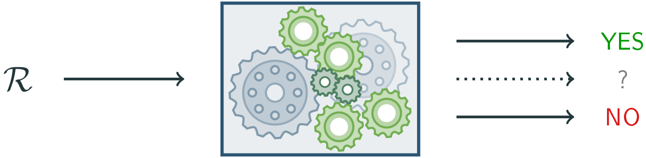
$$\begin{tikzcd}
 & \cdot \arrow[dl, "*"'] \arrow[dr, "*"] & \\
\cdot \arrow[dr, dotted, "*"'] & & \cdot \arrow[dl, dotted, "*"] \\
 & \cdot &
\end{tikzcd}$$

Confluence

CSI

input: term rewrite system $\mathcal{R}$

output: YES + confluence proof, or NO + counterexample



Example (Combinatory Logic)

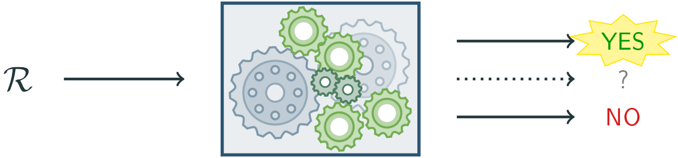
$$
\begin{aligned}
\mathsf{I} \cdot x &\to x \\
(\mathsf{K} \cdot x) \cdot y &\to x \\
((\mathsf{S} \cdot x) \cdot y) \cdot z &\to (x \cdot z) \cdot (y \cdot z)
\end{aligned}
$$

Confluence

CSI

input: term rewrite system $\mathcal{R}$

output: YES + confluence proof, or NO + counterexample



Example (Combinatory Logic)

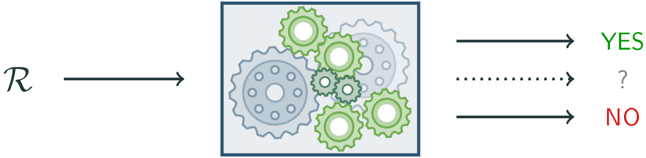
$$
\begin{aligned}
\mathsf{I} \cdot x &\to x \\
(\mathsf{K} \cdot x) \cdot y &\to x \\
((\mathsf{S} \cdot x) \cdot y) \cdot z &\to (x \cdot z) \cdot (y \cdot z)
\end{aligned}
$$

Confluence

CSI

input: term rewrite system $\mathcal{R}$

output: YES + confluence proof, or NO + counterexample



Example (Sieve of Eratosthenes)

$$\mathsf{primes} \to \mathsf{sieve}(\mathsf{from}(\mathsf{s}(\mathsf{s}(0)))) \qquad \mathsf{sieve}(0 : y) \to \mathsf{sieve}(y)$$

$$\mathsf{from}(x) \to x : \mathsf{from}(\mathsf{s}(x)) \qquad \mathsf{sieve}(\mathsf{s}(x) : y) \to \mathsf{s}(x) : \mathsf{sieve}(\mathsf{filter}(x, y, x))$$

$$\mathsf{hd}(x : y) \to x \qquad \mathsf{filter}(0, y : z, w) \to 0 : \mathsf{filter}(w, z, w)$$

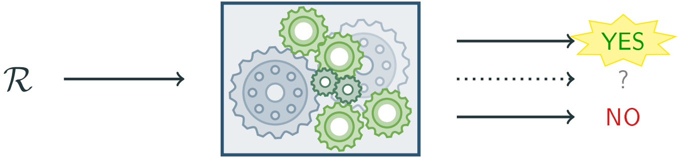
$$\mathsf{tl}(x : y) \to y \qquad \mathsf{filter}(\mathsf{s}(x), y : z, w) \to y : \mathsf{filter}(x, z, w)$$

Confluence

CSI

input: term rewrite system $\mathcal{R}$

output: YES + confluence proof, or NO + counterexample



Example (Sieve of Eratosthenes)

$$
\begin{aligned}
\mathsf{primes} &\to \mathsf{sieve}(\mathsf{from}(\mathsf{s}(\mathsf{s}(0)))) & \mathsf{sieve}(0 : y) &\to \mathsf{sieve}(y) \\
\mathsf{from}(x) &\to x : \mathsf{from}(\mathsf{s}(x)) & \mathsf{sieve}(\mathsf{s}(x) : y) &\to \mathsf{s}(x) : \mathsf{sieve}(\mathsf{filter}(x, y, x)) \\
\mathsf{hd}(x : y) &\to x & \mathsf{filter}(0, y : z, w) &\to 0 : \mathsf{filter}(w, z, w) \\
\mathsf{tl}(x : y) &\to y & \mathsf{filter}(\mathsf{s}(x), y : z, w) &\to y : \mathsf{filter}(x, z, w)
\end{aligned}
$$

Teaching Example: Simple Game

$\bullet\ \bullet \rightarrow \circ \qquad \circ\ \circ \rightarrow \circ \qquad \bullet\ \circ \rightarrow \bullet \qquad \circ\ \bullet \rightarrow \bullet$

Teaching Example: Simple Game

$\bullet\ \bullet \to \circ \qquad \circ\ \circ \to \circ \qquad \bullet\ \circ \to \bullet \qquad \circ\ \bullet \to \bullet$

Lemma (Knuth and Bendix, 1970)

terminating TRS $\mathcal{R}$ is confluent if $s \downarrow_{\mathcal{R}} t$ for all critical pairs $s \approx t$ of $\mathcal{R}$

Teaching Example: Simple Game

$\bullet\ \bullet \to \circ \qquad \circ\ \circ \to \circ \qquad \bullet\ \circ \to \bullet \qquad \circ\ \bullet \to \bullet$

Lemma (Knuth and Bendix, 1970)

terminating TRS $\mathcal{R}$ is confluent if $s \downarrow_{\mathcal{R}} t$ for all critical pairs $s \approx t$ of $\mathcal{R}$

Definition (Critical Pair)

for $\ell_1 \to r_1$ and $\ell_2 \to r_2$ renamings of rules in $\mathcal{R}$ such that

- $p \in \mathcal{P}\mathrm{os}_{\mathcal{F}}(\ell_2)$,
- mgu σ unifies $\ell_2|_p$ and ℓ_1, and
- if $p = \epsilon$ then $\ell_1 \to r_1$ and $\ell_2 \to r_2$ are not variants

$\ell_2\sigma[r_1\sigma]_p \approx r_2\sigma$ is critical pair

Teaching Example: Simple Game

$\bullet\ \bullet \to \circ$ $\qquad$ $\circ\ \circ \to \circ$ $\qquad$ $\bullet\ \circ \to \bullet$ $\qquad$ $\circ\ \bullet \to \bullet$

Lemma (Knuth and Bendix, 1970)

terminating TRS $\mathcal{R}$ is confluent if $s \downarrow_{\mathcal{R}} t$ for all critical pairs $s \approx t$ of $\mathcal{R}$

Definition (Critical Pair)

for $\ell_1 \to r_1$ and $\ell_2 \to r_2$ renamings of rules in $\mathcal{R}$ such that

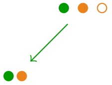
- $p \in \mathcal{P}\mathrm{os}_{\mathcal{F}}(\ell_2)$,
- mgu σ unifies $\ell_2|_p$ and ℓ_1, and
- if $p = \epsilon$ then $\ell_1 \to r_1$ and $\ell_2 \to r_2$ are not variants

$\ell_2\sigma[r_1\sigma]_p \approx r_2\sigma$ is critical pair, $\mathrm{CP}(\mathcal{R})$ are all critical pairs of $\mathcal{R}$.

