$\mathsf{T_TT_2}$ with Termination Templates for Teaching

Jonas Schöpf and Christian Sternagel

Department of Computer Science
University of Innsbruck

16th International Workshop on Termination
Oxford

July 19, 2018

Motivation

$$\begin{aligned} \mathsf{f}(x, \mathsf{g}(x)) &\to \mathsf{g}(\mathsf{f}(x, x)) \\ \mathsf{f}(x, x) &\to \mathsf{h}(\mathsf{g}(x)) \\ \mathsf{g}(\mathsf{f}(x, x)) &\to \mathsf{f}(\mathsf{h}(x), x) \end{aligned}$$

Motivation

$$\begin{aligned} \mathsf{f}(x, \mathsf{g}(x)) &\to \mathsf{g}(\mathsf{f}(x, x)) \\ \mathsf{f}(x, x) &\to \mathsf{h}(\mathsf{g}(x)) \\ \mathsf{g}(\mathsf{f}(x, x)) &\to \mathsf{f}(\mathsf{h}(x), x) \end{aligned}$$

Correct Termination Proofs?

Motivation

$$\mathsf{f}(x, \mathsf{g}(x)) \to \mathsf{g}(\mathsf{f}(x, x))$$
$$\mathsf{f}(x, x) \to \mathsf{h}(\mathsf{g}(x))$$
$$\mathsf{g}(\mathsf{f}(x, x)) \to \mathsf{f}(\mathsf{h}(x), x)$$

Correct Termination Proofs?

student 1:

$$w_0 = 1$$
$$\mathsf{w}(\mathsf{f}) = 4, \mathsf{w}(\mathsf{g}) = 2, \mathsf{w}(\mathsf{h}) = 1$$
$$\mathsf{h} > \mathsf{f} > \mathsf{g}$$

Motivation

$$\mathsf{f}(x, \mathsf{g}(x)) \to \mathsf{g}(\mathsf{f}(x, x))$$
$$\mathsf{f}(x, x) \to \mathsf{h}(\mathsf{g}(x))$$
$$\mathsf{g}(\mathsf{f}(x, x)) \to \mathsf{f}(\mathsf{h}(x), x)$$

Correct Termination Proofs?

student 1:

$$w_0 = 1$$
$$\mathsf{w}(\mathsf{f}) = 4, \mathsf{w}(\mathsf{g}) = 2, \mathsf{w}(\mathsf{h}) = 1$$
$$\mathsf{h} > \mathsf{f} > \mathsf{g}$$

student 2:

$$[\mathsf{h}](x) = x$$
$$[\mathsf{f}](x, y) = 8x + 24y + 16$$
$$[\mathsf{g}](x) = x + 1$$

Motivation

$$\mathsf{f}(x, \mathsf{g}(x)) \to \mathsf{g}(\mathsf{f}(x, x))$$
$$\mathsf{f}(x, x) \to \mathsf{h}(\mathsf{g}(x))$$
$$\mathsf{g}(\mathsf{f}(x, x)) \to \mathsf{f}(\mathsf{h}(x), x)$$

Correct Termination Proofs?

student 1:

$$w_0 = 1$$
$$\mathsf{w}(\mathsf{f}) = 4, \mathsf{w}(\mathsf{g}) = 2, \mathsf{w}(\mathsf{h}) = 1$$
$$\mathsf{h} > \mathsf{f} > \mathsf{g}$$

student 2:

$$[\mathsf{h}](x) = x$$
$$[\mathsf{f}](x, y) = 8x + 24y + 16$$
$$[\mathsf{g}](x) = x + 1$$

student 3:

$$[\mathsf{h}](x) = \begin{pmatrix} 1 & 4 \\ 0 & 0 \end{pmatrix} x, \ [\mathsf{f}](x, y) = \begin{pmatrix} 1 & 15 \\ 0 & 0 \end{pmatrix} x + \begin{pmatrix} 16 & 1 \\ 0 & 0 \end{pmatrix} y + \begin{pmatrix} 8 \\ 0 \end{pmatrix},$$
$$[\mathsf{g}](x) = \begin{pmatrix} 1 & 0 \\ 0 & 4 \end{pmatrix} x + \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

Overview

- main idea
- $\mathsf{T_TT_2}$

Overview

- main idea
- $\mathsf{T_TT_2}$
- template mechanism
- URL encoding via the web interface

Main Idea

- check specific proofs with $\mathsf{T_TT_2}$

Main Idea

- check specific proofs with $\mathsf{T_TT_2}$
- modify proof construction

Main Idea

- check specific proofs with $\mathsf{T_TT_2}$
- modify proof construction
- provide a mechanism to show termination examples

Main Idea

- check specific proofs with $\mathsf{T_TT_2}$
- modify proof construction
- provide a mechanism to show termination examples
- in general: make $\mathsf{T_TT_2}$ more useful for teaching

Main Idea

- check specific proofs with $\mathsf{T_TT_2}$
- modify proof construction
- provide a mechanism to show termination examples
- in general: make $\mathsf{T_TT_2}$ more useful for teaching

Solve following Questions

- is the following termination proof correct?
- exists another termination proof with specific parameters?
- ...

