SAT Solving for Termination Analysis

Nao Hirokawa
JAIST

Term Rewriting

DEFINITION

- pair of terms $l \to r$ is rewrite rule if $l \notin \mathcal{V} \wedge \mathcal{V}\mathrm{ar}(r) \subseteq \mathcal{V}\mathrm{ar}(l)$
- term rewrite system (TRS) is set of rewrite rules
- (rewrite relation) $s \to_{\mathcal{R}} t$ if $\exists l \to r \in \mathcal{R}$, context C, substitution σ. $s = C[l\sigma] \ \wedge \ t = C[r\sigma]$

EXAMPLE

TRS $\mathcal{R}$

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

rewriting

$$\begin{aligned} \mathsf{s}(0) \times \mathsf{s}(0) &\to_{\mathcal{R}} \mathsf{s}(0) \times 0 + \mathsf{s}(0) \\ &\to_{\mathcal{R}} 0 + \mathsf{s}(0) \\ &\to_{\mathcal{R}} \mathsf{s}(0 + 0) \\ &\to_{\mathcal{R}} \mathsf{s}(0) \qquad \text{normal form} \end{aligned}$$

Eratosthenes' sieve

$$
\begin{array}{rl}
1: & x - 0 \to x \\
2: & \mathsf{s}(x) - \mathsf{s}(y) \to x - y \\
3: & 0 \leqslant y \to \mathsf{true} \\
4: & \mathsf{s}(x) \leqslant 0 \to \mathsf{false} \\
5: & \mathsf{s}(x) \leqslant \mathsf{s}(y) \to x \leqslant y \\
6: & \mathsf{if}(\mathsf{true}, x, y) \to x \\
7: & \mathsf{if}(\mathsf{false}, x, y) \to y \\
8: & x \mid 0 \to \mathsf{true} \\
9: & \mathsf{s}(x) \mid \mathsf{s}(y) \to \mathsf{if}(x \leqslant y, \mathsf{s}(x) \mid (y - x), \mathsf{false})
\end{array}
$$

$$
\begin{array}{rl}
10: & \mathsf{filter}(x, \mathsf{nil}) \to \mathsf{nil} \\
11: & \mathsf{filter}(x, y : ys) \to \mathsf{if\_filter}(x \mid y, x, y : ys) \\
12: & \mathsf{if\_filter}(\mathsf{true}, x, y : ys) \to \mathsf{filter}(x, ys) \\
13: & \mathsf{if\_filter}(\mathsf{false}, x, y : ys) \to y : \mathsf{filter}(x, ys) \\
14: & \mathsf{sieve}(\mathsf{nil}) \to \mathsf{nil} \\
15: & \mathsf{sieve}(x : xs) \to x : \mathsf{sieve}(\mathsf{filter}(x, xs))
\end{array}
$$

$$
\begin{aligned}
\mathsf{sieve}(2 : 3 : 4 : 5 : 6 : \mathsf{nil}) &\to_{\mathcal{R}} 2 : \mathsf{sieve}(\mathsf{filter}(2, 3 : 4 : 5 : 6 : \mathsf{nil})) \\
&\to^{+}_{\mathcal{R}} 2 : \mathsf{sieve}(3 : 5 : \mathsf{nil})) \\
&\to_{\mathcal{R}} 2 : 3 : \mathsf{sieve}(\mathsf{filter}(3, 5 : \mathsf{nil})) \\
&\to^{+}_{\mathcal{R}} 2 : 3 : \mathsf{sieve}(5 : \mathsf{nil}) \\
&\to_{\mathcal{R}} 2 : 3 : 5 : \mathsf{sieve}(\mathsf{filter}(5, \mathsf{nil})) \\
&\to^{+}_{\mathcal{R}} 2 : 3 : 5 : \mathsf{sieve}(\mathsf{nil}) \\
&\to_{\mathcal{R}} 2 : 3 : 5 : \mathsf{nil}
\end{aligned}
$$

is this TRS terminating?

Haskell-like program

$$\begin{aligned} \mathsf{id}\ x &\to x \\ \mathsf{add}\ 0\ y &\to y \\ \mathsf{add}\ (\mathsf{s}\ x)\ y &\to \mathsf{s}\ (\mathsf{add}\ x\ y) \end{aligned} \qquad \begin{aligned} \mathsf{map}\ f\ \mathsf{nil} &\to \mathsf{nil} \\ \mathsf{map}\ f\ (x : y) &\to (f\ x) : (\mathsf{map}\ f\ y) \end{aligned}$$

uncurried TRS

$$\begin{aligned} \mathsf{id}_1(x) &\to x \\ \mathsf{add}_2(0, y) &\to \mathsf{id}_1(y) \\ \mathsf{add}_2(\mathsf{s}_1(x), y) &\to \mathsf{s}_1(\mathsf{add}_2(x, y)) \\ \mathsf{map}_2(f, \mathsf{nil}) &\to \mathsf{nil} \\ \mathsf{map}_2(f, :_2(x, y)) &\to :_2(f \circ x, \mathsf{map}_2(f, y)) \end{aligned} \qquad \begin{aligned} : \circ x &\to :_1(x) \\ :_1(x) \circ y &\to :_2(x, y) \end{aligned} \qquad \begin{aligned} \mathsf{id} \circ x &\to \mathsf{id}_1(x) \\ \mathsf{add} \circ x &\to \mathsf{add}_1(x) \\ \mathsf{add}_1(x) \circ y &\to \mathsf{add}_2(x, y) \\ \mathsf{s} \circ x &\to \mathsf{s}_1(x) \\ \mathsf{map} \circ x &\to \mathsf{map}_1(x) \\ \mathsf{map}_1(x) \circ y &\to \mathsf{map}_2(x, y) \end{aligned}$$

APPLICATIONS

- verification for functional programming
- theorem proving
- code optimization in compilers
- symbolic computation in mathematics
- ...

Is Non-termination More Interesting?

DEFINITION

$\mathcal{R}$ is non-terminating if $\exists$infinite rewrite sequence $t_1 \to_{\mathcal{R}} t_2 \to_{\mathcal{R}} \cdots$

- non-termination tools are useful for detecting bugs
- liveness property can be reduced to termination property

Termination Proof Methods

long history and many results

polynomial interpretation order, multiset path order, Knuth-Bendix order, recursive decomposition order, semantic path order, general path order, $\mathcal{C}_\mathcal{E}$-termination, hierarchical modular theorem, freezing, erasure method, size-change termination, lexicographic path order, match-bound, matrix interpretation order, dependency pair method, monotonic semantic path order, forward closure, semantic labeling, self labeling, predictive labeling uncurrying transformation, innermost recursive path order, ...

Termination Proof Methods

long history and many results

polynomial interpretation order, multiset path order, Knuth-Bendix order, recursive decomposition order, semantic path order, general path order, $\mathcal{C}_\mathcal{E}$-termination, hierarchical modular theorem, freezing, erasure method, size-change termination, lexicographic path order, match-bound, matrix interpretation order, dependency pair method, monotonic semantic path order, forward closure, semantic labeling, self labeling, predictive labeling uncurrying transformation, innermost recursive path order, ...

