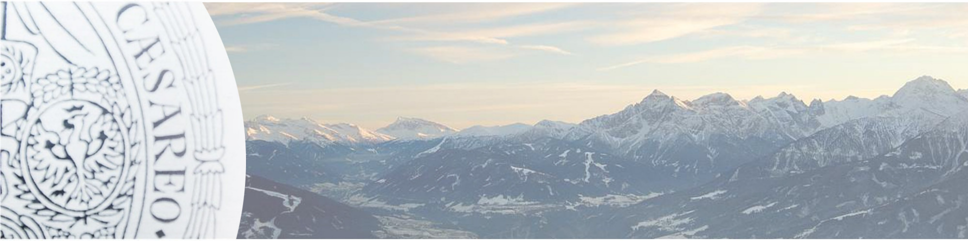
universität innsbruck



SAT and SMT Solving

Sarah Winkler

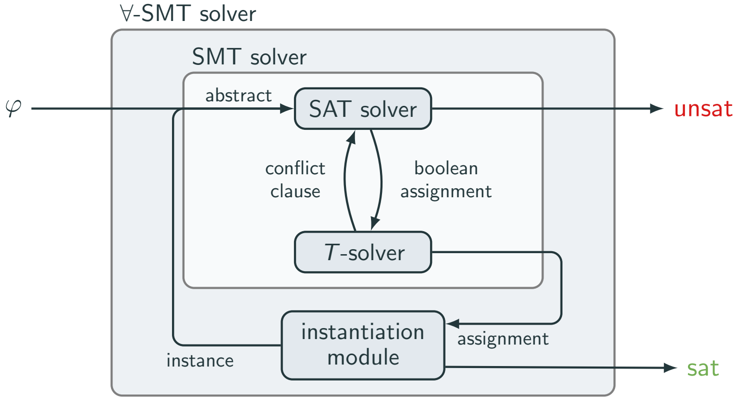
KRDB
Department of Computer Science
Free University of Bozen-Bolzano

lecture 13
WS 2022

Outline

- Summary of Last Week
- Instantiation Techniques
- More on SAT and SMT
- Test
- Evaluation

Instantiation Framework



- split φ into
  - literals φ_Q with quantifiers
  - literals φ_E without quantifiers
- instantiation module generates instances of φ_Q to extend φ_E

SMT solver is in general no decision procedure in presence of ∀ quantifiers

Skolemization

1. bring formula into prenex form
2. replace $\forall x_1, \ldots, x_k \exists y\ \psi[y]$ by $\forall x_1, \ldots, x_k\ \psi[\mathsf{f}(x_1, \ldots, x_k)]$ for fresh f

   until no existential quantifiers left

Theorem

can consider formulas of shape $\forall x_1, \ldots, x_n\ \varphi[x_1, \ldots, x_n]$

if φ' is skolemization of φ then φ and φ' are equisatisfiable

Definition

Herbrand instance of Skolem formula $\forall x_1, \ldots, x_n\ \varphi[x_1, \ldots, x_n]$ is $\varphi[t_1, \ldots, t_n]$ where t_i is term over signature of φ

Definition

set of function symbols and constants

Herbrand instance of Skolem formula $\forall x_1, \ldots, x_n\ \varphi[x_1, \ldots, x_n]$ is $\varphi[t_1, \ldots, t_n]$ where t_i is term over signature of φ

Definition

Herbrand instance of Skolem formula $\forall x_1, \ldots, x_n\ \varphi[x_1, \ldots, x_n]$ is $\varphi[t_1, \ldots, t_n]$ where t_i is term over signature of φ

Remark

Herbrand instances are ground formulas, i.e., without (quantified) variables

Jacques Herbrand

Definition

Herbrand instance of Skolem formula $\forall x_1, \ldots, x_n\ \varphi[x_1, \ldots, x_n]$ is $\varphi[t_1, \ldots, t_n]$
where t_i is term over signature of φ

Remark

Herbrand instances are ground formulas, i.e., without (quantified) variables

Theorem (Herbrand)

Skolem formula φ is unsatisfiable $\Longleftrightarrow$
there exists finite unsatisfiable set of Herbrand instances of φ

Jacques Herbrand

Definition

Herbrand instance of Skolem formula $\forall x_1, \ldots, x_n\ \varphi[x_1, \ldots, x_n]$ is $\varphi[t_1, \ldots, t_n]$
where t_i is term over signature of φ

Remark

Herbrand instances are ground formulas, i.e., without (quantified) variables

Theorem (Herbrand)

Skolem formula φ is unsatisfiable $\Longleftrightarrow$
there exists finite unsatisfiable set of Herbrand instances of φ

Jacques Herbrand

Caveats

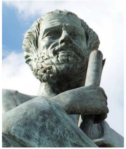
- at least one constant required per sort
- holds for pure first order logic, not necessarily in presence of theories

Example: Is this syllogism correct?