Main Idea: Templates

- based on SAT/SMT encodings

Main Idea: Templates

- based on SAT/SMT encodings
- concrete instances seem entirely random

Main Idea: Templates

- based on SAT/SMT encodings
- concrete instances seem entirely random
- provide method-specific template mechanism to modify proof construction

Main Idea: Templates

- based on SAT/SMT encodings
- concrete instances seem entirely random
- provide method-specific template mechanism to modify proof construction

Main Idea: URL encoded examples

- setting up examples during lecture is error-prone and tedious

Main Idea: Templates

- based on SAT/SMT encodings
- concrete instances seem entirely random
- provide method-specific template mechanism to modify proof construction

Main Idea: URL encoded examples

- setting up examples during lecture is error-prone and tedious
- encode examples into URL

Main Idea: Templates

- based on SAT/SMT encodings
- concrete instances seem entirely random
- provide method-specific template mechanism to modify proof construction

Main Idea: URL encoded examples

- setting up examples during lecture is error-prone and tedious
- encode examples into URL
- automatically restore examples

Main Idea: Templates

- based on SAT/SMT encodings
- concrete instances seem entirely random
- provide method-specific template mechanism to modify proof construction

Main Idea: URL encoded examples

- setting up examples during lecture is error-prone and tedious
- encode examples into URL
- automatically restore examples
- derive a convenient way to present examples

Template Mechanism for LPO, KBO, PIs and MIs

Addition on Natural Numbers

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

Addition on Natural Numbers

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

Termination Proof without restrictions by $\mathsf{T_TT_2}$

LPO:

$$+ > \mathsf{s} \sim 0$$

Addition on Natural Numbers

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

Termination Proof without restrictions by $\mathsf{T_TT_2}$

LPO:

$$+ > \mathsf{s} \sim 0$$

Also correct?

Is there a precedence with

$0 > +$, $\mathsf{s} > +$ or $0 = +$

How to call $\mathsf{T_TT_2}$

```
./ttt2 [options] <file> [timeout]
```

How to call $\mathsf{T_TT_2}$

```
./ttt2 [options] <file> [timeout]
```

Call with specific Strategy

```
./ttt2 -s 'matrix' <file> [timeout]
```

How to call $\mathsf{T_TT_2}$

```
./ttt2 [options] <file> [timeout]
```

Call with specific Strategy

```
./ttt2 -s 'matrix' <file> [timeout]
```

Call with a Template

```
./ttt2 -s 'poly [processor flags]' <file> [timeout]
```

How to call $\mathsf{T_TT_2}$

```
./ttt2 [options] <file> [timeout]
```

Call with specific Strategy

```
./ttt2 -s 'matrix' <file> [timeout]
```

Call with a Template

```
./ttt2 -s 'poly [processor flags]' <file> [timeout]
```

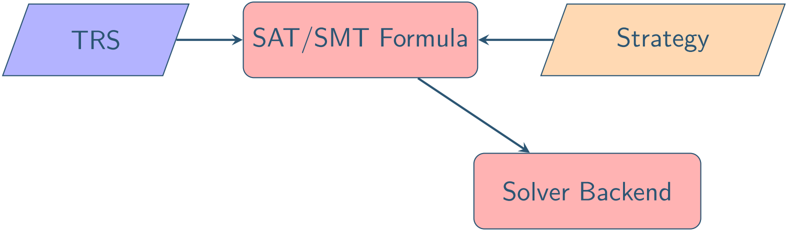
Example: Full Call

```
./ttt2 -s 'poly -inters "+ = x0 + x1"' add.trs
```

