Satisfiability and Term Rewriting

Akihisa Yamada @ AIST Tokyo Waterfront

Term Rewriting

- Variables $x, y, z, \ldots \in \mathcal{V}$
- Functions $f, g, h, \ldots \in \mathcal{F}$, with arity $ar : \mathcal{F} \to \mathbb{N}$
- Terms $s, t, l, r, \ldots \in \mathcal{T} ::= x \mid f\left(s_1, \ldots, s_{ar(f)}\right)$
- Substitutions $\sigma, \tau, \ldots : \mathcal{V} \to \mathcal{T}$
- **Term Rewrite System (TRS)** $\mathcal{R}$ is a set of rules $l \to r$

  meaning "$\mathcal{R}$ **rewrites** $l\sigma$ **to** $r\sigma$" (under any context)

  $C[l\sigma] \to_{\mathcal{R}} C[r\sigma]$
- $\mathcal{R}$ is **terminating**: there is no $s_1 \to_{\mathcal{R}} s_2 \to_{\mathcal{R}} s_3 \to_{\mathcal{R}} \cdots$

SMT for Term Rewriting

Proving termination

- Initialize:

$$\frac{(\mathcal{P}, \mathcal{R})}{\mathcal{R} \text{ is terminating}}$$ where $\mathcal{P}$ is the set of *dependency pairs*

- Divide:

$$\frac{(\mathcal{C}_1, \mathcal{R}) \ \cdots \ (\mathcal{C}_n, \mathcal{R})}{(\mathcal{P}, \mathcal{R})}$$ where $\mathcal{C}_1, \dots, \mathcal{C}_n$ are the SCCs of *dependency graph*

- Concur:

$$\frac{\left(\mathcal{C} \setminus >_{\mathrm{WPO}(\mathcal{A}, \succ, \pi)}, \mathcal{R}\right)}{(\mathcal{C}, \mathcal{R})}$$ if $\mathcal{C} \cup \mathcal{R} \subseteq \geq_{\mathrm{WPO}(\mathcal{A}, \succ, \pi)}$

WPO [Y+, SCP 2014] as SMT

Definition: $\mathrm{WPO}(\mathcal{A}, \succ, \pi)$ is defined by:

$s = f(s_1, \ldots, s_n) \sqsupseteq_{\mathrm{WPO}} t = g(t_1, \ldots, t_m)$ iff

1. $\mathcal{A}[\![s]\!] > \mathcal{A}[\![t]\!]$ or
2. $\mathcal{A}[\![s]\!] \geq \mathcal{A}[\![t]\!]$ and
   - **a.** $\exists i \in \pi(f).\, s_i \sqsupseteq_{\mathrm{WPO}} t$; or
   - **b.** $\forall j \in \pi(g).\, s \sqsupset_{\mathrm{WPO}} t_j$ and
     - **i.** $f \succ g$ or
     - **ii.** $f \succsim g$ and $\pi_f(s_1, \ldots, s_n) \sqsupseteq^{\mathrm{lex}}_{\mathrm{WPO}} \pi_g(t_1, \ldots, t_m)$

$(s \sqsupseteq_{\mathrm{wpo}} t) :=$

1. $\mathcal{A}[\![s]\!] > \mathcal{A}[\![t]\!] \ \lor$

2. $\mathcal{A}[\![s]\!] \geq \mathcal{A}[\![t]\!] \ \land ($

a. $\left(\bigvee_{i \in \pi(f)} s_i \sqsupseteq_{\mathrm{wpo}} t\right) \lor$

b. $\left(\bigwedge_{j \in \pi(g)} s \sqsupset_{\mathrm{wpo}} t_j\right) \land ($

i. $f \succ g \ \lor$

ii. $f \succsim g \ \land$

$\pi_f(s_1, \ldots, s_n) \sqsupseteq^{\mathrm{lex}}_{\mathrm{wpo}} \pi_g(t_1, \ldots, t_m)) \ )$

Can be huge. Be lazy

Lazy SMT encoding

Definition: $\mathrm{WPO}(\mathcal{A}, \succ, \pi)$ is defined by:
$s = f(s_1, \ldots, s_n) \sqsupseteq_{\mathrm{WPO}} t = g(t_1, \ldots, t_m)$ iff