All humans are mortal.

All Greeks are humans.

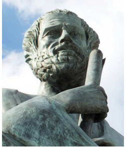
So all Greeks are mortal.



Aristotle

Example: Is this syllogism correct?

All humans are mortal.	$\forall x.\ H(x) \longrightarrow M(x)$
All Greeks are humans.	$\forall x.\ G(x) \longrightarrow H(x)$
So all Greeks are mortal.	$\forall x.\ G(x) \longrightarrow M(x)$

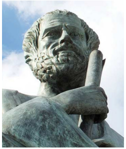


Aristotle

- translate to first-order logic

Example: Is this syllogism correct?

All humans are mortal.	$\forall x.\ H(x) \longrightarrow M(x)$
All Greeks are humans.	$\forall x.\ G(x) \longrightarrow H(x)$
So all Greeks are mortal.	$\forall x.\ G(x) \longrightarrow M(x)$



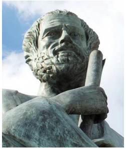
Aristotle

- translate to first-order logic
- check validity of

$$((\forall x.\ H(x) \longrightarrow M(x)) \wedge (\forall x.\ G(x) \longrightarrow H(x))) \longrightarrow (\forall x.\ G(x) \longrightarrow M(x))$$

Example: Is this syllogism correct?

All humans are mortal.	$\forall x.\ H(x) \longrightarrow M(x)$
All Greeks are humans.	$\forall x.\ G(x) \longrightarrow H(x)$
So all Greeks are mortal.	$\forall x.\ G(x) \longrightarrow M(x)$



Aristotle

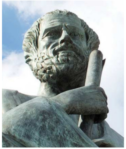
- translate to first-order logic
- check validity of

cannot be answered by SMT solver

$$((\forall x.\ H(x) \longrightarrow M(x)) \wedge (\forall x.\ G(x) \longrightarrow H(x))) \longrightarrow (\forall x.\ G(x) \longrightarrow M(x))$$

Example: Is this syllogism correct?

$$
\begin{array}{l}
\text{All humans are mortal.} \\
\text{All Greeks are humans.} \\
\hline
\text{So all Greeks are mortal.}
\end{array}
\qquad
\begin{array}{c}
\forall x.\ H(x) \longrightarrow M(x) \\
\forall x.\ G(x) \longrightarrow H(x) \\
\hline
\forall x.\ G(x) \longrightarrow M(x)
\end{array}
$$



Aristotle

- translate to first-order logic
- check validity of

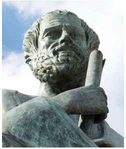
$$((\forall x.\ H(x) \longrightarrow M(x)) \wedge (\forall x.\ G(x) \longrightarrow H(x))) \longrightarrow (\forall x.\ G(x) \longrightarrow M(x))$$

- check unsatisfiability of

$$\forall x.\ H(x) \longrightarrow M(x), \quad \forall x.\ G(x) \longrightarrow H(x), \quad \exists x.\ G(x) \wedge \neg M(x)$$

Example: Is this syllogism correct?

All humans are mortal.	$\forall x.\ H(x) \longrightarrow M(x)$
All Greeks are humans.	$\forall x.\ G(x) \longrightarrow H(x)$
So all Greeks are mortal.	$\forall x.\ G(x) \longrightarrow M(x)$



Aristotle

- translate to first-order logic
- check validity of

$$((\forall x.\ H(x) \longrightarrow M(x)) \wedge (\forall x.\ G(x) \longrightarrow H(x))) \longrightarrow (\forall x.\ G(x) \longrightarrow M(x))$$

- check unsatisfiability of

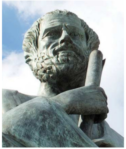
$$\forall x.\ H(x) \longrightarrow M(x), \quad \forall x.\ G(x) \longrightarrow H(x), \quad \exists x.\ G(x) \wedge \neg M(x)$$

- skolemize

$$\forall x.\ H(x) \longrightarrow M(x), \quad \forall x.\ G(x) \longrightarrow H(x), \quad G(a) \wedge \neg M(a)$$

Example: Is this syllogism correct?

$$\frac{\begin{array}{l}\text{All humans are mortal.}\\ \text{All Greeks are humans.}\end{array}}{\text{So all Greeks are mortal.}} \qquad \frac{\begin{array}{l}\forall x.\ H(x) \longrightarrow M(x)\\ \forall x.\ G(x) \longrightarrow H(x)\end{array}}{\forall x.\ G(x) \longrightarrow M(x)}$$