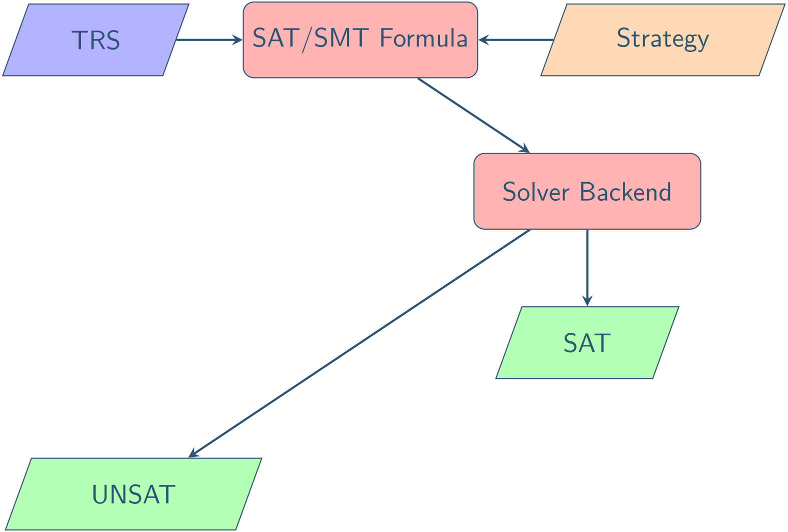
Encoding in $\mathsf{T_TT_2}$

Encoding in $\mathsf{T_TT_2}$

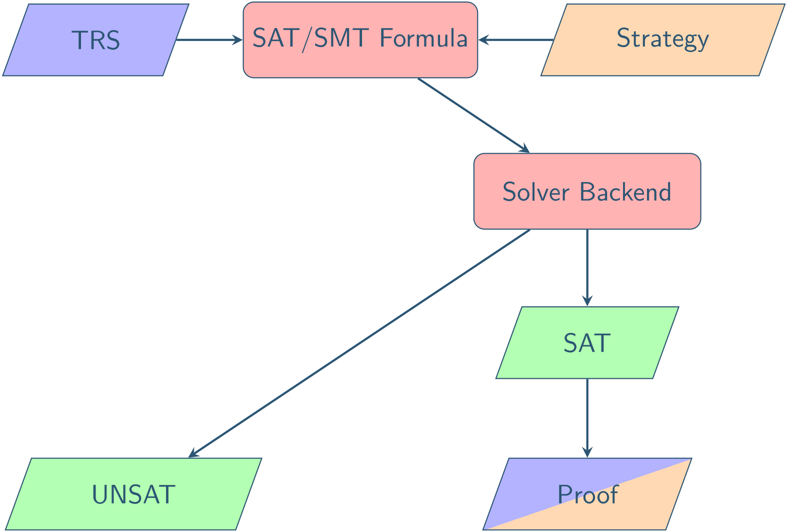
Encoding in $\mathsf{T_TT_2}$



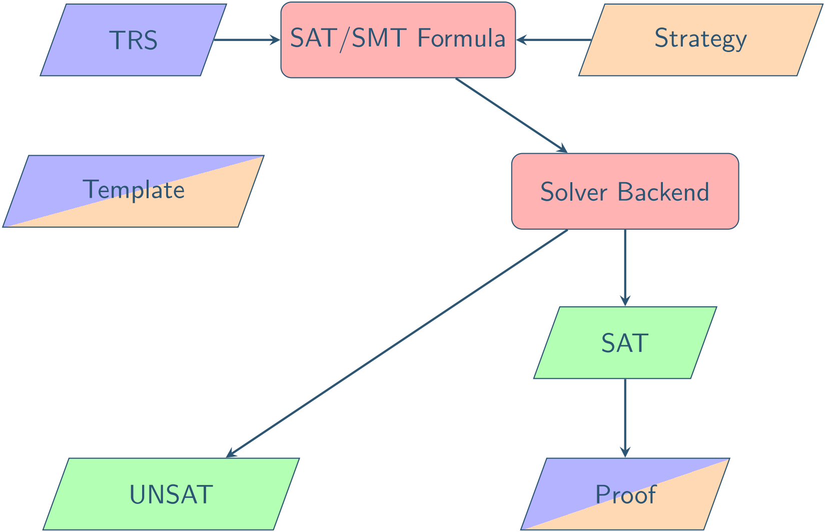
Encoding in $\mathsf{T_TT_2}$



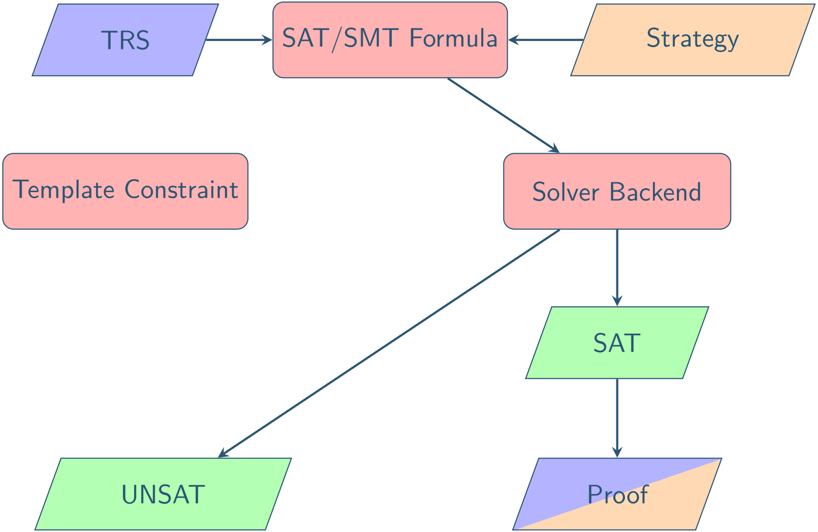
Encoding in $\mathsf{T_TT_2}$



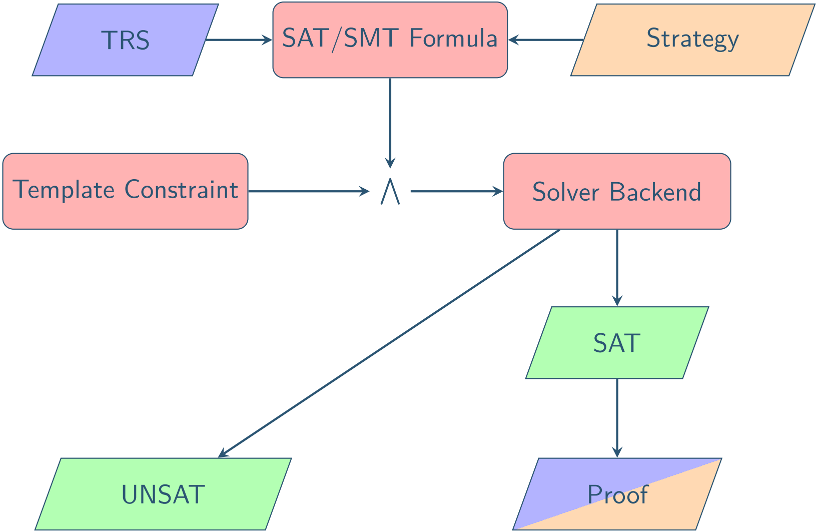
Encoding in $\mathsf{T_TT_2}$



Encoding in $\mathsf{T_TT_2}$



Encoding in $\mathsf{T_TT_2}$



Lexicographic Path Order

- parameter is precedence

Lexicographic Path Order

- parameter is precedence
- `-prec` gets prec template

Lexicographic Path Order

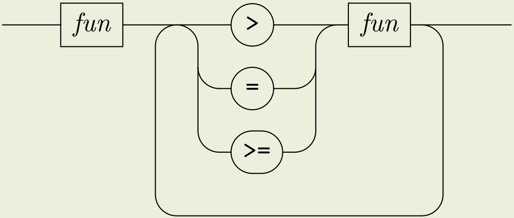
- parameter is precedence
- `-prec` gets prec template
- prec template specifies a (partial) precedence

Lexicographic Path Order

- parameter is precedence
- `-prec` gets prec template
- prec template specifies a (partial) precedence

Syntax

prec



Addition on Natural Numbers

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

Termination Proof without restrictions by $\mathsf{T_TT_2}$

precedence: $+ > \mathsf{s} \sim 0$

Also correct?

Is there a precedence with

$0 > +$, $\mathsf{s} > +$ or $0 = +$

[0]tick, cross and question mark are clickable hyperlinks

Addition on Natural Numbers

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

Termination Proof without restrictions by $\mathsf{T_TT_2}$

precedence: $+ > \mathsf{s} \sim 0$

Also correct?