- polynomial order and lexicographic path order (LPO)
- dependency pair method

This Talk

finding suitable

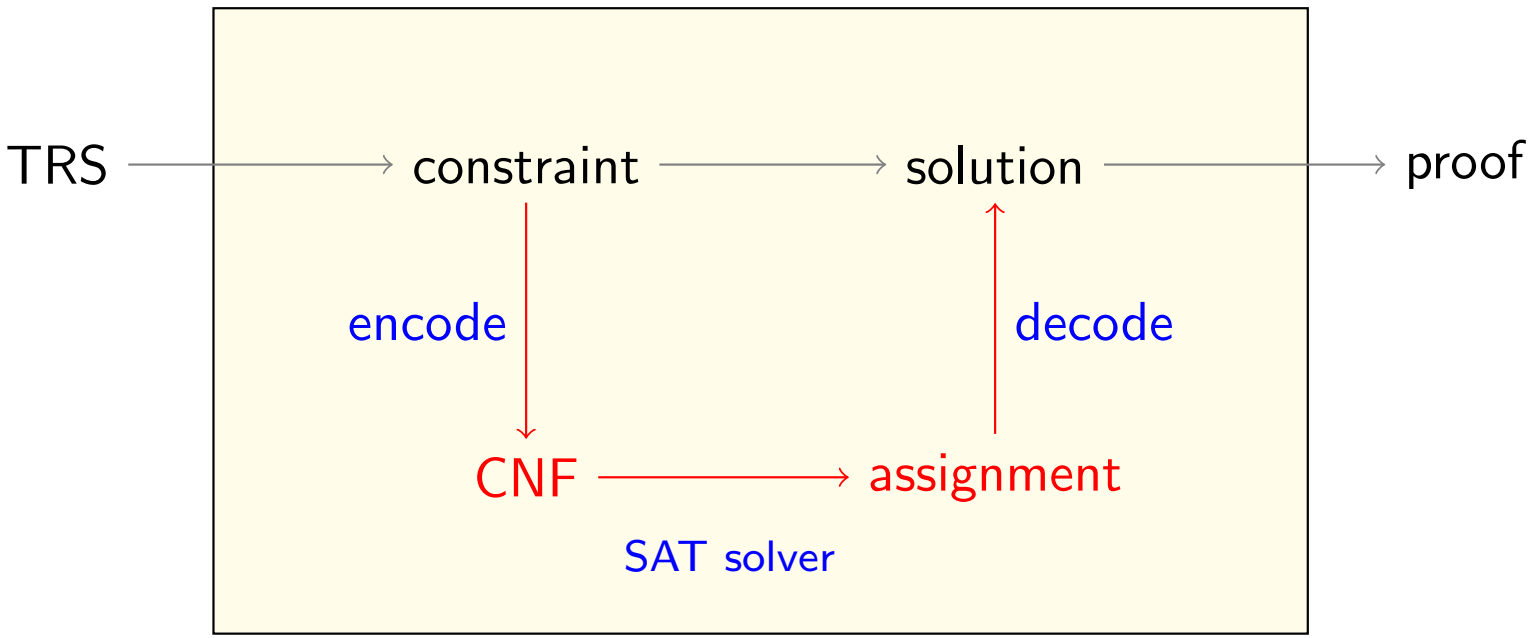
- LPO precedence
- polynomial interpretation
- argument filtering

is main bottleneck

THIS TALK

use SAT solver or Diophantine constraint solver to find them

Implementation of Termination Tools



Overview

- SAT
- Diophantine constraints
- precedence termination
- lexicographic path orders
- polynomial interpretation orders
- dependency pair method

SAT

SAT solvers

DEFINITION CNF

(CNF) $C ::= c_1 \wedge \cdots \wedge c_m$

(clause) $c ::= l_1 \vee \cdots \vee l_n$

(literal) $l ::= x \mid \neg x$

- SAT solvers finds assignment α such that $\alpha \models C$
- popular solvers: zChaff, Minisat, ...

EXAMPLE

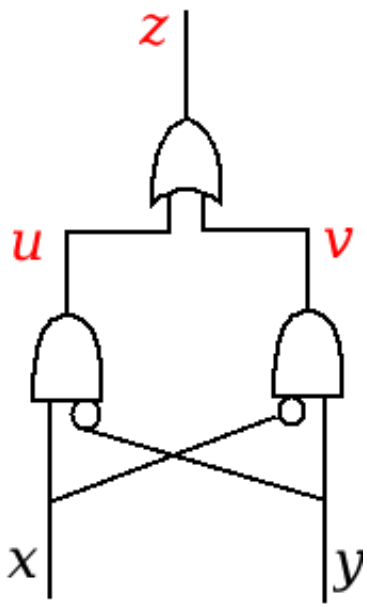
$$\models \bigwedge \left\{ \begin{array}{l} x_1 \vee \neg x_2, \\ \neg x_1 \vee x_2 \vee \neg x_3, \\ x_3 \vee \neg x_2, \\ \neg x_3 \end{array} \right\} ?$$

```
$ cat a.cnf
p cnf 3 #variables 4 #clauses
1 -2 0
-1 2 -3 0
3 -2 0
-3 0

$ minisat a.cnf a.ans
$ cat a.ans
SAT
1 -2 -3 0
```

Tseitin's transformation

$$x \leftrightarrow (y \wedge z) = (\neg x \vee y) \wedge (\neg x \vee z) \wedge (\neg y \vee \neg z \vee x)$$
$$x \leftrightarrow (y \vee z) = x \leftrightarrow \neg(\neg y \wedge \neg z)$$



$$\models (x \wedge \neg y) \vee (\neg x \wedge y)$$
$$\Leftrightarrow \quad \models z \text{ where } \begin{cases} z = u \vee v \\ u = x \vee \neg y \\ v = \neg x \vee y \end{cases}$$
$$\Leftrightarrow \quad \models z \wedge \bigwedge \left\{ \begin{array}{l} z \leftrightarrow (u \vee v), \\ u \leftrightarrow (x \vee \neg y), \\ v \leftrightarrow (\neg x \vee y) \end{array} \right\}$$

Implementation in OCaml

```
type literal = int and cluase = int list and cnf = clause list

let nvars = ref 0
let fresh () = incr nvars; !nvars
let cnf = ref [[]]

let conj xs =
  let y = fresh () in
  cnf := (y :: [ -x | x <- xs ]) :: [ [-y; x] | x <- xs ] @ !cnf; y

let disj xs = - conj [ -x | x <- xs ]
let negate x = -x
```

EXAMPLE

CNF of $(x \wedge \neg y) \vee (\neg x \wedge y)$ is

```
let z = disj [conj [1; -2]; conj [-1; 2]] in [z] :: !cnf
```

Why Very Very Fast?