Aristotle

- translate to first-order logic
- check validity of

$$((\forall x.\ H(x) \longrightarrow M(x)) \wedge (\forall x.\ G(x) \longrightarrow H(x))) \longrightarrow (\forall x.\ G(x) \longrightarrow M(x))$$

- check unsatisfiability of

$$\forall x.\ H(x) \longrightarrow M(x), \quad \forall x.\ G(x) \longrightarrow H(x), \quad \exists x.\ G(x) \wedge \neg M(x)$$

- skolemize

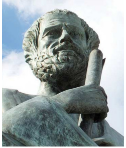
$$\forall x.\ H(x) \longrightarrow M(x), \quad \forall x.\ G(x) \longrightarrow H(x), \quad G(a) \wedge \neg M(a)$$

- already unsatisfiable when replacing quantified formulas by Herbrand instances

$$H(a) \longrightarrow M(a), \quad G(a) \longrightarrow H(a), \quad G(a) \wedge \neg M(a)$$

Example: Is this syllogism correct?

All humans are mortal.	$\forall x.\ H(x) \longrightarrow M(x)$
All Greeks are humans.	$\forall x.\ G(x) \longrightarrow H(x)$
So all Greeks are mortal.	$\forall x.\ G(x) \longrightarrow M(x)$



Aristotle

- translate to first-order logic
- check validity of

$$((\forall x.\ H(x) \longrightarrow M(x)) \wedge (\forall x.\ G(x) \longrightarrow H(x))) \longrightarrow (\forall x.\ G(x) \longrightarrow M(x))$$

- check unsatisfiability of

$$\forall x.\ H(x) \longrightarrow M(x), \quad \forall x.\ G(x) \longrightarrow H(x), \quad \exists x.\ G(x) \wedge \neg M(x)$$

- skolemize

$$\forall x.\ H(x) \longrightarrow M(x),$$

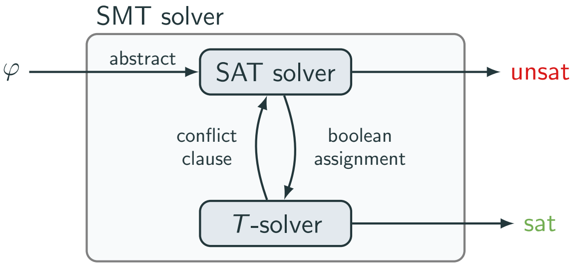
when adding right Herbrand instances
unsatisfiability can be detected by SMT solver

- already unsatisfiable when replacing quantified formulas by Herbrand instances

$$H(a) \longrightarrow M(a), \quad G(a) \longrightarrow H(a), \quad G(a) \wedge \neg M(a)$$

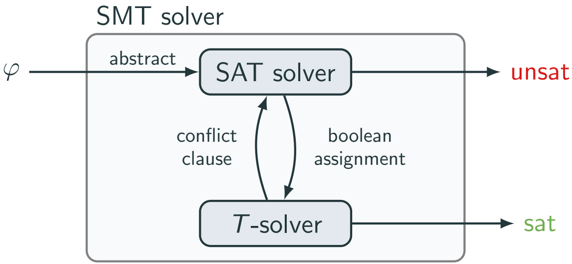
Outline

- Summary of Last Week
- Instantiation Techniques
  - E-Matching
  - Enumerative Instantiation
- More on SAT and SMT
- Test
- Evaluation



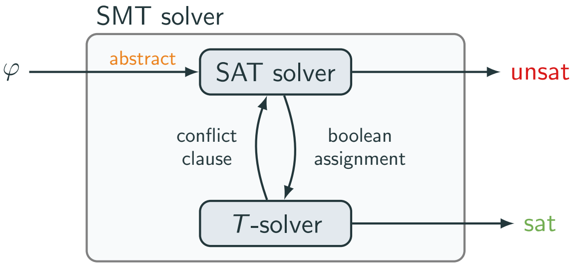
Example

- $a = b \land \mathrm{g}(a) = a \land (\mathrm{f}(a) \neq \mathrm{f}(b) \lor b \neq \mathrm{g}(\mathrm{g}(a)))$



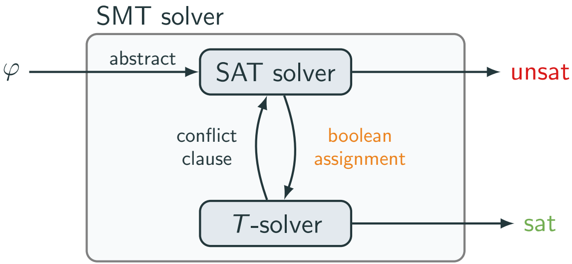
Example

- $a = b \land \mathrm{g}(a) = a \land (\mathrm{f}(a) \neq \mathrm{f}(b) \lor b \neq \mathrm{g}(\mathrm{g}(a)))$