$0 > +$, $\mathsf{s} > +$ or $0 = +$

[0]tick, cross and question mark are clickable hyperlinks

Addition on Natural Numbers

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

Termination Proof without restrictions by $\mathsf{T_TT_2}$

precedence: $+ > \mathsf{s} \sim 0$

Also correct?

$0 > +$, $\mathsf{s} > +$ or $0 = +$

```
./ttt2 -s 'lpo -prec"0 > +"' add.trs
```
✓

[0] tick, cross and question mark are clickable hyperlinks

Addition on Natural Numbers

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

Termination Proof without restrictions by $\mathsf{T_TT_2}$

precedence: $+ > \mathsf{s} \sim 0$

Also correct?

$0 > +$, $\mathsf{s} > +$ or $0 = +$

```
./ttt2 -s 'lpo -prec"0 > +"' add.trs            ✓
./ttt2 -s 'lpo -prec"s > +"' add.trs            ?
```

[0] tick, cross and question mark are clickable hyperlinks

Addition on Natural Numbers

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

Termination Proof without restrictions by $\mathsf{T_TT_2}$

precedence: $+ > \mathsf{s} \sim 0$

Also correct?

$0 > +$, $\mathsf{s} > +$ or $0 = +$

```
./ttt2 -s 'lpo -prec"0 > +"' add.trs      ✓
./ttt2 -s 'lpo -prec"s > +"' add.trs      ?
./ttt2 -s 'lpo -prec"0 = +"' add.trs      ✓
```

[0]tick, cross and question mark are clickable hyperlinks

Addition on Natural Numbers

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

Termination Proof without restrictions by $\mathsf{T_TT_2}$

precedence: $+ > \mathsf{s} \sim 0$

Also correct?