Teaching Example: Simple Game

● ● → ○   ○ ○ → ○   ● ○ → ●   ○ ● → ●

critical pairs:



Lemma (Knuth and Bendix, 1970)

terminating TRS $\mathcal{R}$ is confluent if $s \downarrow_{\mathcal{R}} t$ for all critical pairs $s \approx t$ of $\mathcal{R}$

Definition (Critical Pair)

for $\ell_1 \to r_1$ and $\ell_2 \to r_2$ renamings of rules in $\mathcal{R}$ such that

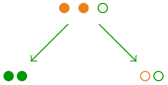
- $p \in \mathcal{P}\mathrm{os}_{\mathcal{F}}(\ell_2)$,
- mgu σ unifies $\ell_2|_p$ and ℓ_1, and
- if $p = \epsilon$ then $\ell_1 \to r_1$ and $\ell_2 \to r_2$ are not variants

$\ell_2\sigma[r_1\sigma]_p \approx r_2\sigma$ is critical pair, $\mathrm{CP}(\mathcal{R})$ are all critical pairs of $\mathcal{R}$.

Teaching Example: Simple Game

$\bullet\ \bullet \to \circ \qquad \circ\ \circ \to \circ \qquad \bullet\ \circ \to \bullet \qquad \circ\ \bullet \to \bullet$

critical pairs:



Lemma (Knuth and Bendix, 1970)

terminating TRS $\mathcal{R}$ is confluent if $s \downarrow_{\mathcal{R}} t$ for all critical pairs $s \approx t$ of $\mathcal{R}$

Definition (Critical Pair)

for $\ell_1 \to r_1$ and $\ell_2 \to r_2$ renamings of rules in $\mathcal{R}$ such that

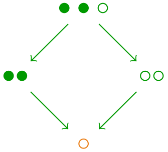
- $p \in \mathcal{P}os_{\mathcal{F}}(\ell_2)$,
- mgu σ unifies $\ell_2|_p$ and ℓ_1, and
- if $p = \epsilon$ then $\ell_1 \to r_1$ and $\ell_2 \to r_2$ are not variants

$\ell_2\sigma[r_1\sigma]_p \approx r_2\sigma$ is critical pair, $\mathrm{CP}(\mathcal{R})$ are all critical pairs of $\mathcal{R}$.

Teaching Example: Simple Game

$\bullet\ \bullet \to \circ$ $\qquad$ $\circ\ \circ \to \circ$ $\qquad$ $\bullet\ \circ \to \bullet$ $\qquad$ $\circ\ \bullet \to \bullet$

critical pairs:



Lemma (Knuth and Bendix, 1970)

terminating TRS $\mathcal{R}$ is confluent if $s \downarrow_{\mathcal{R}} t$ for all critical pairs $s \approx t$ of $\mathcal{R}$

Definition (Critical Pair)

for $\ell_1 \to r_1$ and $\ell_2 \to r_2$ renamings of rules in $\mathcal{R}$ such that

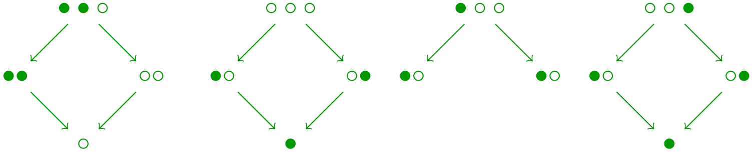
- $p \in \mathcal{P}\mathrm{os}_{\mathcal{F}}(\ell_2)$,
- mgu σ unifies $\ell_2|_p$ and ℓ_1, and
- if $p = \epsilon$ then $\ell_1 \to r_1$ and $\ell_2 \to r_2$ are not variants

$\ell_2\sigma[r_1\sigma]_p \approx r_2\sigma$ is critical pair, $\mathrm{CP}(\mathcal{R})$ are all critical pairs of $\mathcal{R}$.

Teaching Example: Simple Game

$\bullet\,\bullet \to \circ \qquad \circ\,\circ \to \circ \qquad \bullet\,\circ \to \bullet \qquad \circ\,\bullet \to \bullet$

critical pairs:



Lemma (Knuth and Bendix, 1970)

terminating TRS $\mathcal{R}$ is confluent if $s \downarrow_{\mathcal{R}} t$ for all critical pairs $s \approx t$ of $\mathcal{R}$

Definition (Critical Pair)

for $\ell_1 \to r_1$ and $\ell_2 \to r_2$ renamings of rules in $\mathcal{R}$ such that

- $p \in \mathcal{P}\mathrm{os}_{\mathcal{F}}(\ell_2)$,
- mgu σ unifies $\ell_2|_p$ and ℓ_1, and
- if $p = \epsilon$ then $\ell_1 \to r_1$ and $\ell_2 \to r_2$ are not variants

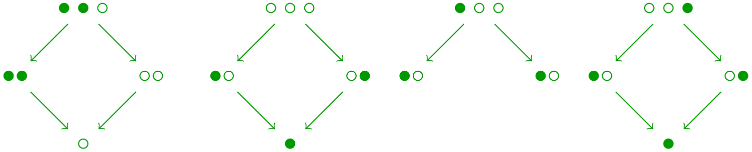
$\ell_2\sigma[r_1\sigma]_p \approx r_2\sigma$ is critical pair, $\mathrm{CP}(\mathcal{R})$ are all critical pairs of $\mathcal{R}$.

Teaching Example: Simple Game

system is confluent

● ● → ○ ○ ○ → ○ ● ○ → ● ○ ● → ●

critical pairs:



Lemma (Knuth and Bendix, 1970)

terminating TRS $\mathcal{R}$ is confluent if $s \downarrow_{\mathcal{R}} t$ for all critical pairs $s \approx t$ of $\mathcal{R}$

Definition (Critical Pair)

for $\ell_1 \to r_1$ and $\ell_2 \to r_2$ renamings of rules in $\mathcal{R}$ such that

- $p \in \mathcal{P}\mathrm{os}_{\mathcal{F}}(\ell_2)$,
- mgu σ unifies $\ell_2|_p$ and ℓ_1, and
- if $p = \epsilon$ then $\ell_1 \to r_1$ and $\ell_2 \to r_2$ are not variants

$\ell_2\sigma[r_1\sigma]_p \approx r_2\sigma$ is critical pair, $\mathrm{CP}(\mathcal{R})$ are all critical pairs of $\mathcal{R}$.

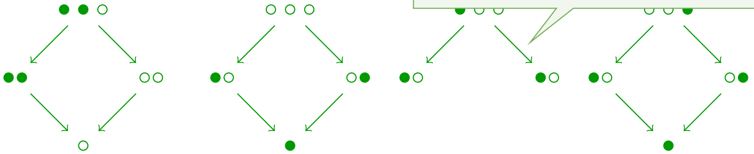
Teaching Example: Simple Game

system is confluent

● ● → ○ ○ ○ → ○ ● ○ → ● ○ ● → ●

initial example:
every strategy is winning strategy!

critical pairs:



Lemma (Knuth and Bendix, 1970)

terminating TRS $\mathcal{R}$ is confluent if $s \downarrow_{\mathcal{R}} t$ for all critical pairs $s \approx t$ of $\mathcal{R}$

Definition (Critical Pair)

for $\ell_1 \to r_1$ and $\ell_2 \to r_2$ renamings of rules in $\mathcal{R}$ such that

- $p \in \mathcal{P}\mathrm{os}_{\mathcal{F}}(\ell_2)$,
- mgu σ unifies $\ell_2|_p$ and ℓ_1, and
- if $p = \epsilon$ then $\ell_1 \to r_1$ and $\ell_2 \to r_2$ are not variants

$\ell_2\sigma[r_1\sigma]_p \approx r_2\sigma$ is critical pair, $\mathrm{CP}(\mathcal{R})$ are all critical pairs of $\mathcal{R}$.