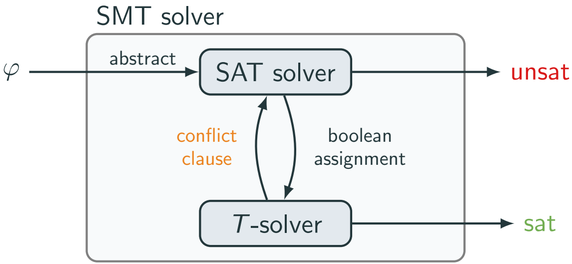
Example

- $a = b \wedge \mathrm{g}(a) = a \wedge (\mathrm{f}(a) \neq \mathrm{f}(b) \vee b \neq \mathrm{g}(\mathrm{g}(a)))$
- abstract to $p_{a=b} \wedge p_{\mathrm{g}(a)=a} \wedge \left(p_{\mathrm{f}(a)\neq\mathrm{f}(b)} \vee p_{b\neq\mathrm{g}(\mathrm{g}(a))}\right)$



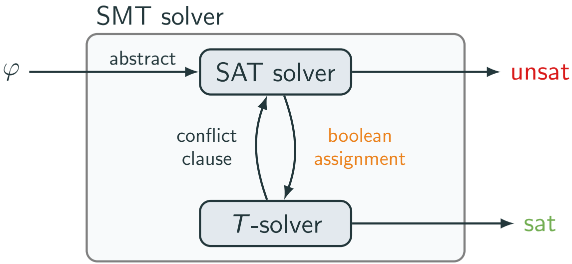
Example

- $a = b \wedge \mathrm{g}(a) = a \wedge (\mathrm{f}(a) \neq \mathrm{f}(b) \vee b \neq \mathrm{g}(\mathrm{g}(a)))$
- abstract to $p_{a=b} \wedge p_{\mathrm{g}(a)=a} \wedge \left(p_{\mathrm{f}(a)\neq\mathrm{f}(b)} \vee p_{b\neq\mathrm{g}(\mathrm{g}(a))}\right)$
- SAT solver: $p_{a=b},\ p_{\mathrm{g}(a)=a},\ p_{\mathrm{f}(a)\neq\mathrm{f}(b)}$



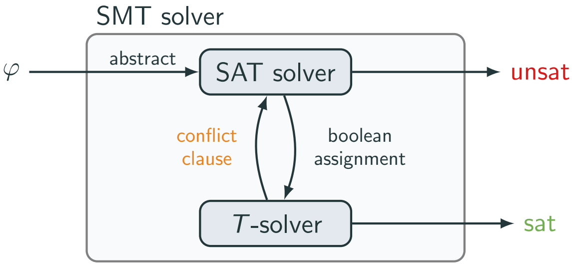
Example

- $a = b \land \mathrm{g}(a) = a \land (\mathrm{f}(a) \neq \mathrm{f}(b) \lor b \neq \mathrm{g}(\mathrm{g}(a)))$
- abstract to $p_{a=b} \land p_{\mathrm{g}(a)=a} \land \left(p_{\mathrm{f}(a)\neq\mathrm{f}(b)} \lor p_{b\neq\mathrm{g}(\mathrm{g}(a))}\right)$
- SAT solver: $p_{a=b},\ p_{\mathrm{g}(a)=a},\ p_{\mathrm{f}(a)\neq\mathrm{f}(b)}$ T-solver: $\neg p_{a=b} \lor \neg p_{\mathrm{f}(a)\neq\mathrm{f}(b)}$



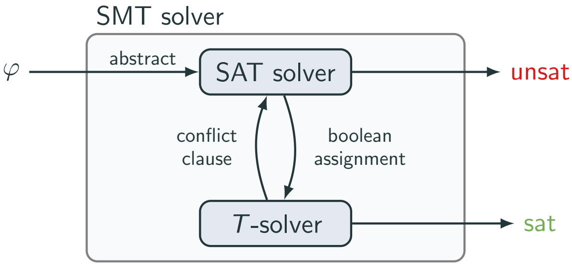
Example

- $a = b \wedge g(a) = a \wedge (f(a) \neq f(b) \vee b \neq g(g(a)))$
- abstract to $p_{a=b} \wedge p_{g(a)=a} \wedge \left(p_{f(a)\neq f(b)} \vee p_{b\neq g(g(a))}\right)$
- SAT solver: $p_{a=b}$, $p_{g(a)=a}$, $p_{f(a)\neq f(b)}$ $\quad$ T-solver: $\neg p_{a=b} \vee \neg p_{f(a)\neq f(b)}$
- SAT solver: $p_{a=b}$, $p_{g(a)=a}$, $p_{b\neq g(g(a))}$