- decision:
  depth-first search with heuristics

- deduction:
  boolean constraint propagation
  two watched literals

- diagnosis (learning):
  conflict-directed backtrack (1-UIP)

Boolean Constraint Propagation

current assignment:

$$x = 1, y = 0$$

current decision: $z = 1$

$\neg x \vee z \vee b \vee c$	✓
$\neg x \vee y \vee \neg z \vee \neg w$	$w = 0$
$w \vee y \vee a$	–
$\neg a \vee z \vee b \vee y$	–
$\neg x \vee c \vee d \vee e$	–

Two Watched Literals

current assignment:

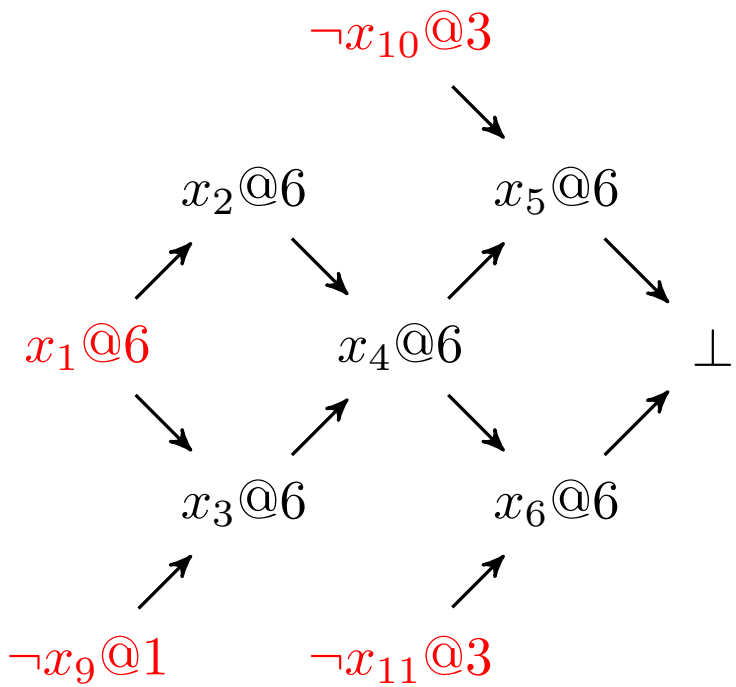
$$x = 1, y = 0$$

current decision: $z = 1$

$$\neg x \vee z \vee \boxed{b} \vee \boxed{c}$$
$$\neg x \vee y \vee \boxed{\neg z} \vee \boxed{\neg w}$$
$$\boxed{w} \vee \boxed{y} \vee a$$
$$\boxed{\neg a} \vee z \vee \boxed{b} \vee y$$
$$\neg x \vee \boxed{c} \vee \boxed{d} \vee e$$

Conflict-Directed Backtracking

current decision: $x_1 = 1@6$



- if $x_1 \neq 0$, back to level 3
- GRASP learns $\neg x_1 \vee x_9 \vee x_{10} \vee x_{11}$ (UIP is $\neg x_4$)

Diophantine Constraints

Diophantine Constraints

SYNTAX

$$
\begin{aligned}
c &::= \neg c \mid c_1 \vee c_2 \mid c_1 \wedge c_2 \\
&\quad \mid p_1 \geq p_2 \mid p_1 \leq p_2 \mid p_1 > p_2 \mid p_1 < p_2 \mid p_1 = p_2 \mid p_1 \neq p_2 \\
p &::= x \mid n \mid p_1 + p_1 \mid p_1 \times p_2
\end{aligned}
$$

where $x \in \mathcal{V}$ and $n \in \mathbb{Z}$

SEMANTICS

- $\alpha \vDash p_1 \geq p_2$ if $[\alpha]_{\mathbb{N}}(p_1) \geq [\alpha]_{\mathbb{N}}(p_2)$
- $\alpha \vDash \neg c$ if $\alpha \vDash c$ does not hold
- $\alpha \vDash c_1 \vee c_2$ if $\alpha \vDash c_1$ or $\alpha \vDash c_2$
- $\alpha \vDash c_1 \wedge c_2$ if $\alpha \vDash c_1$ and $\alpha \vDash c_2$

- $\vDash c$ if $\exists \alpha : \mathcal{V} \to \mathbb{Z}.\ \alpha \vDash c$

Diophantine Constraint Problems

1. $\models \quad (a > b \ \vee \ b > d) \ \wedge \ b > c \ \wedge \ c > a \quad ?$

satisfiable $a = 3, b = 2, c = 1, d = 4$

2. $\models \quad \neg(b > a) \ \wedge \ b > c \ \wedge \ c > a \quad ?$

unsatisfiable

3. $\models a^2 - b^2 = 5 \ \vee \ (2a + b \leq 12 \ \wedge \ ab \geq 15) \quad ?$

satisfiable $a = 3, b = 5$

NOTE

1 and 2 are precedence problem

How To Solve Diophantine Constraints?

- in general undecidable
- so search only from finite domain (incomplete)
- efficient easy way is to use SAT solver to implement Diophantine constraints

Encoding Arithmetics (I)

for $a \leq x \leq b$, introduce

$$x_a, \ldots, x_b$$

add uniqueness condition:

$$(x_a \vee x_{a+1} \vee \cdots \vee x_b) \wedge \bigwedge \{\neg x_i \vee x_j \mid a \leq i, j \leq b,\ i \neq j\}$$

for $z = x + y$ ($a \leq x \leq b$, $c \leq y \leq d$), introduce

$$z_{a+c}, \ldots, z_{b+d}$$

Encoding Arithmetics (II)

for $x + y = z$, introduce propositional variables

$$x = (x_1, \ldots, x_k), y = (y_1, \ldots, y_k), z = (z_1, \ldots, z_k)$$

implement k-bit ripple carry adder...

Precedence Termination

Precedence Termination

$>$ is precedence if $>$ is strict order on function symbols

DEFINITION

$l >_{\mathsf{prec}} r$ if $\mathcal{V}\mathsf{ar}(l) \supseteq \mathcal{V}\mathsf{ar}(r)$, $l = f(\ldots)$ and $\forall g \in \mathcal{F}\mathsf{un}(r).\ f > g$

THEOREM

$\mathcal{R}$ is terminating if there exists well-founded precedence $>$ such that $\mathcal{R} \subseteq >_{\mathsf{prec}}$

NOTATION

$\mathcal{R} \subseteq >$ if and only if $\forall l \to r \in \mathcal{R}.\ l > r$

REMARK

this constraint is Diophantine constraint

Example

prove termination of TRS

$$\begin{aligned}
\mathsf{not}(x) &\to \mathsf{if}(x, \mathsf{false}, \mathsf{true}) & \mathsf{if}(\mathsf{true}, x, y) &\to x\\
\mathsf{and}(x, y) &\to \mathsf{if}(x, y, \mathsf{false}) & \mathsf{if}(\mathsf{false}, x, y) &\to y\\
\mathsf{or}(x, y) &\to \mathsf{if}(x, \mathsf{true}, y)\\
\mathsf{equiv}(x, y) &\to \mathsf{if}(x, y, \mathsf{not}(y))
\end{aligned}$$

$$\bigwedge \left\{ \begin{array}{rlrlrl} \mathsf{not}(x) & >_{\mathsf{prec}} \mathsf{if}(x, \mathsf{false}, \mathsf{true}), & \mathsf{if}(\mathsf{true}, x, y) & >_{\mathsf{prec}} x, \\ \mathsf{and}(x, y) & >_{\mathsf{prec}} \mathsf{if}(x, y, \mathsf{false}), & \mathsf{if}(\mathsf{false}, x, y) & >_{\mathsf{prec}} y, \\ \mathsf{or}(x, y) & >_{\mathsf{prec}} \mathsf{if}(x, \mathsf{true}, y), & & \\ \mathsf{equiv}(x, y) & >_{\mathsf{prec}} \mathsf{if}(x, y, \mathsf{not}(y)) & & \end{array} \right\}$$

$$\Leftrightarrow \left(\begin{array}{rlcrlcrlc} \mathsf{not} & > \mathsf{if} & \wedge & \mathsf{not} & > \mathsf{false} & \wedge & \mathsf{not} & > \mathsf{true} & \wedge \\ \mathsf{and} & > \mathsf{if} & \wedge & \mathsf{and} & > \mathsf{false} & \wedge & & & \\ \mathsf{or} & > \mathsf{if} & \wedge & \mathsf{or} & > \mathsf{true} & \wedge & & & \\ \mathsf{equiv} & > \mathsf{if} & \wedge & \mathsf{equiv} & > \mathsf{not} & & & & \end{array} \right)$$