CSI: Techniques

- decomposition techniques, e.g., layer systems
- transformation techniques, e.g. saturation
- criteria: Knuth-Bendix, orthogonality, Jouannaud-Kirchner, development closedness, ...
- support for higher-order systems
- criteria to establish unique normal form properties

CSI: Techniques

- decomposition techniques, e.g., layer systems
- transformation techniques, e.g. saturation
- criteria: Knuth-Bendix, orthogonality, Jouannaud-Kirchner, development closedness, ...
- support for higher-order systems
- criteria to establish unique normal form properties

Confluence Competition

- annual competition
- categories: standard TRS, string rewriting, commutation, conditional rewriting, higher-order rewriting, infeasibility, unique normal forms, normal form property, ground confluence, certified confluence
- http://project-coco.uibk.ac.at/

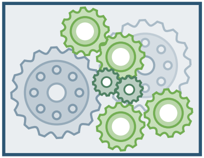
Knuth-Bendix Completion

KBCV

input: set of equations $\mathcal{E}$

output: terminating and confluent TRS $\mathcal{R}$ such that $\leftrightarrow_{\mathcal{E}}^{*} = \leftrightarrow_{\mathcal{R}}^{*}$

$\mathcal{E} \longrightarrow$



$\longrightarrow \mathcal{R}$

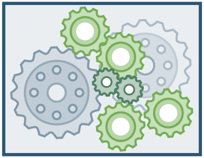
$\dashrightarrow$?

Knuth-Bendix Completion

KBCV

input: set of equations $\mathcal{E}$

output: terminating and confluent TRS $\mathcal{R}$ such that $\leftrightarrow^*_{\mathcal{E}} = \leftrightarrow^*_{\mathcal{R}}$

$\mathcal{E} \longrightarrow$  $\longrightarrow \mathcal{R}$ / $\dashrightarrow$?

Definition

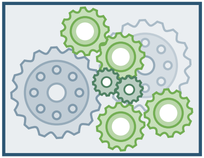
- TRS $\mathcal{R}$ is complete if terminating and confluent

KBCV

input: set of equations $\mathcal{E}$

output: terminating and confluent TRS $\mathcal{R}$ such that $\leftrightarrow^*_{\mathcal{E}} = \leftrightarrow^*_{\mathcal{R}}$

$\mathcal{E} \longrightarrow$



$\longrightarrow \mathcal{R}$

$\dashrightarrow$?

Definition

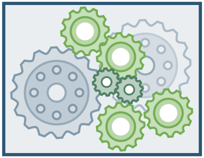
- TRS $\mathcal{R}$ is complete if terminating and confluent
- TRS $\mathcal{R}$ is presentation of set of equations $\mathcal{E}$ if $\leftrightarrow^*_{\mathcal{E}} = \leftrightarrow^*_{\mathcal{R}}$

Knuth-Bendix Completion

KBCV

input: set of equations $\mathcal{E}$

output: terminating and confluent TRS $\mathcal{R}$ such that $\leftrightarrow^*_\mathcal{E} = \leftrightarrow^*_\mathcal{R}$

$\mathcal{E} \longrightarrow$  $\longrightarrow \mathcal{R}$

$\dashrightarrow$?

Definition

- TRS $\mathcal{R}$ is complete if terminating and confluent
- TRS $\mathcal{R}$ is presentation of set of equations $\mathcal{E}$ if $\leftrightarrow^*_\mathcal{E} = \leftrightarrow^*_\mathcal{R}$

Fact

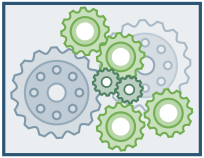
in complete TRS every term has unique normal form

Knuth-Bendix Completion

KBCV

input: set of equations $\mathcal{E}$

output: terminating and confluent TRS $\mathcal{R}$ such that $\leftrightarrow^*_{\mathcal{E}} = \leftrightarrow^*_{\mathcal{R}}$

$\mathcal{E}$ $\longrightarrow$  $\longrightarrow$ $\mathcal{R}$ / $\cdots\cdots\rightarrow$?

Example (Group Theory)

$$\mathcal{E}: \quad \begin{aligned} 1 \cdot x &\approx x \\ (-x) \cdot x &\approx 1 \\ x \cdot (y \cdot z) &\approx (x \cdot y) \cdot z \end{aligned}$$

$$\mathcal{R}: \quad \begin{aligned} -1 &\to 1 & -(-x) &\to x \\ 1 \cdot x &\to x & (-x) \cdot (x \cdot y) &\to y \\ x \cdot 1 &\to x & y \cdot ((-y) \cdot x) &\to x \\ (-x) \cdot x &\to 1 & x \cdot (y \cdot z) &\to (x \cdot y) \cdot z \\ x \cdot (-x) &\to 1 & -(x \cdot y) &\to -y \cdot (-x) \end{aligned}$$

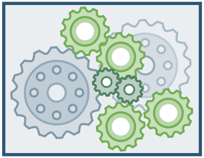
Knuth-Bendix Completion

KBCV

input: set of equations $\mathcal{E}$

output: terminating and confluent TRS $\mathcal{R}$ such that $\leftrightarrow^*_{\mathcal{E}} = \leftrightarrow^*_{\mathcal{R}}$

$\mathcal{E} \longrightarrow$



$\longrightarrow \mathcal{R}$

$\dashrightarrow$?

Example (Group Theory)

$$
\mathcal{E}: \quad \begin{aligned} 1 \cdot x &\approx x \\ (-x) \cdot x &\approx 1 \\ x \cdot (y \cdot z) &\approx (x \cdot y) \cdot z \end{aligned}
$$

$$
\mathcal{R}: \quad \begin{aligned} -1 &\to 1 & -(-x) &\to x \\ 1 \cdot x &\to x & (-x) \cdot (x \cdot y) &\to y \\ x \cdot 1 &\to x & y \cdot ((-y) \cdot x) &\to x \\ (-x) \cdot x &\to 1 & x \cdot (y \cdot z) &\to (x \cdot y) \cdot z \\ x \cdot (-x) &\to 1 & -(x \cdot y) &\to -y \cdot (-x) \end{aligned}
$$

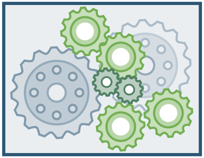
Knuth-Bendix Completion

KBCV, mkbtt, mædmax

input: set of equations $\mathcal{E}$

output: terminating and confluent TRS $\mathcal{R}$ such that $\leftrightarrow_{\mathcal{E}}^{*} = \leftrightarrow_{\mathcal{R}}^{*}$

$\mathcal{E} \longrightarrow$



$\longrightarrow \mathcal{R}$

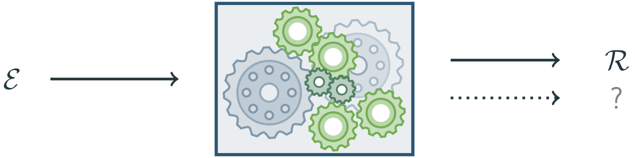
$\dashrightarrow$?