1. $\mathcal{A}[\![s]\!] > \mathcal{A}[\![t]\!]$ or
2. $\mathcal{A}[\![s]\!] \geq \mathcal{A}[\![t]\!]$ and
   - a. $\exists i \in \pi(f).\, s_i \sqsupseteq_{\mathrm{WPO}} t$; or
   - b. $\forall j \in \pi(g).\, s \sqsupset_{\mathrm{WPO}} t_j$ and
     - i. $f \succ g$ or
     - ii. $f \succsim g$ and $\pi_f(s_1, \ldots, s_n) \sqsupseteq^{\mathrm{lex}}_{\mathrm{WPO}} \pi_g(t_1, \ldots, t_m)$

$(s \sqsupseteq_{\mathrm{wpo}} t) :=$

1. $\mathcal{A}[\![s]\!] > \mathcal{A}[\![t]\!] \lor$
2. $\mathcal{A}[\![s]\!] \geq \mathcal{A}[\![t]\!] \land (\boldsymbol{\lambda\_.}$
   - a. $\left(\bigvee_{i \in \pi(f)} s_i \sqsupseteq_{\mathrm{wpo}} t\right) \lor$
   - b. $\left(\bigwedge_{j \in \pi(g)} s \sqsupset_{\mathrm{wpo}} t_j\right) \land (\boldsymbol{\lambda\_.}$
     - i. $f \succ g \lor$
     - ii. $f \succsim g \land \boldsymbol{\lambda\_.}$ $\pi_f(s_1, \ldots, s_n) \sqsupseteq^{\mathrm{lex}}_{\mathrm{wpo}} \pi_g(t_1, \ldots, t_m)))$

$$\phi \land (\lambda\_.\psi) \hookrightarrow \begin{cases} \text{False} & \text{if } \phi \text{ is obviously false} \\ \phi \land \psi & \text{otherwise} \end{cases}$$

Further ideas

- Context-aware encoding?

  $$\phi_1 \wedge \Big(\ldots \wedge \big(\phi_{n-1} \wedge \big(\phi_n \wedge (\lambda\_.\, \psi) \big) \vee \cdots \big) \Big) \hookrightarrow \phi_1 \wedge \big(\ldots \wedge (\phi_{n-1} \vee \cdots) \big)$$

  if $\phi_1 \wedge \cdots \wedge \phi_n$ is (trivially) unsat
- Encode by need?
  - SMT solver might not need to know ψ
  - e.g. $x > 0 \vee (\lambda\_.\, \psi)$ is SAT, whatever ψ

Satisfiability modulo rewriting

Reachability (in term rewriting)

- **Example**: Let $\mathcal{R}$ be a rewrite system (or functional program)

$$\begin{aligned} \mathrm{hd}(\mathrm{Cons}(x, xs)) &\to x \\ \mathrm{hd}(\mathrm{Nil}) &\to \mathrm{error}(\text{"nil access"}) \end{aligned}$$

- **Question**:
  - From $\mathrm{hd}(x)$, is $\mathrm{error}(y)$ reachable?
- **Classic answer**:
  - **NO!** because $\mathrm{hd}(x) \not\to_{\mathcal{R}}^{*} \mathrm{error}(y)$
- **Our answer**:
  - **SAT!** solution $[x \mapsto \mathrm{Nil}, y \mapsto \text{"nil access"}]$

Our motivation stems from...

- **termination analysis**:

  $$\mathrm{String}(x) + y \to \mathrm{int\_of\_string}(x) + y \;\in \mathcal{R}_1$$

  is non-looping, if

  $\mathrm{int\_of\_string}(x) \twoheadrightarrow \mathrm{String}(x')$ is UNSAT modulo $\mathcal{R}_1$

- **confluence analysis** for conditional rewriting:

  $$\mathrm{sgn}(x) \to 1 \quad \Leftarrow \; x > 0 \twoheadrightarrow \mathrm{True},$$
  $$\mathrm{sgn}(x) \to -1 \;\Leftarrow\; x < 0 \twoheadrightarrow \mathrm{True} \;\in \mathcal{R}_2$$

  are harmless, if

  $x > 0 \twoheadrightarrow \mathrm{True} \;\wedge\; x < 0 \twoheadrightarrow \mathrm{True}$ is UNSAT modulo $\mathcal{R}_2$

Problem is not new

- ... but not so old
  - called "(in)feasibility" [Lucas & Guitiérrez 2018]
- SAT/SMT friendly formulation [Sternagel & Yamada, TACAS 2019]
  - look-ahead reachability
  - implementations
- Contribution of [Yamada, IJCAR 2022]
  - a model-based UNSAT