Diophantine constraint

```
% diosat if.dio
SAT
not = -13967; if = -28612; false = -16738; true = -18786;
and = -1; or = -1; equiv = -370;
```

take precedence

$$\mathsf{and}, \mathsf{or} > \mathsf{equiv} > \mathsf{not} > \mathsf{false} > \mathsf{true} > \mathsf{if}$$

then $\mathcal{R} \subseteq >_{\mathsf{prec}}$, hence TRS is terminating

Example

prove termination of TRS $\mathcal{R}$

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

REMARK

precedence termination does not hold:

$$\begin{aligned} & \mathcal{R} \subseteq >_{\mathsf{prec}} \\ \Leftrightarrow\ & \mathsf{s}(x) + y >_{\mathsf{prec}} \mathsf{s}(x + y) \ \wedge \ \cdots \\ \Leftrightarrow\ & + > \mathsf{s} \ \wedge \ + > + \ \wedge \ \cdots \end{aligned}$$

is unsatisfiable Diophantine constraint

Decidability

- Diophantine constraint problems are undecidable in general
- precedence constraint problems are decidable
- completeness threshold is number of function symbols

Lexicographic Path Orders

Lexicographic Path Order

DEFINITION LPO

for precedence $>$

$s >_{\mathsf{lpo}} t$ if $s = f(s_1, \ldots, s_m)$, and $t \in \mathcal{V}\mathrm{ar}(s)$ or $t = g(t_1, \ldots, t_n)$, and

- $\forall i \in \{1, \ldots, m\}.\ s_i >_{\mathsf{lpo}} t \ \vee \ s_i = t$,
- $f > g \ \wedge \ \forall i \in \{1, \ldots, n\}.\ s >_{\mathsf{lpo}} t_i$, or
- $f = g \ \wedge \ \exists i \in \{1, \ldots, n\}.$
  $s_1 = t_1 \ \wedge \cdots \wedge \ s_{i-1} = t_{i-1} \ \wedge \ t_i >_{\mathsf{lpo}} s_i \ \wedge \ s >_{\mathsf{lpo}} t_{i+1}, \ldots s >_{\mathsf{lpo}} t_n$

THEOREM Kamin and Levy, 1980

finite TRS $\mathcal{R}$ is terminating if $\exists$well-founded precedence $>$ such that $\mathcal{R} \subseteq >_{\mathsf{lpo}}$

REMARK

constraints are Diophantine constraints (precedence constraints)

Example

prove termination of TRS $\mathcal{R}$

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

$$\mathcal{R} \subseteq >_{\mathsf{lpo}}$$
$$\Leftrightarrow\ (+ > \mathsf{s} \ \vee\ \bot) \ \wedge\ (\times > 0 \ \vee\ \top) \ \wedge\ (\times > + \ \vee\ (\mathsf{s} > + \ \wedge\ \mathsf{s} > \times \ \wedge\ \bot))$$

PROOF

take precedence $\times > + > \mathsf{s} > 0$

$\mathcal{R} \subseteq >_{\mathsf{lpo}}$ holds, hence TRS is terminating

Polynomial Interpretation Orders

Polynomial Constraints (with Unknown Coefficients)

SYNTAX

$$c ::= r \mid \neg c \mid c_1 \vee c_2 \mid c_1 \wedge c_2$$
$$r ::= p_1 \geq p_2 \mid p_1 \leq p_2 \mid p_1 > p_2 \mid p_1 < p_2 \mid p_1 = p_2 \mid p_1 \neq p_2$$
$$p ::= x \mid X \mid n \mid p_1 + p_1 \mid p_1 \times p_2$$

where $x \in \mathcal{V}$, $X \in \mathcal{X}$, and $n \in \mathbb{Z}$

SEMANTICS

- $\sigma \vDash p_1 \geq p_2$ if $\forall \alpha : \mathcal{X} \to \mathbb{Z}.\ \ [\alpha]_{\mathbb{Z}}(p_1\sigma) \geq [\alpha]_{\mathbb{Z}}(p_2\sigma)$
- $\sigma \vDash \neg c$ if $\sigma \vDash c$ does not hold
- $\sigma \vDash c_1 \vee c_2$ if $\sigma \vDash c_1$ or $\sigma \vDash c_2$
- $\sigma \vDash c_1 \wedge c_2$ if $\sigma \vDash c_1$ and $\sigma \vDash c_2$

- $\vDash c$ if $\exists \sigma : \mathcal{V} \to \mathbb{Z}.\ \sigma \vDash c$

Polynomial Constraint Problems

1. $\models \quad aX + bY > bX + 2b \ \wedge \ a \geq 0 \ \wedge \ b \geq 0 \quad ?$

unsatisfiable

2. $\models \quad aX + b > aX^2 + c \ \wedge \ a > 0 \quad ?$

unsatisfiable

3. $\models aX + aY + bY + c > aX + Y + 3 \quad ?$

satisfiable $a = 1, b = 1, c = 4$

Polynomial Interpretation Orders

Lankford 1979

DEFINITION interpretation order

$s >_A t$ if $[\alpha]_{\mathcal{A}}(s) > [\alpha]_{\mathcal{A}}(t)$ for all assignments $\alpha : \mathcal{V} \to A$

DEFINITION strict monotonicity

$f(a_1, \ldots, a_i, \ldots, a_n) > f(a_1, \ldots, a'_i, \ldots, a_n)$ if $a_i > a'_i$

THEOREM

TRS $\mathcal{R}$ is terminating if there exists well-founded strictly monotone algebra $(\mathcal{A}, >)$ such that $\mathcal{R} \subseteq >_A$

Example

TRS $\mathcal{R}$

$$\begin{array}{llll} x + 0 & \to & x & \qquad x + \mathsf{s}(y) & \to & \mathsf{s}(x + y) \\ x \times 0 & \to & 0 & \qquad x \times \mathsf{s}(y) & \to & x \times y + x \end{array}$$

take well-founded strictly monotone interpretations

$$\begin{array}{rlrl} 0_{\mathbb{N}} &= 1 & \qquad \times_{\mathbb{N}}(X, Y) &= 2XY + Y + 1 \\ \mathsf{s}_{\mathbb{N}}(X) &= X + 1 & \qquad +_{\mathbb{N}}(X, Y) &= X + 2Y \end{array}$$

then $\mathcal{R} \subseteq >_{\mathbb{N}}$ corresponds to

$$\bigwedge \left\{ \begin{array}{llcll} [\alpha]_{\mathbb{N}}(x + 0) & = \alpha(x) + 2 & > & x & = [\alpha]_{\mathbb{N}}(x) \\ [\alpha]_{\mathbb{N}}(x + \mathsf{s}(y)) & = \alpha(x) + 2\alpha(y) + 2 & > & \alpha(x) + 2\alpha(y) + 1 & = [\alpha]_{\mathbb{N}}(\mathsf{s}(x + y)) \\ [\alpha]_{\mathbb{N}}(x \times 0) & = 2\alpha(x) + 2 & > & 1 & = [\alpha]_{\mathbb{N}}(0) \\ [\alpha]_{\mathbb{N}}(x \times \mathsf{s}(y)) & = \ldots & > & \ldots & = [\alpha]_{\mathbb{N}}(x \times y + x) \end{array} \right\}$$

for all $\alpha : \mathcal{V} \to \mathbb{N}$. this holds, hence $\mathcal{R}$ is terminating

how to check constraint automatically?