- KBCV: step-by-step completion and visualization

KBCV, mkbtt, mædmax

input: set of equations $\mathcal{E}$

output: terminating and confluent TRS $\mathcal{R}$ such that $\leftrightarrow^*_{\mathcal{E}} = \leftrightarrow^*_{\mathcal{R}}$



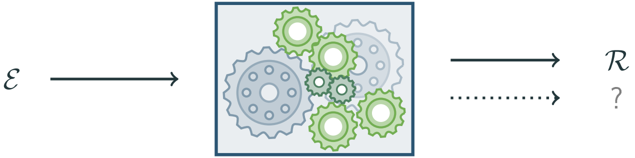
- KBCV: step-by-step completion and visualization
- mkbtt: automatic completion using termination tools

Knuth-Bendix Completion

KBCV, mkbtt, mædmax

input: set of equations $\mathcal{E}$

output: terminating and confluent TRS $\mathcal{R}$ such that $\leftrightarrow_{\mathcal{E}}^{*} = \leftrightarrow_{\mathcal{R}}^{*}$



- KBCV: step-by-step completion and visualization
- mkbtt: automatic completion using termination tools
- mædmax: equational theorem proving

Teaching Example: Cola Gene Puzzle (1)

Genetic engineers want to create cows that produce cola instead of milk. To that end they have to transform the DNA of the milk gene

TAGCTAGCTAGCT

in every fertilized egg into the cola gene

CTGACTGACT

Techniques exist to perform the following DNA transformations:

TCAT $\leftrightarrow$ T   GAG $\leftrightarrow$ AG   CTC $\leftrightarrow$ TC   AGTA $\leftrightarrow$ A   TAT $\leftrightarrow$ CT

Recently it has been discovered that the mad cow disease is caused by a retrovirus with the following DNA sequence

CTGCTACTGACT

What now, if accidentally cows with this virus are created? According to the engineers there is little risk because this never happened in their experiments, but various action groups demand absolute assurance.

Teaching Example: Cola Gene Puzzle (2)

equational system of known DNA transformations $\mathcal{E}$:

$\mathsf{TCAT} \approx \mathsf{T} \quad \mathsf{GAG} \approx \mathsf{AG} \quad \mathsf{CTC} \approx \mathsf{TC} \quad \mathsf{AGTA} \approx \mathsf{A} \quad \mathsf{TAT} \approx \mathsf{CT}$

Teaching Example: Cola Gene Puzzle (2)

equational system of known DNA transformations $\mathcal{E}$:

$\mathsf{TCAT} \approx \mathsf{T} \quad \mathsf{GAG} \approx \mathsf{AG} \quad \mathsf{CTC} \approx \mathsf{TC} \quad \mathsf{AGTA} \approx \mathsf{A} \qquad \mathsf{TAT} \approx \mathsf{CT}$

has complete presentation $\mathcal{R}$:

$\mathsf{GA} \to \mathsf{A} \quad \mathsf{AGT} \to \mathsf{AT} \quad \mathsf{ATA} \to \mathsf{A} \qquad \mathsf{TCA} \to \mathsf{TA} \quad \mathsf{TAT} \to \mathsf{T} \qquad \mathsf{CT} \to \mathsf{T}$

Teaching Example: Cola Gene Puzzle (2)

equational system of known DNA transformations $\mathcal{E}$:

$\mathsf{TCAT} \approx \mathsf{T} \quad \mathsf{GAG} \approx \mathsf{AG} \quad \mathsf{CTC} \approx \mathsf{TC} \quad \mathsf{AGTA} \approx \mathsf{A} \quad \mathsf{TAT} \approx \mathsf{CT}$

has complete presentation $\mathcal{R}$:

$\mathsf{GA} \to \mathsf{A} \quad \mathsf{AGT} \to \mathsf{AT} \quad \mathsf{ATA} \to \mathsf{A} \quad \mathsf{TCA} \to \mathsf{TA} \quad \mathsf{TAT} \to \mathsf{T} \quad \mathsf{CT} \to \mathsf{T}$

- (milk) $\mathsf{TAGCTAGCTAGCT} \overset{*}{\underset{\mathcal{E}}{\longleftrightarrow}} \mathsf{CTGACTGACT}$ (cola gene)

  $\mathsf{TAGCTAGCTAGCT} \overset{!}{\underset{\mathcal{R}}{\longrightarrow}} \mathsf{T} \overset{!}{\underset{\mathcal{R}}{\longleftarrow}} \mathsf{CTGACTGACT}$

Teaching Example: Cola Gene Puzzle (2)

equational system of known DNA transformations $\mathcal{E}$:

$\mathsf{TCAT} \approx \mathsf{T} \quad \mathsf{GAG} \approx \mathsf{AG} \quad \mathsf{CTC} \approx \mathsf{TC} \quad \mathsf{AGTA} \approx \mathsf{A} \quad \mathsf{TAT} \approx \mathsf{CT}$

has complete presentation $\mathcal{R}$:

$\mathsf{GA} \to \mathsf{A} \quad \mathsf{AGT} \to \mathsf{AT} \quad \mathsf{ATA} \to \mathsf{A} \quad \mathsf{TCA} \to \mathsf{TA} \quad \mathsf{TAT} \to \mathsf{T} \quad \mathsf{CT} \to \mathsf{T}$

- (milk) $\mathsf{TAGCTAGCTAGCT} \overset{*}{\underset{\mathcal{E}}{\longleftrightarrow}} \mathsf{CTGACTGACT}$ (cola gene)

  $\mathsf{TAGCTAGCTAGCT} \overset{!}{\underset{\mathcal{R}}{\longrightarrow}} \mathsf{T} \overset{!}{\underset{\mathcal{R}}{\longleftarrow}} \mathsf{CTGACTGACT}$

- (milk) $\mathsf{TAGCTAGCTAGCT} \overset{*}{\underset{\mathcal{E}}{\not\longleftrightarrow}} \mathsf{CTGCTACTGACT}$ (retrovirus)

  $\mathsf{TAGCTAGCTAGCT} \overset{!}{\underset{\mathcal{R}}{\longrightarrow}} \mathsf{T} \neq \mathsf{TGT} \overset{!}{\underset{\mathcal{R}}{\longleftarrow}} \mathsf{CTGCTACTGACT}$

Teaching Example: Cola Gene Puzzle (2)

equational system of known DNA transformations $\mathcal{E}$:

$\mathsf{TCAT} \approx \mathsf{T} \quad \mathsf{GAG} \approx \mathsf{AG} \quad \mathsf{CTC} \approx \mathsf{TC} \quad \mathsf{AGTA} \approx \mathsf{A} \quad \mathsf{TAT} \approx \mathsf{CT}$

has complete presentation $\mathcal{R}$:

$\mathsf{GA} \to \mathsf{A} \quad \mathsf{AGT} \to \mathsf{AT} \quad \mathsf{ATA} \to \mathsf{A} \quad \mathsf{TCA} \to \mathsf{TA} \quad \mathsf{TAT} \to \mathsf{T} \quad \mathsf{CT} \to \mathsf{T}$

- (milk) $\mathsf{TAGCTAGCTAGCT} \xleftrightarrow[\mathcal{E}]{*} \mathsf{CTGACTGACT}$ (cola gene)

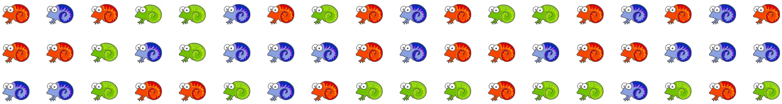
  $$\mathsf{TAGCTAGCTAGCT} \xrightarrow[\mathcal{R}]{!} \mathsf{T} \xleftarrow[\mathcal{R}]{!} \mathsf{CTGACTGACT}$$