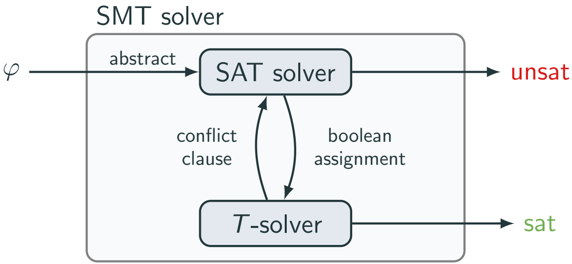
Example

- $a = b \land \mathrm{g}(a) = a \land (\mathrm{f}(a) \neq \mathrm{f}(b) \lor b \neq \mathrm{g}(\mathrm{g}(a)))$
- abstract to $p_{a=b} \land p_{\mathrm{g}(a)=a} \land \left(p_{\mathrm{f}(a)\neq\mathrm{f}(b)} \lor p_{b\neq\mathrm{g}(\mathrm{g}(a))}\right)$
- SAT solver: $p_{a=b},\ p_{\mathrm{g}(a)=a},\ p_{\mathrm{f}(a)\neq\mathrm{f}(b)}$ T-solver: $\neg p_{a=b} \lor \neg p_{\mathrm{f}(a)\neq\mathrm{f}(b)}$
- SAT solver: $p_{a=b},\ p_{\mathrm{g}(a)=a},\ p_{b\neq\mathrm{g}(\mathrm{g}(a))}$ T-solver: $\neg p_{a=b} \lor \neg p_{\mathrm{g}(a)=a} \lor \neg p_{b\neq\mathrm{g}(\mathrm{g}(a))}$



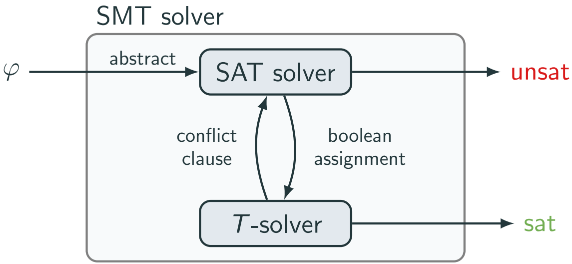
Example

- $a = b \wedge \mathrm{g}(a) = a \wedge (\mathrm{f}(a) \neq \mathrm{f}(b) \vee b \neq \mathrm{g}(\mathrm{g}(a)))$
- abstract to $p_{a=b} \wedge p_{\mathrm{g}(a)=a} \wedge \left(p_{\mathrm{f}(a)\neq\mathrm{f}(b)} \vee p_{b\neq\mathrm{g}(\mathrm{g}(a))}\right)$
- SAT solver: $p_{a=b},\ p_{\mathrm{g}(a)=a},\ p_{\mathrm{f}(a)\neq\mathrm{f}(b)}$ T-solver: $\neg p_{a=b} \vee \neg p_{\mathrm{f}(a)\neq\mathrm{f}(b)}$
- SAT solver: $p_{a=b},\ p_{\mathrm{g}(a)=a},\ p_{b\neq\mathrm{g}(\mathrm{g}(a))}$ T-solver: $\neg p_{a=b} \vee \neg p_{\mathrm{g}(a)=a} \vee \neg p_{b\neq\mathrm{g}(\mathrm{g}(a))}$
- SAT solver: unsat



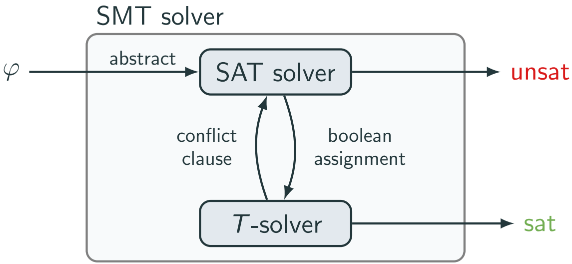
Example

- $a = b \wedge \mathrm{g}(a) \neq b \wedge (\mathrm{f}(a) \neq \mathrm{f}(b) \vee \forall x.\, x = \mathrm{g}(x)))$



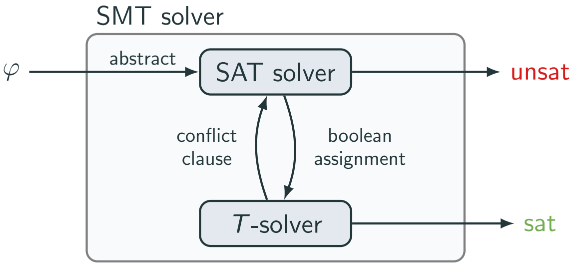
Example

- $a = b \wedge \mathrm{g}(a) \neq b \wedge (\mathrm{f}(a) \neq \mathrm{f}(b) \vee \forall x.\, x = \mathrm{g}(x)))$
- abstract to $p_{a=b} \wedge p_{\mathrm{g}(a)\neq b} \wedge \left(p_{\mathrm{f}(a)\neq\mathrm{f}(b)} \vee p_{\forall x.x=\mathrm{g}(x)}\right)$