Example

TRS $\mathcal{R}$

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

take well-founded strictly monotone interpretations

$$0_{\mathbb{N}} = 1 \qquad \times_{\mathbb{N}}(X, Y) = 2XY + Y + 1$$
$$\mathsf{s}_{\mathbb{N}}(X) = X + 1 \qquad +_{\mathbb{N}}(X, Y) = X + 2Y$$

then $\mathcal{R} \subseteq >_{\mathbb{N}}$ corresponds to

$$\bigwedge \left\{ \begin{array}{llll} [\alpha]_{\mathbb{N}}(x + 0) & = X + 2 & > X & = [\alpha]_{\mathbb{N}}(x) \\ [\alpha]_{\mathbb{N}}(x + \mathsf{s}(y)) & = X + 2Y + 2 & > X + 2Y + 1 & = [\alpha]_{\mathbb{N}}(\mathsf{s}(x + y)) \\ [\alpha]_{\mathbb{N}}(x \times 0) & = 2X + 2 & > 1 & = [\alpha]_{\mathbb{N}}(0) \\ [\alpha]_{\mathbb{N}}(x \times \mathsf{s}(y)) & = \ldots & > \ldots & = [\alpha]_{\mathbb{N}}(x \times y + x) \end{array} \right\}$$

this holds, hence TRS is terminating

how to check constraint automatically?

How to Solve Polynomial Constraints?

in general undecidable but there is easy (incomplete) criterion

FACT

polynomial is non-negative if all coefficients of are non-negative

EXAMPLE

$$
\begin{aligned}
& 2XY + 2X + 2Y + 2 > X + Y \\
\Leftrightarrow\ & (2XY + 2X + 2Y + 2) - (X + Y) - 1 \geq 0 \\
\Leftrightarrow\ & 2XY + 1X + 1Y + 1 \geq 0 \\
\Leftarrow\ & \bigwedge \left\{ \begin{array}{ll} 2 & \geq 0 \\ 1 & \geq 0 \\ 1 & \geq 0 \\ 1 & \geq 0 \end{array} \right\}
\end{aligned}
$$

Example

TRS $\mathcal{R}$

$$\begin{array}{lll} x + 0 & \to & x \\ x \times 0 & \to & 0 \end{array} \qquad \begin{array}{lll} x + \mathsf{s}(y) & \to & \mathsf{s}(x + y) \\ x \times \mathsf{s}(y) & \to & x \times y + x \end{array}$$

take well-founded strictly monotone interpretations

$$\begin{array}{rl} 0_{\mathbb{N}} = 1 & \qquad \times_{\mathbb{N}}(X, Y) = 2XY + Y + 1 \\ \mathsf{s}_{\mathbb{N}}(X) = X + 1 & \qquad +_{\mathbb{N}}(X, Y) = X + 2Y \end{array}$$

then $\mathcal{R} \subseteq >_{\mathbb{N}}$ corresponds to

$$\bigwedge \left\{ \begin{array}{llll} [\alpha]_{\mathbb{N}}(x + 0) & = X + 2 & > X & = [\alpha]_{\mathbb{N}}(x) \\ [\alpha]_{\mathbb{N}}(x + \mathsf{s}(y)) & = X + 2Y + 2 & > X + 2Y + 1 & = [\alpha]_{\mathbb{N}}(\mathsf{s}(x + y)) \\ [\alpha]_{\mathbb{N}}(x \times 0) & = 2X + 2 & > 1 & = [\alpha]_{\mathbb{N}}(0) \\ [\alpha]_{\mathbb{N}}(x \times \mathsf{s}(y)) & = \ldots & > \ldots & = [\alpha]_{\mathbb{N}}(x \times y + x) \end{array} \right\}$$

this holds, hence TRS is terminating

how to find suitable interpretations?

Example

TRS $\mathcal{R}$

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

take well-founded strictly monotone interpretations

$$0_{\mathbb{N}} = a \qquad +_{\mathbb{N}}(X, Y) = dXY + eX + fY + g$$
$$\mathsf{s}_{\mathbb{N}}(X) = bX + c \qquad \times_{\mathbb{N}}(X, Y) = iXY + jX + kY + l$$

then $\mathcal{R} \subseteq >_{\mathbb{N}}$ corresponds to

$$\vDash \bigwedge \left\{ \begin{array}{lll} [\alpha]_{\mathbb{N}}(x + 0) & = bdX + eX + af + g & \\ & > X & = [\alpha]_{\mathbb{N}}(x) \\ [\alpha]_{\mathbb{N}}(x + \mathsf{s}(y)) & = bXY + eX + bfY + cf + g & \\ & > bdXY + beX + bfY + bg + c & = [\alpha]_{\mathbb{N}}(\mathsf{s}(x + y)) \\ & \ldots & \end{array} \right\}$$

how to solve polynomial constraint problem?

Translating to Diophantine Constraints

in general undecidable but there is easy (incomplete) criterion

FACT

polynomial is non-negative if all coefficients are non-negative

EXAMPLE

$$
\begin{aligned}
& \models aXY + bX + cY + d > eX + fY + g \\
\Leftrightarrow \quad & \models aXY + (b-e)X + (c-f)Y + (d-g-1) \geq 0 \\
\Leftarrow \quad & \models \bigwedge \left\{ \begin{array}{rl} a & \geq 0 \\ b-e & \geq 0 \\ c-f & \geq 0 \\ d-g-1 & \geq 0 \end{array} \right\}
\end{aligned}
$$

Example

TRS $\mathcal{R}$

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

take well-founded strictly monotone interpretations

$$0_{\mathbb{N}} = a \qquad +_{\mathbb{N}}(X, Y) = dXY + eX + fY + g$$
$$\mathsf{s}_{\mathbb{N}}(X) = bX + c \qquad \times_{\mathbb{N}}(X, Y) = iXY + jX + kY + l$$

then $\mathcal{R} \subseteq >_{\mathbb{N}}$ corresponds to

$$\models \bigwedge \left\{ \begin{array}{lll} [\alpha]_{\mathbb{N}}(x + 0) & = bdX + eX + af + g & \\ & > X & = [\alpha]_{\mathbb{N}}(x) \\ [\alpha]_{\mathbb{N}}(x + \mathsf{s}(y)) & = bXY + eX + bfY + cf + g & \\ & > bdXY + beX + bfY + bg + c & = [\alpha]_{\mathbb{N}}(\mathsf{s}(x + y)) \\ & \ldots & \end{array} \right\}$$

$\wedge$?