- (milk) $\mathsf{TAGCTAGCTAGCT} \not\xleftrightarrow[\mathcal{E}]{*} \mathsf{CTGCTACTGACT}$ (retrovirus)

  $$\mathsf{TAGCTAGCTAGCT} \xrightarrow[\mathcal{R}]{!} \mathsf{T} \neq \mathsf{TGT} \xleftarrow[\mathcal{R}]{!} \mathsf{CTGCTACTGACT}$$

▸ KBCV

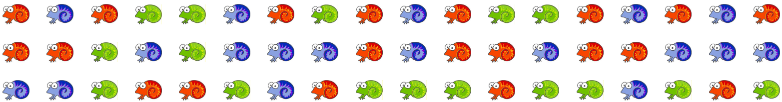
Example (Chameleon Puzzle)



A colony of chameleons consists of 20 red, 18 blue, and 16 green animals. Whenever two of different color meet, both change to the third color.

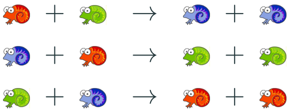
Is it possible that all 54 chameleons become the same color?

Example (Chameleon Puzzle)

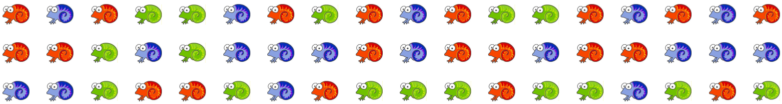


A colony of chameleons consists of 20 red, 18 blue, and 16 green animals. Whenever two of different color meet, both change to the third color.

Is it possible that all 54 chameleons become the same color?

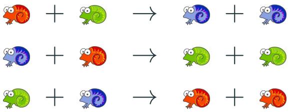


Example (Chameleon Puzzle)



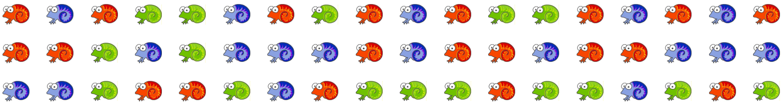
A colony of chameleons consists of 20 red, 18 blue, and 16 green animals. Whenever two of different color meet, both change to the third color.

Is it possible that all 54 chameleons become the same color?



$$
\begin{aligned}
x + y &\approx y + x \\
(x + y) + z &\approx x + (y + z)
\end{aligned}
$$

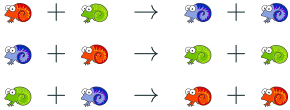
Example (Chameleon Puzzle)



A colony of chameleons consists of 20 red, 18 blue, and 16 green animals. Whenever two of different color meet, both change to the third color.

Is it possible that all 54 chameleons become the same color?

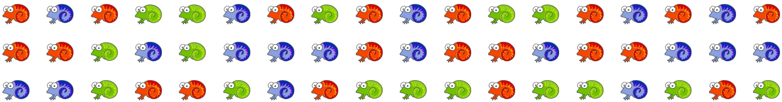
mædmax produces TRS which is complete (modulo AC):



$$x + y \approx y + x$$

$$(x + y) + z \approx x + (y + z)$$

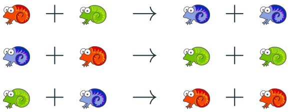
Example (Chameleon Puzzle)



A colony of chameleons consists of 20 red, 18 blue, and 16 green animals. Whenever two of different color meet, both change to the third color.

Is it possible that all 54 chameleons become the same color?

mædmax produces TRS which is complete (modulo AC):

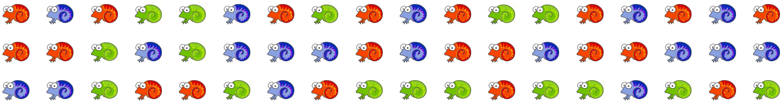


$$x + y \approx y + x$$

$$(x + y) + z \approx x + (y + z)$$

- initial colony: 20  + 18  + 16 

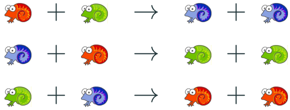
Example (Chameleon Puzzle)



A colony of chameleons consists of 20 red, 18 blue, and 16 green animals. Whenever two of different color meet, both change to the third color.

Is it possible that all 54 chameleons become the same color?

mædmax produces TRS which is complete (modulo AC):

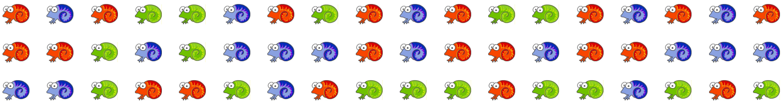


$$x + y \approx y + x$$

$$(x + y) + z \approx x + (y + z)$$

- initial colony: 20  $+$ 18  $+$ 16  $\to^!$ 52  $+$ 2 

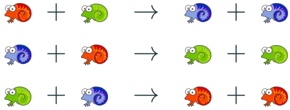
Example (Chameleon Puzzle)



A colony of chameleons consists of 20 red, 18 blue, and 16 green animals. Whenever two of different color meet, both change to the third color.

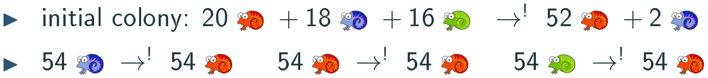
Is it possible that all 54 chameleons become the same color?

mædmax produces TRS which is complete (modulo AC):



$$x + y \approx y + x$$

$$(x + y) + z \approx x + (y + z)$$



Maximal Ordered Completion

$$
\begin{tikzcd}
\mathcal{R} \subseteq \mathcal{E} \cup \mathcal{E}^{-1} \arrow[dr, bend left, "\mathcal{R}"] & \\
& \text{success?} \arrow[dl, bend left, "\text{no}"] \\
\mathcal{E} := \mathcal{E} \cup S_{\mathcal{E}} \arrow[uu, bend left, "\mathcal{E}"] &
\end{tikzcd}
$$

Klein and Hirokawa, Maximal Completion, 2011.

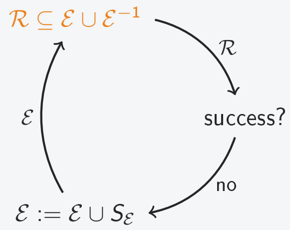
Maximal Ordered Completion

$$\begin{tikzcd}
\mathcal{R} \subseteq \mathcal{E} \cup \mathcal{E}^{-1} \arrow[dr, bend left, "\mathcal{R}"] & \\
& \text{success?} \arrow[dl, bend left, "\text{no}"] \\
\mathcal{E} := \mathcal{E} \cup S_{\mathcal{E}} \arrow[uu, bend left, "\mathcal{E}"] &
\end{tikzcd}$$

Procedure (equations only)

0. initialize equations $\mathcal{E}$ to input equalities $\mathcal{E}_0$

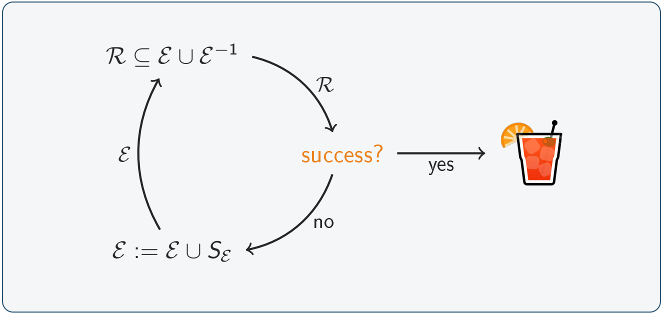
Maximal Ordered Completion



Procedure (equations only)