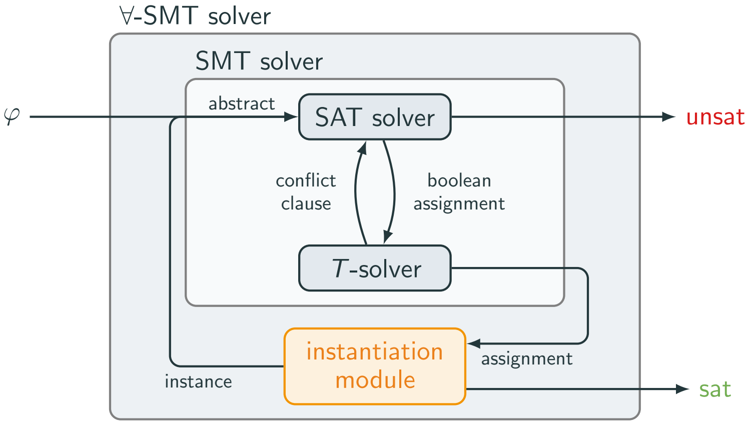
Example

- $a = b \wedge \mathrm{g}(a) \neq b \wedge (\mathrm{f}(a) \neq \mathrm{f}(b) \vee \forall x.\, x = \mathrm{g}(x)))$
- abstract to $p_{a=b} \wedge p_{\mathrm{g}(a)\neq b} \wedge \left(p_{\mathrm{f}(a)\neq \mathrm{f}(b)} \vee p_{\forall x. x=\mathrm{g}(x)}\right)$
- SAT solver: $p_{a=b},\ p_{\mathrm{g}(a)\neq b},\ p_{\forall x. x=\mathrm{g}(x)}$



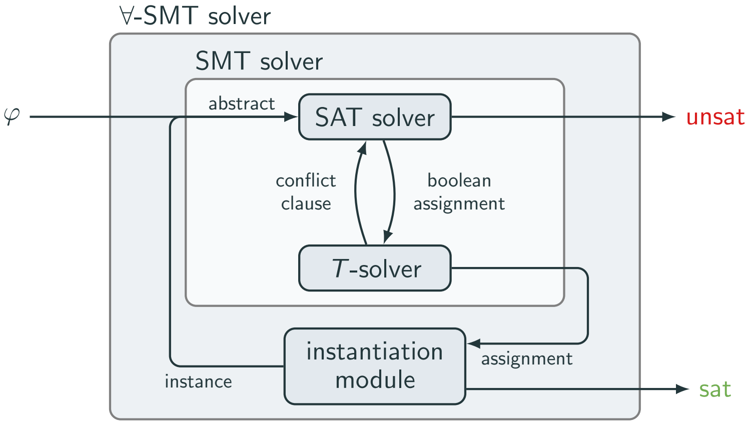
Example

- $a = b \wedge \mathrm{g}(a) \neq b \wedge (\mathrm{f}(a) \neq \mathrm{f}(b) \vee \forall x.\, x = \mathrm{g}(x)))$
- abstract to $p_{a=b} \wedge p_{\mathrm{g}(a)\neq b} \wedge \left(p_{\mathrm{f}(a)\neq \mathrm{f}(b)} \vee p_{\forall x.x=\mathrm{g}(x)}\right)$
- SAT solver: $p_{a=b},\ p_{\mathrm{g}(a)\neq b},\ p_{\forall x.x=\mathrm{g}(x)}$ T-solver: ok, but what is $\forall$?