Reachability constraint satisfaction

- Syntax
  $\phi, \psi, \ldots ::= s \twoheadrightarrow t \mid \top \mid \bot \mid \phi \wedge \psi \mid \phi \vee \psi \mid \phi \Rightarrow \psi \mid \neg\phi \mid \exists x.\phi \mid \forall x.\phi$
- Semantics
  - substitution σ **satisfies** ϕ **modulo** rewrite system $\mathcal{R}$ $\quad \sigma \vDash_{\mathcal{R}} \phi$
    - $\sigma \vDash_{\mathcal{R}} s \twoheadrightarrow t \iff s\sigma \twoheadrightarrow_{\mathcal{R}} t\sigma$
    - $\sigma \vDash_{\mathcal{R}} \phi \wedge \psi \iff (\sigma \vDash_{\mathcal{R}} \phi) \wedge (\sigma \vDash_{\mathcal{R}} \psi)$
    - …
  - ϕ is **satisfiable modulo** $\mathcal{R}$: $\quad \mathrm{SAT}_{\mathcal{R}}(\phi)$
    - there exists σ such that $\sigma \vDash_{\mathcal{R}} \phi$
  - ϕ and ψ are **equisatisfiable modulo** $\mathcal{R}$: $\quad \phi \equiv_{\mathcal{R}} \psi$
    - $\mathrm{SAT}_{\mathcal{R}}(\phi) \iff \mathrm{SAT}_{\mathcal{R}}(\psi)$

Co-rewrite pair

Definition: $(\succcurlyeq, \sqsubset)$ is a **co-rewrite pair** if

- $\succcurlyeq$ and $\sqsubset$ are closed under substitutions ($s \succcurlyeq t \Longrightarrow s\theta \succcurlyeq t\theta$)
- $\succcurlyeq$ is closed under contexts ($s \succcurlyeq t \Longrightarrow C[s] \succcurlyeq C[t]$)
- $\succcurlyeq$ is a quasi-order
- $\succcurlyeq \cap \sqsubset = \emptyset$

Theorem [Y, IJCAR 2022]:

$s \twoheadrightarrow t$ is $\mathcal{R}$-unsat **iff** there's a co-rewrite pair $\langle \succcurlyeq, \sqsubset \rangle$ s.t. $\mathcal{R} \subseteq \succcurlyeq$ and $s \sqsubset t$

Proposition: WPO forms a co-rewrite pair (under mild modification)

Clause refuter

Corollary [Y, IJCAR 2022]: $s \twoheadrightarrow t$ is $\mathcal{R}$-unsat **iff**
there is $\mathcal{A}, \succcurlyeq, \pi$ s.t. $\mathcal{R} \subseteq \geq_{\mathrm{WPO}(\mathcal{A},\succcurlyeq,\pi)} \wedge s \sqsubset_{\mathrm{WPO}(\mathcal{A},\succcurlyeq,\pi)} t$

(Almost) Corollary [new]: $s_1 \twoheadrightarrow t_1 \wedge \cdots \wedge s_n \twoheadrightarrow t_n$ is $\mathcal{R}$-unsat **iff**
there is $\mathcal{A}, \succcurlyeq, \pi$ s.t.
$\mathcal{R} \subseteq \geq_{\mathrm{WPO}(\mathcal{A},\succcurlyeq,\pi)} \wedge \left(s_1 \sqsubset_{\mathrm{WPO}(\mathcal{A},\succcurlyeq,\pi)} t_1 \vee \cdots \vee s_n \sqsubset_{\mathrm{WPO}(\mathcal{A},\succcurlyeq,\pi)} t_n\right)$

Constructors evaluate to constructors

- **Observation** (folklore):

  if $\mathrm{Cons}(\ldots) \to \cdots \notin \mathcal{R}$ then

  - $\mathrm{Cons}(s, ss) \twoheadrightarrow_{\mathcal{R}} \mathrm{Cons}(t, ts)$ iff $s \twoheadrightarrow_{\mathcal{R}} t$ and $ss \twoheadrightarrow_{\mathcal{R}} ts$
  - $\mathrm{Cons}(s, ss) \twoheadrightarrow_{\mathcal{R}} \mathrm{Nil}$ never happen