FACT

- polynomial interpretation is weakly monotone if all coefficients are non-negative
- polynomial interpretation is strictly monotone if all coefficients are positive
- polynomial interpretation is non-negative if it is weakly monotone and constant is non-negative

Example

TRS $\mathcal{R}$

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

take well-founded strictly monotone interpretations

$$0_{\mathbb{N}} = a \qquad +_{\mathbb{N}}(X, Y) = dXY + eX + fY + g$$
$$\mathsf{s}_{\mathbb{N}}(X) = bX + c \qquad \times_{\mathbb{N}}(X, Y) = iXY + jX + kY + l$$

then $\mathcal{R} \subseteq >_{\mathbb{N}}$ corresponds to

$$\vDash \bigwedge \left\{ \begin{array}{lll} [\alpha]_{\mathbb{N}}(x + 0) & = bdX + eX + af + g & \\ & > X & = [\alpha]_{\mathbb{N}}(x) \\ [\alpha]_{\mathbb{N}}(x + \mathsf{s}(y)) & = bXY + eX + bfY + cf + g & \\ & > bdXY + beX + bfY + bg + c & = [\alpha]_{\mathbb{N}}(\mathsf{s}(x + y)) \\ & \ldots & \end{array} \right\}$$

$$\wedge \boxed{\bigwedge \{b, d, e, f, i, j, k > 0, \quad a, c, g, l \geq 0\}}$$

Dependency Pair Method

Dependency Pair Method

Arts, Giesl TCS 2000

NOTATION

for $t = f(t_1, \ldots, t_n)$, $t^\sharp$ denotes $f^\sharp(t_1, \ldots, t_n)$ where $f^\sharp$ is fresh symbol

DEFINITION

$l^\sharp \to t^\sharp$ is dependency pair of TRS $\mathcal{R}$ if $\exists l \to r \in \mathcal{R}$:
$r \trianglerighteq t$, $l \not\triangleright t$, $\mathrm{root}(t)$ is defined symbol

DEFINITION

dependency graph of $\mathcal{R}$

- nodes are dependency pairs of $\mathcal{R}$
- draw arrow from $s^\sharp \to t^\sharp$ to $u^\sharp \to v^\sharp$
  if $t\sigma \to_{\mathcal{R}}^* u\tau$ for some substitution σ and τ

dependency graph $\approx$ recursive call graph

Dependency Pair Methods

dependency pair method (informally):

- termination is shown by termination of recursive functions
  Giesl, Arts, Ohlebusch JSC 2002

powerful refinements

- argument filterings Arts, Giesl TCS 2000

- usable rules Hirokawa, Middeldorp IC 2007
  Giesl, Thiemann, Schneider-Kamp, Falke JAR 2007

Argument Filterings

- argument filtering $\pi(f^{(n)})$ is of form i or $[i_1, \ldots, i_m]$
  $(0 \leq i \leq n,\ 1 \leqslant i_1 < \cdots < i_m \leqslant n)$
- extension to term

$$\pi(t) = \begin{cases} t & \text{if } t \text{ is variable} \\ \pi(t_i) & \text{if } t = f(t_1, \ldots, t_n),\ \pi(f) = i \\ f(\pi(t_{i_1}), \ldots, \pi(t_{i_m})) & \text{if } t = f(t_1, \ldots, t_n),\ \pi(f) = [i_1, \ldots, i_m] \end{cases}$$

EXAMPLE

for $\pi(\mathsf{f}) = [2]$, $\pi(\mathsf{g}) = 1$

$$\pi(\mathsf{f}(\mathsf{g}(x), \mathsf{g}(\mathsf{g}(x)))) = \mathsf{f}(\mathsf{g}(x), \mathsf{g}(\mathsf{g}(x))) = \mathsf{f}(x)$$

Eratosthenes' sieve

TRS $\mathcal{R}$

$$
\begin{array}{rlrl}
1\colon & x - 0 \to x & 10\colon & \mathsf{filter}(x, \mathsf{nil}) \to \mathsf{nil} \\
2\colon & \mathsf{s}(x) - \mathsf{s}(y) \to x - y & 11\colon & \mathsf{filter}(x, y : ys) \to \mathsf{if\_filter}(x \mid y, x, y : ys) \\
3\colon & 0 \leqslant y \to \mathsf{true} & 12\colon & \mathsf{if\_filter}(\mathsf{true}, x, y : ys) \to \mathsf{filter}(x, ys) \\
4\colon & \mathsf{s}(x) \leqslant 0 \to \mathsf{false} & 13\colon & \mathsf{if\_filter}(\mathsf{false}, x, y : ys) \to y : \mathsf{filter}(x, ys) \\
5\colon & \mathsf{s}(x) \leqslant \mathsf{s}(y) \to x \leqslant y & 14\colon & \mathsf{sieve}(\mathsf{nil}) \to \mathsf{nil} \\
6\colon & \mathsf{if}(\mathsf{true}, x, y) \to x & 15\colon & \mathsf{sieve}(x : xs) \to x : \mathsf{sieve}(\mathsf{filter}(x, xs)) \\
7\colon & \mathsf{if}(\mathsf{false}, x, y) \to y & & \\
8\colon & x \mid 0 \to \mathsf{true} & & \\
9\colon & \mathsf{s}(x) \mid \mathsf{s}(y) \to \mathsf{if}(x \leqslant y, \mathsf{s}(x) \mid (y - x), \mathsf{false}) & &
\end{array}
$$

Eratosthenes' sieve

$$
\begin{array}{rl}
1: & x - 0 \to x \\
2: & \mathsf{s}(x) - \mathsf{s}(y) \to x - y \\
3: & 0 \leqslant y \to \mathsf{true} \\
4: & \mathsf{s}(x) \leqslant 0 \to \mathsf{false} \\
5: & \mathsf{s}(x) \leqslant \mathsf{s}(y) \to x \leqslant y \\
6: & \mathsf{if}(\mathsf{true}, x, y) \to x \\
7: & \mathsf{if}(\mathsf{false}, x, y) \to y \\
8: & x \mid 0 \to \mathsf{true} \\
9: & \mathsf{s}(x) \mid \mathsf{s}(y) \to \mathsf{if}(x \leqslant y, \mathsf{s}(x) \mid (y - x), \mathsf{false})
\end{array}
$$

$$
\begin{array}{rl}
10: & \mathsf{filter}(x, \mathsf{nil}) \to \mathsf{nil} \\
11: & \mathsf{filter}(x, y : ys) \to \mathsf{if\_filter}(x \mid y, x, y : ys) \\
12: & \mathsf{if\_filter}(\mathsf{true}, x, y : ys) \to \mathsf{filter}(x, ys) \\
13: & \mathsf{if\_filter}(\mathsf{false}, x, y : ys) \to y : \mathsf{filter}(x, ys) \\
14: & \mathsf{sieve}(\mathsf{nil}) \to \mathsf{nil} \\
15: & \mathsf{sieve}(x : xs) \to x : \mathsf{sieve}(\mathsf{filter}(x, xs))
\end{array}
$$

termination of $\mathcal{R}$ is established by termination of

- recursion of $-$-rules
- recursion of $\leq$-rules
- recursion of $\mid$-rules
- mutual recursion of filter- and if_filter-rules, and
- recursion of sieve-rules (hardest)

Termination of sieve

dependency pair $\mathcal{C} \subseteq >$

$$\mathsf{sieve}^\sharp(x : xs) > \mathsf{sieve}^\sharp(\mathsf{filter}(x, xs))$$