0. initialize equations $\mathcal{E}$ to input equalities $\mathcal{E}_0$
1. guess terminating rewrite system $\mathcal{R}$ from $\mathcal{E}$

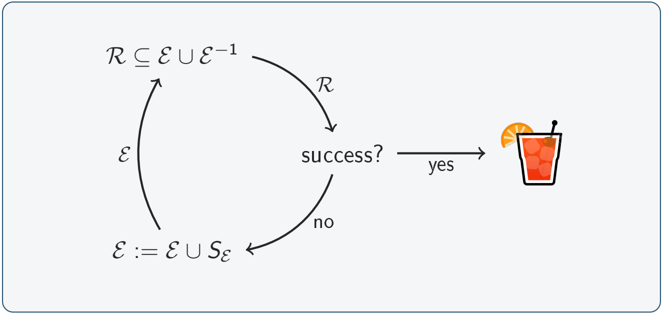
Maximal Ordered Completion



Procedure (equations only)

0. initialize equations $\mathcal{E}$ to input equalities $\mathcal{E}_0$
1. guess terminating rewrite system $\mathcal{R}$ from $\mathcal{E}$
2. check whether ground complete

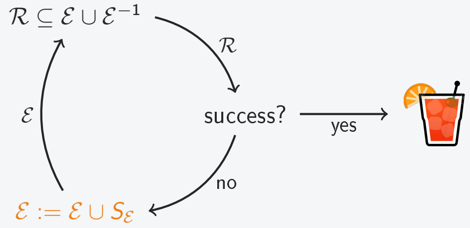
Maximal Ordered Completion



Procedure (equations only)

0. initialize equations $\mathcal{E}$ to input equalities $\mathcal{E}_0$
1. guess terminating rewrite system $\mathcal{R}$ from $\mathcal{E}$
2. check whether ground complete

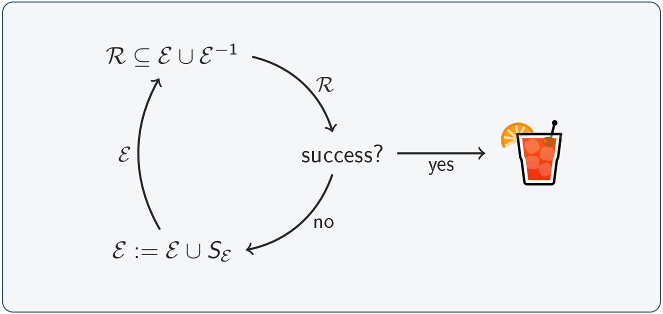
Maximal Ordered Completion



Procedure (equations only)

0. initialize equations $\mathcal{E}$ to input equalities $\mathcal{E}_0$
1. guess terminating rewrite system $\mathcal{R}$ from $\mathcal{E}$
2. check whether ground complete
3. add some critical pairs $S_{\mathcal{E}} \subseteq \mathrm{CP}(\mathcal{R} \cup \mathcal{E})$

Maximal Ordered Completion



Procedure (equations only)

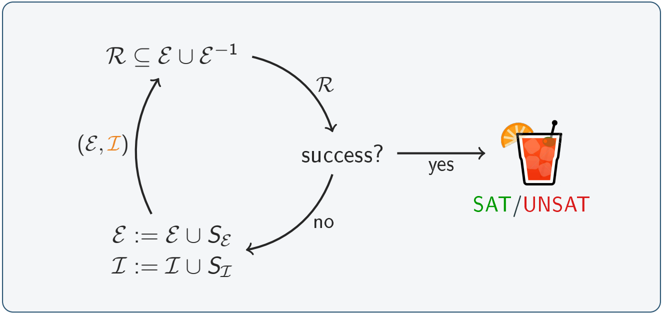
0 initialize equations $\mathcal{E}$ to input equalities $\mathcal{E}_0$

1 guess terminating rewrite system $\mathcal{R}$ from $\mathcal{E}$

2 check whether ground complete

3 add some critical pairs $S_{\mathcal{E}} \subseteq \mathrm{CP}(\mathcal{R} \cup \mathcal{E})$,
repeat from 1

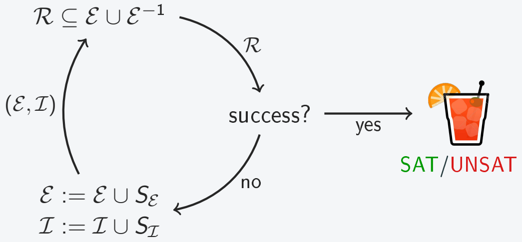
Maximal Ordered Completion



Procedure (with inequalities)

0. initialize equations and inequalities $(\mathcal{E}, \mathcal{I})$ to input $(\mathcal{E}_0, \{s \not\approx t\})$
1. guess terminating rewrite system $\mathcal{R}$ from $\mathcal{E}$
2. check whether ground complete or inequality joinable
3. add some critical pairs $S_{\mathcal{E}} \subseteq \mathrm{CP}(\mathcal{R} \cup \mathcal{E})$ and $S_{\mathcal{I}} \subseteq \mathrm{CP}(\mathcal{R} \cup \mathcal{E}, \mathcal{I})$, repeat from 1

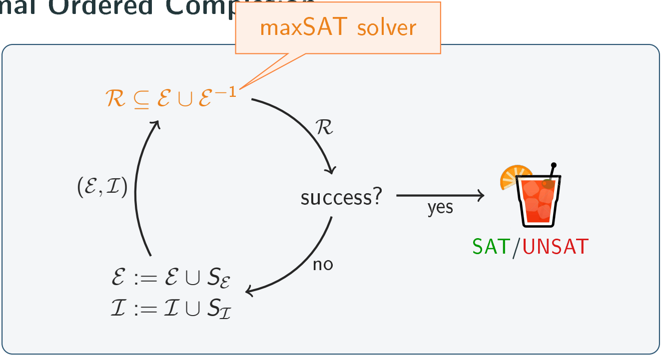
Maximal Ordered Completion



Remark

maximal ordered completion is conflict-driven approach

Maximal Ordered Completion

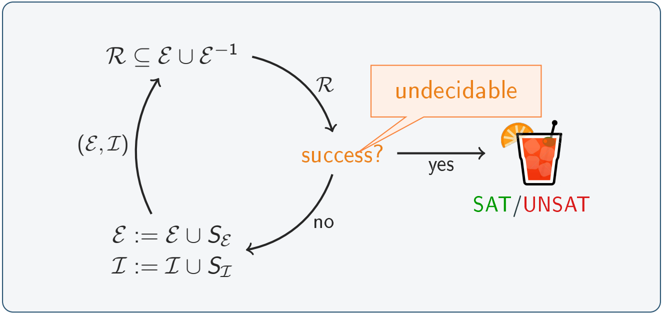


Critical Aspects

- use maxSAT solver to find rewrite system maximizing some goal, e.g. to orient as many equations as possible

Winkler, A Ground Joinability Criterion for Ordered Completion, 2017.