$0 > +$, $\mathsf{s} > +$ or $0 = +$

```
./ttt2 -s 'lpo -prec"0 > +"' add.trs        ✓
./ttt2 -s 'lpo -prec"s > +"' add.trs        ?
./ttt2 -s 'lpo -prec"0 = +"' add.trs        ✓
./ttt2 -s 'lpo -prec"+ > 0 > +"' add.trs    ✗
```

[0]tick, cross and question mark are clickable hyperlinks

Addition on Natural Numbers

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

Termination Proof without restrictions by $\mathsf{T_TT_2}$

precedence: $+ > \mathsf{s} \sim 0$

Also correct?

$0 > +$, $\mathsf{s} > +$ or $0 = +$

```
./ttt2 -s 'lpo -prec"0 > +"' add.trs          ✓
./ttt2 -s 'lpo -prec"s > +"' add.trs          ?
./ttt2 -s 'lpo -prec"0 = +"' add.trs          ✓
./ttt2 -s 'lpo -prec"+ > 0 > +"' add.trs      ✗
./ttt2 -s 'lpo -prec"0 = + > s"' add.trs      ✓
```

[0] tick, cross and question mark are clickable hyperlinks

Knuth-Bendix Order

- parameters are precedence and weights

Knuth-Bendix Order

- parameters are precedence and weights
- `-weights` accepts a weights template

Knuth-Bendix Order

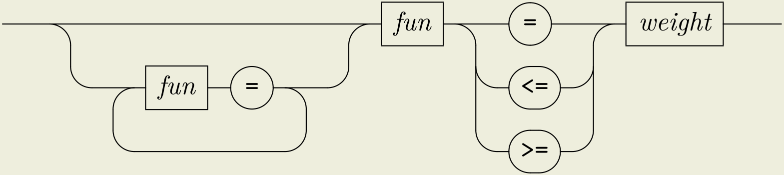
- parameters are precedence and weights
- `-weights` accepts a weights template
- `-w0` takes a value for w_0

Knuth-Bendix Order

- parameters are precedence and weights
- `-weights` accepts a weights template
- `-w0` takes a value for w_0

Syntax

weights



Addition on Natural Numbers

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

Examples

Addition on Natural Numbers

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

Examples

$$8 \leq \mathsf{w}(+) \leq 16$$

Addition on Natural Numbers

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

Examples

$$8 \le \mathsf{w}(+) \le 16$$

```
./ttt2 -s 'kbo -weights "+ <= 16, + >= 8"' add.trs
```
✓

Addition on Natural Numbers

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

Examples

$$8 \leq \mathsf{w}(+) \leq 16$$

```
./ttt2 -s 'kbo -weights "+ <= 16, + >= 8"' add.trs
```
✓

$$16 \leq \mathsf{w}(+) \text{ and } \mathsf{w}(+) \leq 8$$

Addition on Natural Numbers

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

Examples

$$8 \leq \mathsf{w}(+) \leq 16$$

```
./ttt2 -s 'kbo -weights "+ <= 16, + >= 8"' add.trs
```
✓

$$16 \leq \mathsf{w}(+) \text{ and } \mathsf{w}(+) \leq 8$$

```
./ttt2 -s 'kbo -weights "+ <= 8, + >= 16"' add.trs
```
✗

Addition on Natural Numbers

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

Examples

$$8 \leq \mathsf{w}(+) \leq 16$$

```
./ttt2 -s 'kbo -weights "+ <= 16, + >= 8"' add.trs
```
✓

$$16 \leq \mathsf{w}(+) \text{ and } \mathsf{w}(+) \leq 8$$

```
./ttt2 -s 'kbo -weights "+ <= 8, + >= 16"' add.trs
```
✗

$$0 = + > \mathsf{s}, \mathsf{w}(+) = 7, \mathsf{w}(\mathsf{s}) = 3 \text{ and } w_0 = 3$$

Addition on Natural Numbers

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

Examples

$$8 \leq \mathsf{w}(+) \leq 16$$

```
./ttt2 -s 'kbo -weights "+ <= 16, + >= 8"' add.trs
```
✓

$$16 \leq \mathsf{w}(+) \text{ and } \mathsf{w}(+) \leq 8$$

```
./ttt2 -s 'kbo -weights "+ <= 8, + >= 16"' add.trs
```
✗

$$0 = + > \mathsf{s}, \mathsf{w}(+) = 7, \mathsf{w}(\mathsf{s}) = 3 \text{ and } w_0 = 3$$

```
./ttt2 -s 'kbo -prec "0 = + > s" -weights "+ = s = 3"
-w0 3' add.trs
```
✓

Linear Interpretations

- supported by PIs and MIs

Linear Interpretations

- supported by PIs and MIs
- accepted by the `-inters` flag

Linear Interpretations

- supported by PIs and MIs
- accepted by the `-inters` flag
- sums of linear monomials

Linear Interpretations

- supported by PIs and MIs
- accepted by the `-inters` flag
- sums of linear monomials
- underscore denotes arbitrary parts