usable rules $\mathcal{U}(\mathcal{C}) \subseteq \gtrsim$

$$\boxed{1}: \quad x - 0 \gtrsim x$$
$$\boxed{2}: \quad \mathsf{s}(x) - \mathsf{s}(y) \gtrsim x - y$$
$$\boxed{3}: \quad 0 \leqslant y \gtrsim \mathsf{true}$$
$$\boxed{4}: \quad \mathsf{s}(x) \leqslant 0 \gtrsim \mathsf{false}$$
$$\boxed{5}: \quad \mathsf{s}(x) \leqslant \mathsf{s}(y) \gtrsim x \leqslant y$$
$$\boxed{6}: \quad \mathsf{if}(\mathsf{true}, x, y) \gtrsim x$$
$$\boxed{7}: \quad \mathsf{if}(\mathsf{false}, x, y) \gtrsim y$$
$$\boxed{8}: \quad x \mid 0 \gtrsim \mathsf{true}$$
$$\boxed{9}: \quad \mathsf{s}(x) \mid \mathsf{s}(y) \gtrsim \mathsf{if}(x \leqslant y, \mathsf{s}(x) \mid (y - x), \mathsf{false})$$
$$\boxed{10}: \quad \mathsf{filter}(x, \mathsf{nil}) \gtrsim \mathsf{nil}$$
$$\boxed{11}: \quad \mathsf{filter}(x, y : ys) \gtrsim \mathsf{if\_filter}(x \mid y, x, y : ys)$$
$$\boxed{12}: \quad \mathsf{if\_filter}(\mathsf{true}, x, y : ys) \gtrsim \mathsf{filter}(x, ys)$$
$$\boxed{13}: \quad \mathsf{if\_filter}(\mathsf{false}, x, y : ys) \gtrsim y : \mathsf{filter}(x, ys)$$
$$14: \quad \mathsf{sieve}(\mathsf{nil}) \gtrsim \mathsf{nil}$$
$$15: \quad \mathsf{sieve}(x : xs) \gtrsim x : \mathsf{sieve}(\mathsf{filter}(x, xs))$$

no suitable LPO, so apply argument filtering

Termination of sieve

dependency pair $\pi(\mathcal{C}) \subseteq >$

$$\mathsf{sieve}^\sharp(x : xs) > \mathsf{sieve}^\sharp(\mathsf{filter}(x, xs))$$

usable rules $\pi(\mathcal{U}(\mathcal{C}, \pi)) \subseteq \gtrsim$

$$1: \quad x - 0 \gtrsim x$$
$$2: \quad \mathsf{s}(x) - \mathsf{s}(y) \gtrsim x - y$$
$$3: \quad 0 \leqslant y \gtrsim \mathsf{true}$$
$$4: \quad \mathsf{s}(x) \leqslant 0 \gtrsim \mathsf{false}$$
$$5: \quad \mathsf{s}(x) \leqslant \mathsf{s}(y) \gtrsim x \leqslant y$$
$$6: \quad \mathsf{if}(\mathsf{true}, x, y) \gtrsim x$$
$$7: \quad \mathsf{if}(\mathsf{false}, x, y) \gtrsim y$$
$$8: \quad x \shortmid 0 \gtrsim \mathsf{true}$$
$$9: \quad \mathsf{s}(x) \shortmid \mathsf{s}(y) \gtrsim \mathsf{if}(x \leqslant y, \mathsf{s}(x) \shortmid (y - x), \mathsf{false})$$
$$10: \quad \mathsf{filter}(x, \mathsf{nil}) \gtrsim \mathsf{nil}$$
$$11: \quad \mathsf{filter}(x, y : ys) \gtrsim \mathsf{if\_filter}(x \shortmid y, x, y : ys)$$
$$12: \quad \mathsf{if\_filter}(\mathsf{true}, x, y : ys) \gtrsim \mathsf{filter}(x, ys)$$
$$13: \quad \mathsf{if\_filter}(\mathsf{false}, x, y : ys) \gtrsim y : \mathsf{filter}(x, ys)$$
$$14: \quad \mathsf{sieve}(\mathsf{nil}) \gtrsim \mathsf{nil}$$
$$15: \quad \mathsf{sieve}(x : xs) \gtrsim x : \mathsf{sieve}(\mathsf{filter}(x, xs))$$

use LPO with empty precedence

how to find suitable argument filtering?

Finding Suitable Argument Filtering

argument filtering and usable rules are powerful

$$\pi(\mathcal{C}) \cap > \neq \varnothing \quad \wedge \quad \pi(\mathcal{U}(\mathcal{C}, \pi) \cup \mathcal{C}) \subseteq \gtrsim$$

PROBLEM

- difficult to find suitable argument filtering:
  $4,234,032$ possible argument filterings for TRS of sieve

- simultaneous constraint between usable rules and argument filtering is problematic

SOLUTION

encode simultaneous constraints into propositional formulas

Extended Diophantine Constraints

SYNTAX

$$c ::= b \mid r \mid \neg c \mid c_1 \vee c_2 \mid c_1 \wedge c_2$$
$$r ::= p_1 \geq p_2 \mid p_1 \leq p_2 \mid p_1 > p_2 \mid p_1 < p_2 \mid p_1 = p_2 \mid p_1 \neq p_2$$
$$p ::= x \mid n \mid p_1 + p_1 \mid p_1 \times p_2$$

where $b \in \mathcal{B}$, $x \in \mathcal{V}$ and $n \in \mathbb{Z}$

SEMANTICS

- $\alpha, \beta \vDash b$ if $\beta(b) = \text{true}$
- $\alpha, \beta \vDash p_1 \geq p_2$ if $[\alpha]_{\mathbb{N}}(p_1) \geq [\alpha]_{\mathbb{N}}(p_2)$
- $\alpha, \beta \vDash \neg c$ if $\alpha, \beta \vDash c$ does not hold
- $\alpha, \beta \vDash c_1 \vee c_2$ if $\alpha, \beta \vDash c_1$ or $\alpha \vDash c_2$
- $\alpha, \beta \vDash c_1 \wedge c_2$ if $\alpha, \beta \vDash c_1$ and $\alpha, \beta \vDash c_2$

- $\vDash c$ if $\exists \alpha : \mathcal{V} \to \mathbb{Z}, \beta : \mathcal{B} \to \{\text{true}, \text{false}\}.\ \alpha, \beta \vDash c$

Encoding Argument Filtering Constraints

for n-ary function f introduce propositional variables

$$f^{\mathsf{list}}, f^1, f^2, \ldots, f^n$$

- replace $\pi(f) = i$ by

$$\neg f^{\mathsf{list}} \wedge f^i \wedge \bigwedge_{k \neq i} \neg f^k$$

- replace $\pi(f) = [i_1, \ldots, i_m]$ by

$$f^{\mathsf{list}} \wedge \bigwedge_{k \in \{i_1, \ldots, i_m\}} f^k \wedge \bigwedge_{k \notin \{i_1, \ldots, i_m\}} \neg f^k$$

- add valid form condition (e.g. $\neg f^{\mathsf{list}} \wedge f^1 \wedge f^2$ is invalid)

$$f^{\mathsf{list}} \vee \bigvee_{i \in \{1, \ldots, n\}} \left(f^i \wedge \bigwedge_{k \neq i} \neg f^i \right)$$