Maximal Ordered Completion



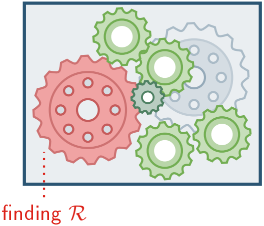
Critical Aspects

- use maxSAT solver to find rewrite system maximizing some goal, e.g. to orient as many equations as possible
- checking ground completeness is undecidable: overapproximate

---

Winkler, **A Ground Joinability Criterion for Ordered Completion**, 2017.

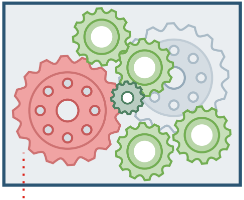
A Glimpse into the Guts



Finding Rewrite Systems

- use SMT encodings of orders: LPO and KBO, linear polynomials

A Glimpse into the Guts



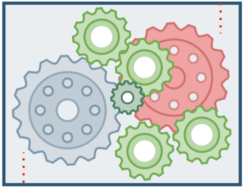
Finding Rewrite Systems

- use SMT encodings of orders: LPO and KBO, linear polynomials
- optimization for maxSMT uses weighted combination of

  (a) maximize $\mathcal{R}$-reducible subset of $\mathcal{E}$ (b) maximize $|\mathcal{R}|$

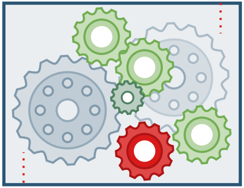
  (c) maximize ground-joinable equations in $\mathcal{E}$ (d) minimize $|\mathrm{CP}(\mathcal{R})|$

A Glimpse into the Guts



Success Checks

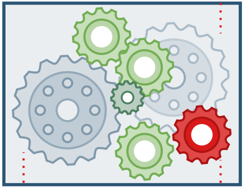
- rewriting/unifiability for goals
- narrowing for ground complete systems
- ground confluence criteria [MartinNipkow90], [W17] (SAT problem)



Avoiding the Blowup

- compute extended critical pairs wrt order orienting $\mathcal{R}$
- filter out equations known to be ground joinable [AHL03]

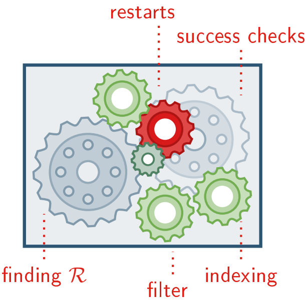
A Glimpse into the Guts



Fingerprint Indexing

- for matching and unifiability [Schulz12]

A Glimpse into the Guts

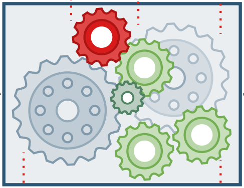


Restarts

- do restarts keeping small lemmas when state is stuck

A Glimpse into the Guts

$$0 + x \approx x$$
$$(-x) + x \approx 0$$
$$x + (y + z) \approx (x + y) + z$$
$$x + y \approx y + x$$
$$x \cdot (y \cdot z) \approx (x \cdot y) \cdot z$$
$$(x + y) \cdot z \approx (x \cdot z) + (y \cdot z)$$
$$\mathsf{a} \cdot 0 \not\approx 0$$



<certificationProblem> <input>
<orderedCompletionInput>
<equations> <rules> ...</rules>
</equations> <trs> <rules>
<rule> <lhs> <funapp>
<name>mult</name>...<arg>
<funapp> <name>inv</name> <arg>
<var>Y</var> </arg> </funapp>
</arg> </funapp> </lhs> <rhs>
<funapp> <name>mult</name>
<arg> <var>X</var> </arg>
<arg> <var>Y</var> </arg>
</funapp> </rhs> </rule> </trs>
<proof> <orderedCompletionProof>
</orderedCompletionProof>
</proof> </certificationProblem>

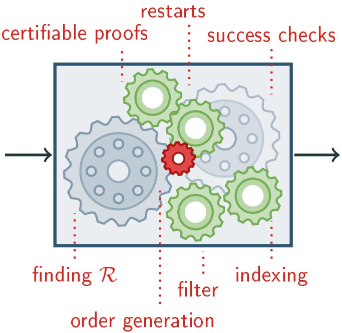
Certifiable Proofs

- support CPF output for unsatisfiable and satisfiable problems
- 90% of proofs validated by Isabelle-based certifier CeTA [ST15] (not all ground confluence criteria are supported by CeTA yet)

A Glimpse into the Guts

$$
\begin{aligned}
0 + x &\approx x \\
(-x) + x &\approx 0 \\
x + (y + z) &\approx (x + y) + z \\
x + y &\approx y + x \\
x \cdot (y \cdot z) &\approx (x \cdot y) \cdot z \\
(x + y) \cdot z &\approx (x \cdot z) + (y \cdot z) \\
\mathsf{a} \cdot 0 &\not\approx 0
\end{aligned}
$$

$\longrightarrow$



$\longrightarrow$

LPO

$\cdot > - > + > 0$

Order Generation Mode

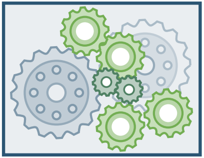
- output "best" order found after some iterations

Complexity

TcT

input: term rewrite system $\mathcal{R}$

output: derivational complexity $\mathsf{dc}_{\mathcal{R}}$

$\mathcal{R} \longrightarrow$  $\longrightarrow \mathcal{O}(n^k)$

$\dashrightarrow$?

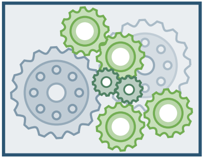
Avanzini, Moser, and Schaper, **TcT: Tyrolean Complexity Tool**, 2016.

Complexity

T_CT

input: term rewrite system $\mathcal{R}$

output: derivational complexity $\mathrm{dc}_{\mathcal{R}}$

$\mathcal{R} \longrightarrow$  $\longrightarrow \mathcal{O}(n^k)$ / $\dashrightarrow$?

Definitions

- derivation height $\qquad \mathrm{dh}_{\mathcal{R}}(t) = \max\{\, n \mid \exists\, u \colon t \to_{\mathcal{R}}^{n} u \,\}$

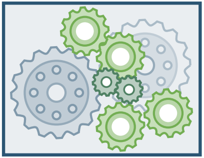
Complexity

T

input: term rewrite system $\mathcal{R}$

output: derivational complexity $\mathrm{dc}_{\mathcal{R}}$

$\mathcal{R} \longrightarrow$



$\longrightarrow \mathcal{O}(n^k)$

$\dashrightarrow$?

Definitions

- derivation height $\quad \mathrm{dh}_{\mathcal{R}}(t) = \max \{ n \mid \exists\, u \colon t \to_{\mathcal{R}}^{n} u \}$
- derivational complexity $\quad \mathrm{dc}_{\mathcal{R}}(n) = \max \{ \mathrm{dh}_{\mathcal{R}}(t) \mid |t| = n \}$

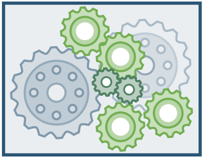
Complexity

TcT

input: term rewrite system $\mathcal{R}$

output: derivational complexity $\mathrm{dc}_{\mathcal{R}}$

$\mathcal{R} \longrightarrow$



$\longrightarrow \mathcal{O}(n^k)$

$\dashrightarrow$?