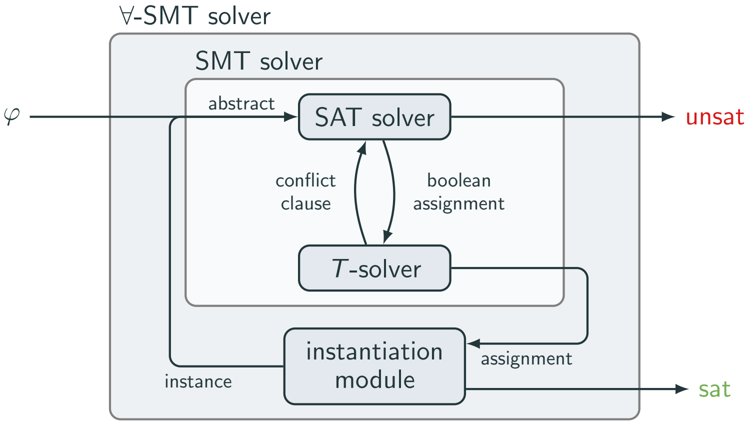
Example

- $a = b \wedge \mathrm{g}(a) \neq b \wedge (\mathrm{f}(a) \neq \mathrm{f}(b) \vee \forall x.\, x = \mathrm{g}(x)))$
- abstract to $p_{a=b} \wedge p_{\mathrm{g}(a)\neq b} \wedge \left(p_{\mathrm{f}(a)\neq \mathrm{f}(b)} \vee p_{\forall x. x=\mathrm{g}(x)}\right)$
- SAT solver: $p_{a=b}$, $p_{\mathrm{g}(a)\neq b}$, $p_{\forall x. x=\mathrm{g}(x)}$ T-solver: ok, but what is $\forall$?



Example

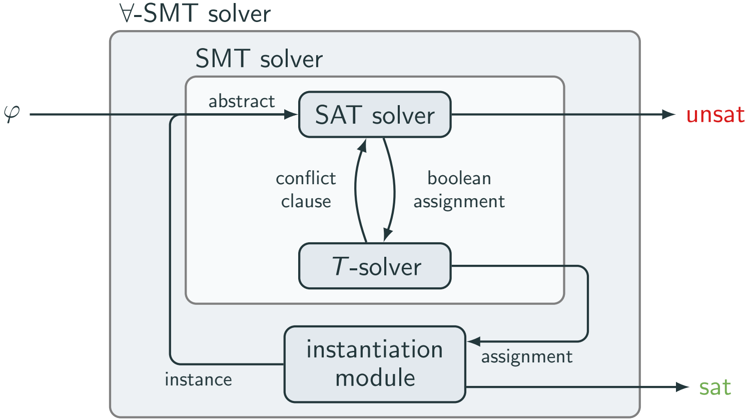
- $a = b \wedge \mathrm{g}(a) \neq b \wedge (\mathrm{f}(a) \neq \mathrm{f}(b) \vee \forall x.\, x = \mathrm{g}(x)))$
- abstract to $p_{a=b} \wedge p_{\mathrm{g}(a)\neq b} \wedge \left(p_{\mathrm{f}(a)\neq \mathrm{f}(b)} \vee p_{\forall x. x=\mathrm{g}(x)}\right)$
- SAT solver: $p_{a=b}$, $p_{\mathrm{g}(a)\neq b}$, $p_{\forall x. x=\mathrm{g}(x)}$ T-solver: ok, but what is $\forall$?
- instantiation module: find clause to do with $\forall x.\, x = \mathrm{g}(x)$ to exclude model!



Example

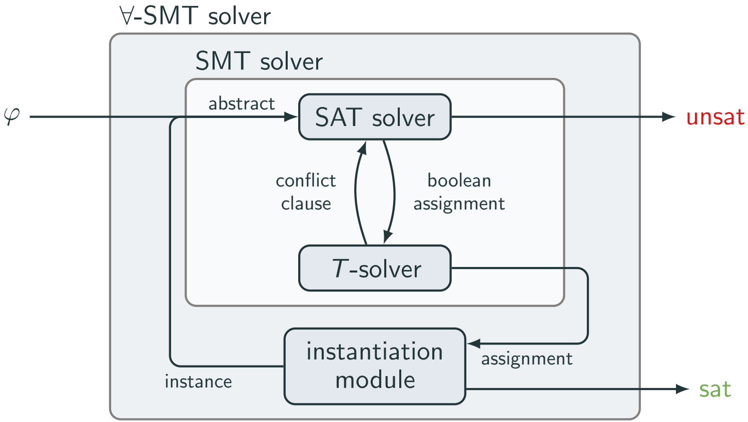
- $a = b \wedge g(a) \neq b \wedge (f(a) \neq f(b) \vee \forall x.\, x = g(x))$
- abstract to $p_{a=b} \wedge p_{g(a)\neq b} \wedge (p_{f(a)\neq f(b)} \vee$
- SAT solver: $p_{a=b}$, $p_{g(a)\neq b}$, $p_{\forall x. x=g(x)}$ T-solver: ok, but what is $\forall$?
- instantiation module: find clause to do with $\forall x.\, x = g(x)$ to exclude model!