Linear Interpretations

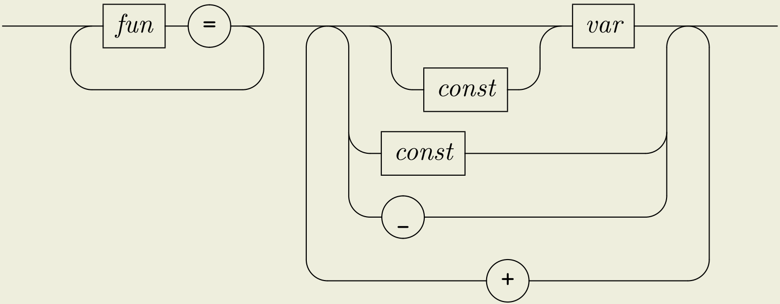
- supported by PIs and MIs
- accepted by the `-inters` flag
- sums of linear monomials
- underscore denotes arbitrary parts
- integer at the end of var denotes the position of the argument

Linear Interpretations

- supported by PIs and MIs
- accepted by the `-inters` flag
- sums of linear monomials
- underscore denotes arbitrary parts
- integer at the end of var denotes the position of the argument

Syntax

inters



Polynomial Interpretations

take natural numbers for “const” in the inters template

Polynomial Interpretations

take natural numbers for "const" in the inters template

Addition on Natural Numbers

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

Examples

$$[+](x, y) = 2x + y + 1, [\mathsf{s}](x) = x + 1, [0] = 0$$

Polynomial Interpretations

take natural numbers for "const" in the inters template

Addition on Natural Numbers

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

Examples

$$[+](x, y) = 2x + y + 1, [\mathsf{s}](x) = x + 1, [0] = 0$$

```
./ttt2 -s 'poly -inters"+ = 2x0 + x1 + 1, s = x0 + 1,
0 = 0"' add.trs
```
✓

Examples cont'd

$$[+](x, y) = 2x + ay + 1, [\mathsf{s}](x) = bx + c, [\mathsf{0}] = 0,$$

where a, b and c are arbitrary

Examples cont'd

$$[+](x, y) = 2x + ay + 1, [\mathsf{s}](x) = bx + c, [\mathsf{0}] = 0,$$

where a, b and c are arbitrary

```
./ttt2 -s 'poly -inters "+ = 2x0 + _ + 1, s = _, 0 = 0"' add.trs
```
✓

Examples cont'd

$$[+](x,y) = 2x + ay + 1, [\mathsf{s}](x) = bx + c, [\mathsf{0}] = 0,$$

where a, b and c are arbitrary

```
./ttt2 -s 'poly -inters "+ = 2x0 + _ + 1, s = _, 0 = 0"' ✓
add.trs
```

Examples cont'd

$$[+](x, y) = 2x + ay + 1, [\mathsf{s}](x) = bx + c, [0] = 0,$$

where a, b and c are arbitrary

```
./ttt2 -s 'poly -inters "+ = 2x0 + _ + 1, s = _, 0 = 0"' ✓
add.trs
```

$$[+](x, y) = 2x + y$$

Examples cont'd

$$[+](x, y) = 2x + ay + 1, [\mathsf{s}](x) = bx + c, [\mathsf{0}] = 0,$$

where a, b and c are arbitrary

```
./ttt2 -s 'poly -inters "+ = 2x0 + _ + 1, s = _, 0 = 0"' add.trs
```
✓

$$[+](x, y) = 2x + y$$

```
./ttt2 -s 'poly -inters "+ = 2x0 + x1"' add.trs
```
✓

Examples cont'd

$$[+](x, y) = 2x + ay + 1, [\mathsf{s}](x) = bx + c, [0] = 0,$$

where a, b and c are arbitrary

```
./ttt2 -s 'poly -inters "+ = 2x0 + _ + 1, s = _, 0 = 0"' add.trs
```
✓

$$[+](x, y) = 2x + y$$

```
./ttt2 -s 'poly -inters "+ = 2x0 + x1"' add.trs
```
✓

$[+](x, y) = 2x + ay$, where a is arbitrary

Examples cont'd

$$[+](x, y) = 2x + ay + 1, [\mathsf{s}](x) = bx + c, [0] = 0,$$

where a, b and c are arbitrary

```
./ttt2 -s 'poly -inters "+ = 2x0 + _ + 1, s = _, 0 = 0"' add.trs
```
✓

$$[+](x, y) = 2x + y$$

```
./ttt2 -s 'poly -inters "+ = 2x0 + x1"' add.trs
```
✓

$[+](x, y) = 2x + ay,$ where a is arbitrary

```
./ttt2 -s 'poly -inters "+ = 2x0 + _ + 0"' add.trs
```
✓

Matrix Interpretations

- matrices of natural numbers as “const”