OBSERVE

$\pi(s) >_{\mathsf{lpo}} \pi(t)$ is extended Diophantine constraint

EXAMPLE

$$
\begin{aligned}
& \pi(\mathsf{f}(\mathsf{a}, x)) >_{\mathsf{lpo}} \pi(\mathsf{c}) \ \wedge \ \text{(valid form condition)} \\
\Leftrightarrow \ & (\mathsf{f}^{\mathsf{list}} \ \wedge \ \mathsf{c}^{\mathsf{list}} \ \wedge \ \mathsf{f} > \mathsf{c}) \vee \\
& (\neg \mathsf{f}^{\mathsf{list}} \ \wedge \ ((\mathsf{a}^1 \ \wedge \ \mathsf{c}^{\mathsf{list}} \ \wedge \ \mathsf{a}^{\mathsf{list}} \ \wedge \ \mathsf{a} > \mathsf{c}) \vee (\mathsf{a}^2 \ \wedge \ \mathsf{c}^{\mathsf{list}} \ \wedge \ \bot))) \wedge \\
& (\mathsf{f}^{\mathsf{list}} \ \vee \ (\mathsf{f}^1 \ \wedge \ \neg \mathsf{f}^2) \ \vee \ (\neg \mathsf{f}^1 \wedge \mathsf{f}^2)) \ \wedge \ \mathsf{a}^{\mathsf{list}} \ \wedge \ \mathsf{b}^{\mathsf{list}})
\end{aligned}
$$

solutions of this constraint

- $\mathsf{f}^{\mathsf{list}} = \mathsf{f}^1 = \mathsf{f}^2 = \mathsf{a}^{\mathsf{list}} = \mathsf{c}^{\mathsf{list}} = \mathsf{true}, \mathsf{f} = 1, \mathsf{a} = \mathsf{c} = 0$
  $\pi(f) = [1; 2], \pi(\mathsf{a}) = \pi(\mathsf{c}) = [], \mathsf{f} > \mathsf{a}, \mathsf{f} > \mathsf{c}$
- $\mathsf{f}^{\mathsf{list}} = \mathsf{f}^2 = \mathsf{false}, \mathsf{f}^1 = \mathsf{a}^{\mathsf{list}} = \mathsf{c}^{\mathsf{list}} = \mathsf{true}, \mathsf{a} = 1, \mathsf{f} = \mathsf{c} = 0$
  $\pi(f) = 1, \pi(\mathsf{a}) = \pi(\mathsf{c}) = [], \mathsf{a} > \mathsf{f}, \mathsf{a} > \mathsf{c}$
- ...

Encoding Usable Rule Criterion

for function f, introduce propositional variable

$$U_f$$

constraint

$$\pi(\mathcal{U}(\mathcal{C},\pi)\cup\mathcal{C})\subseteq\gtrsim$$

is encoded to

$$\bigwedge_{l\to r\in\mathcal{C}}\big(U_{\mathrm{root}(l)}\wedge(\pi(l)\gtrsim\pi(r))\big)\ \wedge\ \bigwedge_{l\to r\in\mathcal{R}}\big(U_{\mathrm{root}(l)}\to(\pi(l)\gtrsim\pi(r))\big)\ \wedge$$

$$\bigwedge_{l\to r\in\mathcal{R}\cup\mathcal{C}}\left(U_{\mathrm{root}(l)}\to\bigwedge_{\substack{p\in\mathcal{P}\mathsf{os}_{\mathcal{F}}(r)\\ \mathrm{root}(r|_p)\text{ is defined}}}\left(\bigwedge_{q,\,i\colon qi\leqslant p}\mathrm{root}(r|_q)^i\to U_{\mathrm{root}(r|_p)}\right)\right)$$

Experimental Results for LPO

dependency pair method with LPO

	IC 2005	Dio/SAT solving	
	$\mathcal{U}(\mathcal{C})$	$\mathcal{U}(\mathcal{C})$	$\mathcal{U}(\mathcal{C},\pi)$
# of success	188	188	247
# of timeout (60 s)	5	0	0
total time (s)	338	61	120

termination tool: TTT with Minisat problem set: 773 TRSs from Termination Problem Database
PC: Intel Xeon 2.40 GHz 512 MB memory

Experimental Results for Polynomial Orders

dependency pair method with linear form polynomial interpretation orders

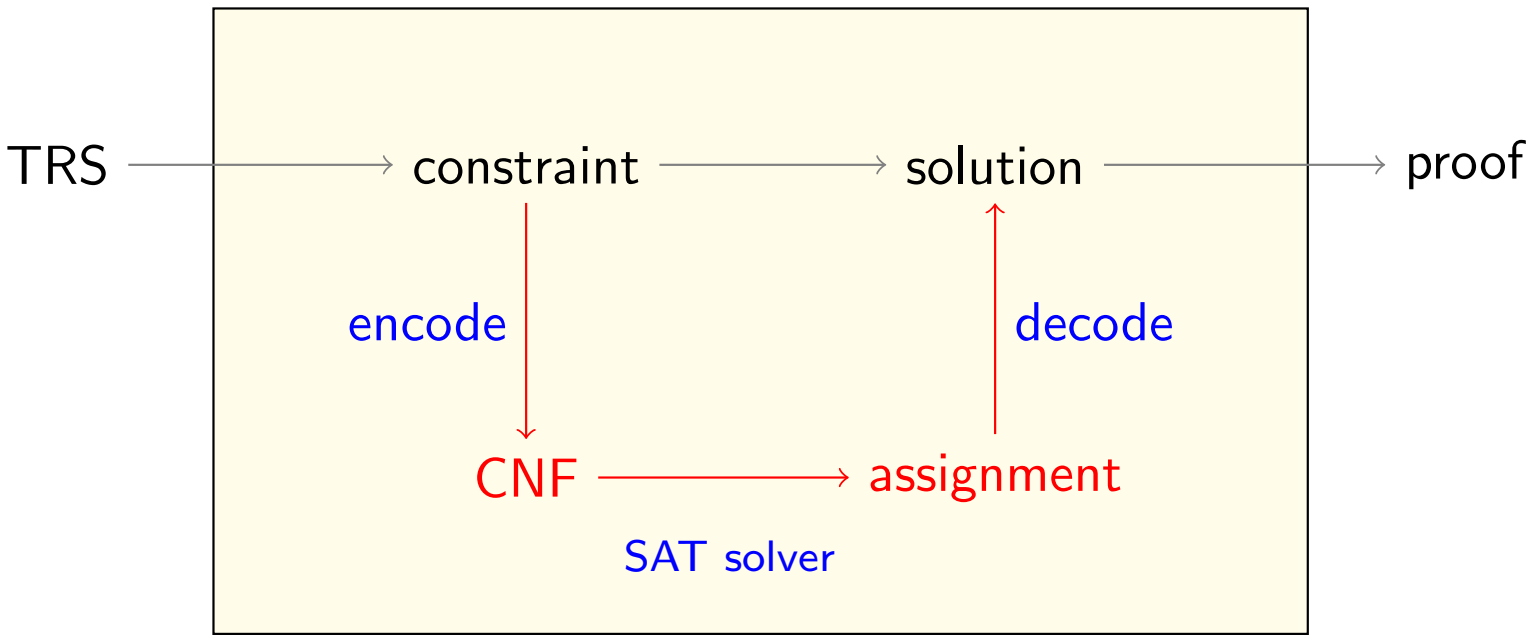
$$f_{\mathbb{N}}(x_1, \ldots, x_n) = a_1 x_1 + \cdots + a_n x_n + b \qquad a_1, \ldots, a_n, b \in \{0, 1, 2\}$$

	Gen & Test	Dio/SAT solving
	$\mathcal{U}(\mathcal{C})$	$\mathcal{U}(\mathcal{C}, \pi)$
# of success	247	431
# of timeout (60 s)	32	0
total time (s)	2568	91

termination tools: AProVE and TTT with Minisat problem set: 865 TRSs from Termination Problem Database
PC: AMD Athlon 64 2.20 GHz 512 MB memory

Conclusion

currently all termination tools use SAT solver or Diophantine constraint solver



References

Annov, Codish, Stuckey	RTA 2006
Codish, Schneider-Kamp, Lagoon, Thiemann, Giesl	LPAR 2006
Contejean, Marché, Tomás, Urbain	JAR 2005
Fuhs, Giesl, Middeldorp, Schneider-Kamp, Thiemann, Zankl	SAT 2007
Kurihara, Kondo	IEA/AIE 2004
Zankl, Middeldorp, Hirokawa	SOFSEM 2007