- In our language:
  - $\mathrm{Cons}(s, ss) \twoheadrightarrow \mathrm{Cons}(t, ts) \equiv_{\mathcal{R}} s \twoheadrightarrow t \land ss \twoheadrightarrow ts$
  - $\mathrm{Cons}(s, ss) \twoheadrightarrow \mathrm{Nil} \equiv_{\mathcal{R}} \bot$

- **Proposition**: If $f(\ldots) \to \cdots \notin \mathcal{R}$, then

  $f(s_1, \ldots, s_n) \twoheadrightarrow f(t_1, \ldots, t_n) \equiv_{\mathcal{R}} s_1 \twoheadrightarrow t_1 \land \cdots \land s_n \twoheadrightarrow t_n$

  $f(\ldots) \twoheadrightarrow g(\ldots) \equiv_{\mathcal{R}} \bot$ if $f \neq g$

Constructors evaluate to constructors

- **<u>Observation</u>** (folklore):

  if $\mathrm{Cons}(\dots) \to \cdots \notin \mathcal{R}$ then
  - $\mathrm{Cons}(s, ss) \twoheadrightarrow_{\mathcal{R}} \mathrm{Cons}(t, ts)$ iff $s \twoheadrightarrow_{\mathcal{R}} t$ and $ss \twoheadrightarrow_{\mathcal{R}} ts$
  - $\mathrm{Cons}(s, ss) \twoheadrightarrow_{\mathcal{R}} \mathrm{Nil}$ never happen
- In our language:
  - $\mathrm{Cons}(s, ss) \twoheadrightarrow \mathrm{Cons}(t, ts) \equiv_{\mathcal{R}} s \twoheadrightarrow t \land ss \twoheadrightarrow ts$
  - $\mathrm{Cons}(s, ss) \twoheadrightarrow \mathrm{Nil} \equiv_{\mathcal{R}} \bot$
- **<u>Proposition</u>**: If $f(\dots) \to \cdots \notin \mathcal{R}$, then

  $f(s_1, \dots, s_n) \twoheadrightarrow g(t_1, \dots, t_m) \ \equiv_{\mathcal{R}}$

$$f(s_1, \dots, s_n) \twoheadrightarrow^{>\epsilon} g(t_1, \dots, t_m) \ := \begin{cases} s_1 \twoheadrightarrow t_1 \land \cdots \land s_n \twoheadrightarrow t_n & \text{if } f = g \\ \bot & \text{if } f \neq g \end{cases}$$

Look-ahead

- $\mathcal{R} = \{\, 0 > x \to \text{False},\ s(x) > 0 \to \text{True},\ s(x) > s(y) \to x > y \,\}$

Is $0 > z \twoheadrightarrow \text{True}$ SAT modulo $\mathcal{R}$?

Look-ahead

- $\mathcal{R} = \{\, 0 > x \to \mathrm{False},\ s(x) > 0 \to \mathrm{True},\ s(x) > s(y) \to x > y \,\}$

Is $0 > z \twoheadrightarrow \mathrm{True}$ SAT?

Look-ahead

- $\mathcal{R} = \{\, 0 > x \to \text{False},\ \ s(x) > 0 \to \text{True},\ \ s(x) > s(y) \to x > y \,\}$

Is $0 > z \twoheadrightarrow \text{True}$ SAT?

Look-ahead

- $\mathcal{R} = \{\, 0 > x \to \text{False},\ s(x) > 0 \to \text{True},\ s(x) > s(y) \to x > y \,\}$

Is $0 > z \twoheadrightarrow \text{True}$ SAT?

- <u>**Theorem**</u> [Sternagel & Y., TACAS 2019]:

$f(s_1 \ldots) \twoheadrightarrow g(t_1 \ldots) \equiv_{\mathcal{R}}$

$$f(s_1 \ldots) \twoheadrightarrow^{>\epsilon} g(t_1 \ldots) \lor \bigvee_{\substack{l \to r \in \mathcal{R} \\ \text{renamed}}} f(s_1 \ldots) \twoheadrightarrow^{>\epsilon} l \land r \twoheadrightarrow g(t_1 \ldots)$$