Matrix Interpretations

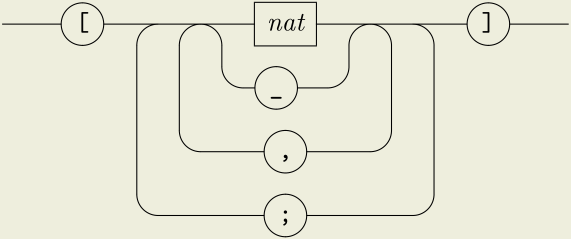
- matrices of natural numbers as "const"
- 0 and 1 for zero-vector and one-vector

Matrix Interpretations

- matrices of natural numbers as “const”
- 0 and 1 for zero-vector and one-vector

Syntax

matrix



Example: Syntax

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$$

in template syntax for MIs: `[1,2,3;4,5,6]`

Example: Syntax

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$$

in template syntax for MIs: `[1,2,3;4,5,6]`

Examples

$$[0] = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \qquad [\mathsf{s}](x) = x + \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

$$[+](x, y) = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} x + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} y + \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

Example: Syntax

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$$

in template syntax for MIs: `[1,2,3;4,5,6]`

Examples

$$[0] = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \qquad [\mathsf{s}](x) = x + \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

$$[+](x,y) = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} x + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} y + \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

```
./ttt2 -s 'matrix -inters "+ = [1,1;0,1]x0 + [1,0;0,1]x1
+ [1;0], s = x0 + 1, 0 = 0"' add.trs
```
✓

Examples cont'd

$$[0] = \begin{pmatrix} a \\ 0 \end{pmatrix} \qquad [\mathsf{s}](x) = x + m_1$$

$$[+](x, y) = \begin{pmatrix} 1 & b \\ c & d \end{pmatrix} x + m_2 y + \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

where a, b, c, d, m_1 and m_2 are arbitrary

Examples cont'd

$$[0] = \begin{pmatrix} a \\ 0 \end{pmatrix} \qquad [\mathsf{s}](x) = x + m_1$$

$$[+](x, y) = \begin{pmatrix} 1 & b \\ c & d \end{pmatrix} x + m_2 y + \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

where a, b, c, d, m_1 and m_2 are arbitrary

```
./ttt2 -s 'matrix -inters "+ = [1,_;_,_]x0 + _ + [1;0],
s = x0 + _, 0 = [_;0]' add.trs
```

✓

Arbitrary Boolean Structure

- arbitrary boolean combinations of atomic constraints

Arbitrary Boolean Structure

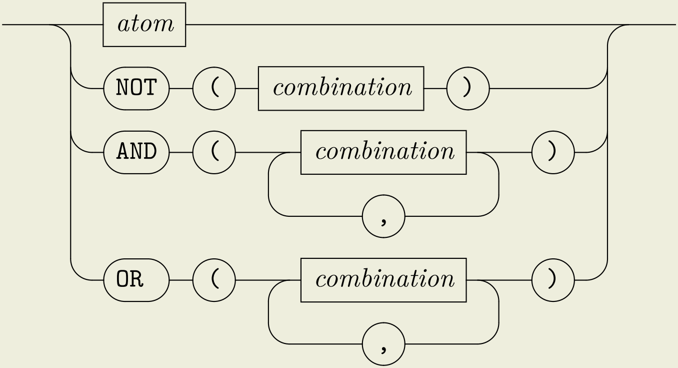
- arbitrary boolean combinations of atomic constraints
- special case: comma-separated list of atoms

Arbitrary Boolean Structure

- arbitrary boolean combinations of atomic constraints
- special case: comma-separated list of atoms

Syntax

combination



Examples

f should not be bigger than h and g at the same time:

Examples

f should not be bigger than h and g at the same time:

```
./ttt2 -s 'lpo -prec "NOT(AND(f > g, f > h))"' 2.60.xml ✓
```

Examples

f should not be bigger than h and g at the same time:

`./ttt2 -s 'lpo -prec "NOT(AND(f > g, f > h))"' 2.60.xml` ✓

constant part of $+$ should be two and when 0 is zero then s should be the successor function:

Examples

f should not be bigger than h and g at the same time:

```
./ttt2 -s 'lpo -prec "NOT(AND(f > g, f > h))"' 2.60.xml
```
✓

constant part of $+$ should be two and when 0 is zero then s should be the successor function:

```
./ttt2 -s 'poly -inters "AND(+ = _ + 2, OR(NOT(0 = 0),
s = x0 + 1))"' 4.11.xml
```
✓

Examples

f should not be bigger than h and g at the same time:
`./ttt2 -s 'lpo -prec "NOT(AND(f > g, f > h))"' 2.60.xml` ✓

constant part of $+$ should be two and when 0 is zero then s should be the successor function:
`./ttt2 -s 'poly -inters "AND(+ = _ + 2, OR(NOT(0 = 0), s = x0 + 1))"' 4.11.xml` ✓

upper triangular matrices with only ones in the diagonals for unary f and binary g and h:

Examples

f should not be bigger than h and g at the same time:

```
./ttt2 -s 'lpo -prec "NOT(AND(f > g, f > h))"' 2.60.xml
```
✓

constant part of + should be two and when 0 is zero then s should be the successor function:

```
./ttt2 -s 'poly -inters "AND(+ = _ + 2, OR(NOT(0 = 0),
s = x0 + 1))"' 4.11.xml
```
✓

upper triangular matrices with only ones in the diagonals for unary f and binary g and h:

```
./ttt2 -s 'poly -inters "f=g=h=[1,_,_;0,1,_;0,0,1]x0+_,
g=h=[1,_,_;0,1,_;0,0,1]x1+_"' 27.xml
```
✓

Extension of the Web interface

Web Interface

Tyrolean Termination Tool 2 (1.17)

1. Input Term Rewrite System

For input use the standard TRS format.

select example ... ▾ or upload file Choose File No file chosen

```
(VAR x y)
(RULES
 add(0,y) -> y
 add(s(x),y) -> s(add(x,y))
 mul(0,y) -> 0
 mul(s(x),y) -> add(y,mul(x,y))
)
```

2. Select Strategy

◉ FAST ○ FBI ○ HYDRA ○ LPO ○ KBO ○ POLY ○ MATRIX(2) ○ MATRIX(3) ○ COMP ○ COMPLEXITY

○ EXPERT

3. Encode State into URL (optional)

encode URL | clear URL

4. Start TTT2

check ☐ use HTML output if available (*experimental feature*)

Web Interface - KBO

Tyrolean Termination Tool 2 (1.17)

1. Input Term Rewrite System

For input use the standard TRS format.

select example ... ▾ or upload file Choose File No file chosen

```
(VAR x y)
(RULES
 add(0,y) -> y
 add(s(x),y) -> s(add(x,y))
 mul(0,y) -> 0
 mul(s(x),y) -> add(y,mul(x,y))
)
```

2. Select Strategy

○ FAST ○ FBI ○ HYDRA ○ LPO ◉ KBO ○ POLY ○ MATRIX(2) ○ MATRIX(3) ○ COMP ○ COMPLEXITY

○ EXPERT

KBO:

PRECEDENCE

WEIGHTS

W0

3. Encode State into URL (optional)

encode URL clear URL

4. Start TTT2

check ☐ use HTML output if available (*experimental feature*)

URL encoding in the Web interface

- idea: encode content of web interface into URL

URL encoding in the Web interface

- idea: encode content of web interface into URL
- we encode the TRS, the strategy and the templates

URL encoding in the Web interface

- idea: encode content of web interface into URL
- we encode the TRS, the strategy and the templates
- encoded into the query string of the URL

URL encoding in the Web interface

- idea: encode content of web interface into URL
- we encode the TRS, the strategy and the templates
- encoded into the query string of the URL

Normal URL

`http://colo6-c703.uibk.ac.at/ttt2/web/`

URL encoding in the Web interface

- idea: encode content of web interface into URL
- we encode the TRS, the strategy and the templates
- encoded into the query string of the URL

Normal URL

`http://colo6-c703.uibk.ac.at/ttt2/web/`

URL with encoded Query String

http://colo6-c703.uibk.ac.at/ttt2/web/?problem=(VAR%20x%20y)%0A(RULES%0A%20add(0%2Cy)%20-%3E%20y%0A%20add(s(x)%2Cy)%20-%3E%20s(add(x%2Cy))%0A)&strategy=kbo&template=add%20%3E%20s&template1=add%20%3D%20s%20%3D%208&template2=3

Conclusion

- template mechanism to check specific proofs

Conclusion

- template mechanism to check specific proofs
- available for LPO, KBO, PIs and MIs

Conclusion

- template mechanism to check specific proofs
- available for LPO, KBO, PIs and MIs
- web interface extension to save the configurations for an example

Conclusion

- template mechanism to check specific proofs
- available for LPO, KBO, PIs and MIs
- web interface extension to save the configurations for an example
- to have near absolute certainty you should use CeTA to validate the output of $\mathsf{T_TT_2}$

Conclusion

- template mechanism to check specific proofs
- available for LPO, KBO, PIs and MIs
- web interface extension to save the configurations for an example
- to have near absolute certainty you should use CeTA to validate the output of $\mathsf{T_TT_2}$

Future Work

- templates for other methods like arctic interpretations, the (generalized) subterm criterion, ...

Conclusion

- template mechanism to check specific proofs
- available for LPO, KBO, PIs and MIs
- web interface extension to save the configurations for an example
- to have near absolute certainty you should use CeTA to validate the output of $\mathsf{T_TT_2}$

Future Work

- templates for other methods like arctic interpretations, the (generalized) subterm criterion, ...
- which extensions to our templates are most useful for teaching?

Thank you for your attention!