Example

$$[] \mathbin{@} xs \to xs \qquad (x : xs) \mathbin{@} ys \to x : (xs \mathbin{@} ys)$$

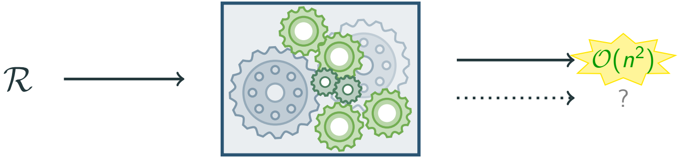
$$\mathsf{flatten}([]) \to [] \qquad \mathsf{flatten}(x : xs) \to x \mathbin{@} \mathsf{flatten}(xs)$$

Complexity

TcT

input: term rewrite system $\mathcal{R}$

output: derivational complexity $\mathsf{dc}_{\mathcal{R}}$



Example

$$
\begin{aligned}
[\,] \mathbin{@} xs &\to xs & \qquad (x : xs) \mathbin{@} ys &\to x : (xs \mathbin{@} ys) \\
\mathsf{flatten}([\,]) &\to [\,] & \qquad \mathsf{flatten}(x : xs) &\to x \mathbin{@} \mathsf{flatten}(xs)
\end{aligned}
$$

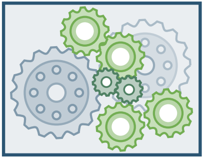
Complexity

TcT

input: term rewrite system $\mathcal{R}$

output: derivational complexity $\mathrm{dc}_{\mathcal{R}}$

$\mathcal{R} \longrightarrow$



$\longrightarrow \mathcal{O}(n^2)$

$\dashrightarrow$?

Example (Sieve of Eratosthenes)

- system is innermost terminating
- can analyze innermost complexity

▸ TcT

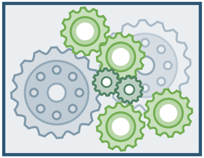
Proof Terms

ProTeM

tool to create proof terms from rewrite steps and manipulate them

$\mathcal{R} \longrightarrow$

$\xrightarrow{*} \longrightarrow$

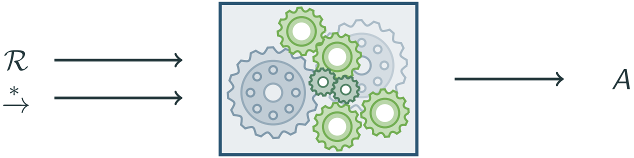


$\longrightarrow A$

Proof Terms

ProTeM

tool to create proof terms from rewrite steps and manipulate them



Proof Terms

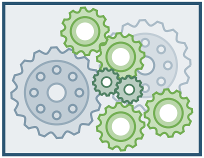
- representation of rewrite sequences as terms

ProTeM

tool to create proof terms from rewrite steps and manipulate them

$\mathcal{R} \longrightarrow$

$\xrightarrow{*} \longrightarrow$



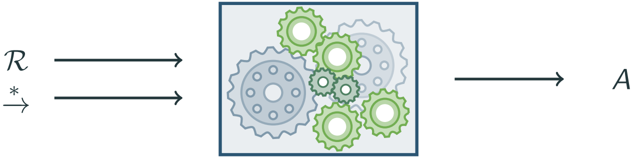
$\longrightarrow A$

Proof Terms

- representation of rewrite sequences as terms
- admit concise analysis of equivalence of rewrite sequences

ProTeM

tool to create proof terms from rewrite steps and manipulate them



Proof Terms

- representation of rewrite sequences as terms
- admit concise analysis of equivalence of rewrite sequences

August 28 @ CADE: Presentation by Christina Kohl

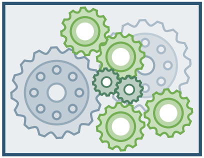
composition of proof terms and implementation in ProTeM

ProTeM

tool to create proof terms from rewrite steps and manipulate them

$\mathcal{R}$

$\xrightarrow{*}$



A

Proof Terms

- representation of rewrite sequences as terms
- admit concise analysis of equivalence of rewrite sequences

August 28 @ CADE: Presentation by Christina Kohl

composition of proof terms and implementation in ProTeM

Outline

Motivating Examples

Term Rewriting

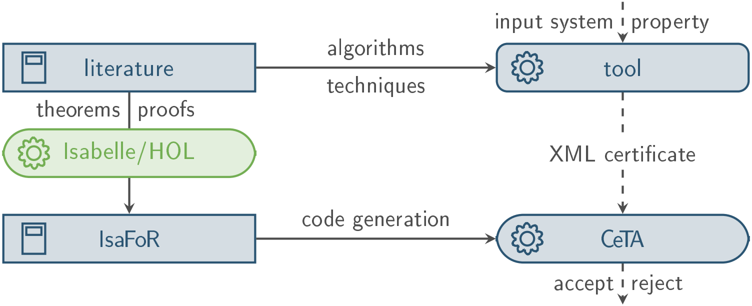
Tools

Formalization and Certification

Conclusion

why trust these tools?

The IsaFoR/CeTA Framework



Thiemann and Sternagel, **Certification of Termination Proofs Using CeTA**, 2009.

Certification of Tool Output: Overview

- $\mathsf{T_TT_2}$

standard TRS:	829 YES	(750)	200 NO	(193)
standard SRS:	733 YES	(670)	43 NO	(24)

- CSI

standard TRS:	42 YES	(28)	33 NO	(23)

- $\mathsf{T_CT}$

TPDB:	203 YES	(165)

- KBCV

completion systems:	89 YES	(89)

- mædmax

TPTP:	112 SAT	(69)	621 UNSAT	(612)

Data taken from Termination and Confluence Competitions 2019, [AST15], [WM18], [SWZ15].

Outline

Conclusion

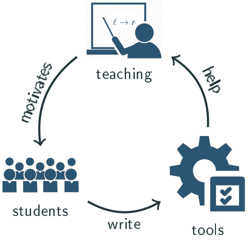
Summary

- rewrite tools for different theorem proving tasks
  - termination analysis
  - Knuth-Bendix completion
  - confluence analysis
  - complexity analysis

Conclusion

Summary

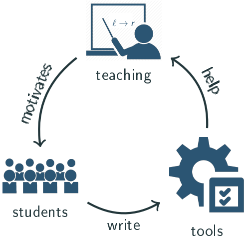
- rewrite tools for different theorem proving tasks
  - termination analysis
  - Knuth-Bendix completion
  - confluence analysis
  - complexity analysis



Conclusion

Summary

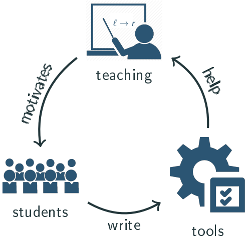
- rewrite tools for different theorem proving tasks
  - termination analysis
  - Knuth-Bendix completion
  - confluence analysis
  - complexity analysis
- certification framework: Isabelle library IsaFoR and proof checker CeTA



Conclusion

Summary

- rewrite tools for different theorem proving tasks
  - termination analysis
  - Knuth-Bendix completion
  - confluence analysis
  - complexity analysis
- certification framework: Isabelle library IsaFoR and proof checker CeTA



Tool Features Helpful for Students

- web interfaces
- control over many options

Acknowledgements

Aart Middeldorp, Georg Moser, René Thiemann, Harald Zankl, Christian Sternagel, Martin Korp, Friedrich Neurauter, Andreas Schnabl, Martin Avanzini, Michael Schaper, Thomas Sternagel, Julian Nagele, Bertram Felgenhauer, Cynthia Kop, Manuel Schneckenreither, David Obwaller, Sebastiaan Joosten, Akihisa Yamada, Ralph Bottesch, T. V. H. Prathamesh, Franziska Rapp, Maria Schett, Max Haslbeck, Simon Legner, Christina Kohl, Alexander Lochmann, Jonas Schöpf