Look-ahead example

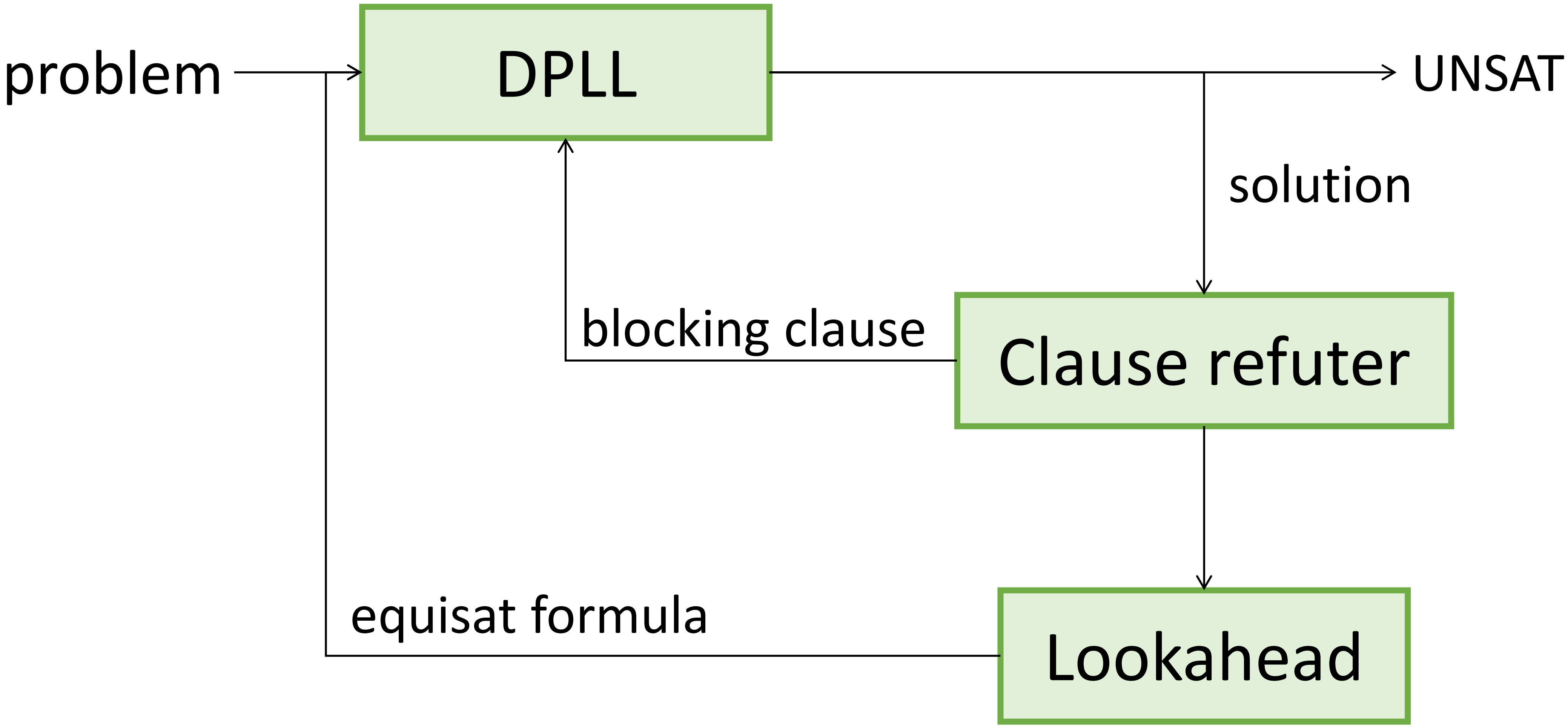
- $\mathcal{R} = \{\, 0 > x \rightarrow \text{False},\ s(x) > 0 \rightarrow \text{True},\ s(x) > s(y) \rightarrow x > y \,\}$
- $0 > z \twoheadrightarrow \text{True}$

$\equiv_{\mathcal{R}}$ $0 \twoheadrightarrow 0 \wedge z \twoheadrightarrow x \wedge$ ~~$\text{False} \twoheadrightarrow \text{True}$~~ $\vee$

~~$0 \twoheadrightarrow s(x)$~~ $\wedge\, z \twoheadrightarrow 0 \wedge \text{True} \twoheadrightarrow \text{True} \vee$

~~$0 \twoheadrightarrow s(x)$~~ $\wedge\, z \twoheadrightarrow s(y) \wedge x > y \twoheadrightarrow \text{True}$

Idea:


problem
DPLL
UNSAT
solution
blocking clause
Clause refuter
equisat formula
Lookahead

Conclusions

- Use of SMT for rewriting
  - Context-aware encoding?
  - Encoding by need?
- Satisfiability modulo rewriting
  - DPLL(R)?