want $a = g(a)$ whenever $p_{\forall x. x=g(x)}$ true



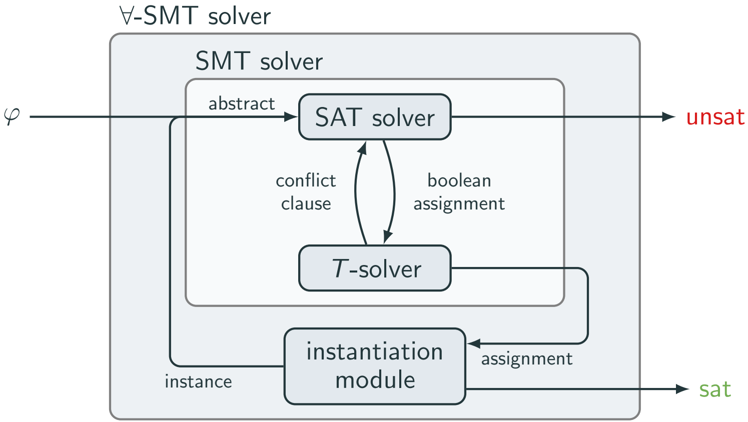
Example

- $a = b \wedge \mathrm{g}(a) \neq b \wedge (\mathrm{f}(a) \neq \mathrm{f}(b) \vee \forall x.\, x = \mathrm{g}(x)))$
- abstract to $p_{a=b} \wedge p_{\mathrm{g}(a)\neq b} \wedge \left(p_{\mathrm{f}(a)\neq \mathrm{f}(b)} \vee p_{\forall x.x=\mathrm{g}(x)}\right)$
- SAT solver: $p_{a=b}$, $p_{\mathrm{g}(a)\neq b}$, $p_{\forall x.x=\mathrm{g}(x)}$ $\quad$ T-solver: ok, but what is $\forall$?
- instantiation module: add $(\forall x.x = \mathrm{g}(x)) \longrightarrow a = \mathrm{g}(a)$



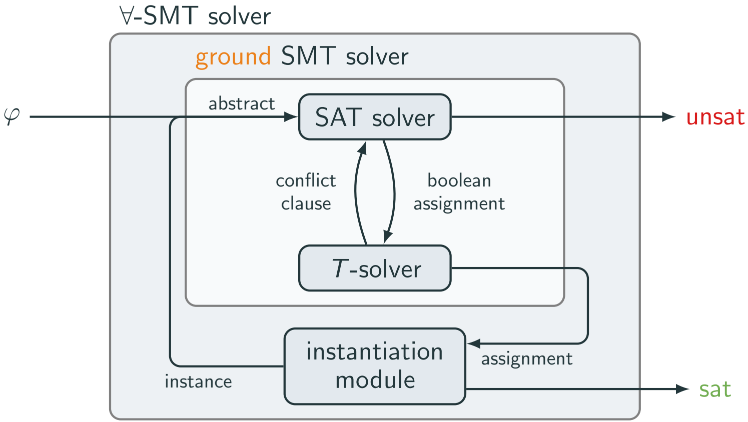
Example

- $a = b \wedge \mathrm{g}(a) \neq b \wedge (\mathrm{f}(a) \neq \mathrm{f}(b) \vee \forall x.\, x = \mathrm{g}(x)))$
- abstract to $p_{a=b} \wedge p_{\mathrm{g}(a)\neq b} \wedge \left(p_{\mathrm{f}(a)\neq \mathrm{f}(b)} \vee p_{\forall x.x=\mathrm{g}(x)}\right)$
- SAT solver: $p_{a=b},\ p_{\mathrm{g}(a)\neq b},\ p_{\forall x.x=\mathrm{g}(x)}$ T-solver: ok, but what is $\forall$?
- instantiation module: add $(\forall x.x = \mathrm{g}(x)) \longrightarrow a = \mathrm{g}(a)$
- SAT solver: $p_{a=b},\ p_{\mathrm{g}(a)\neq b},\ p_{\forall x.x=\mathrm{g}(x)},\ p_{a=\mathrm{g}(a)}$



Example

- $a = b \wedge \mathrm{g}(a) \neq b \wedge (\mathrm{f}(a) \neq \mathrm{f}(b) \vee \forall x.\, x = \mathrm{g}(x))$
- abstract to $p_{a=b} \wedge p_{\mathrm{g}(a)\neq b} \wedge \left(p_{\mathrm{f}(a)\neq \mathrm{f}(b)} \vee p_{\forall x. x=\mathrm{g}(x)}\right)$
- SAT solver: $p_{a=b}$, $p_{\mathrm{g}(a)\neq b}$, $p_{\forall x. x=\mathrm{g}(x)}$ T-solver: ok, but what is $\forall$?
- instantiation module: add $(\forall x. x = \mathrm{g}(x)) \longrightarrow a = \mathrm{g}(a)$
- SAT solver: $p_{a=b}$, $p_{\mathrm{g}(a)\neq b}$, $p_{\forall x. x=\mathrm{g}(x)}$, $p_{a=\mathrm{g}(a)}$ T: $\neg p_{a=b} \vee p_{\mathrm{g}(a)=b} \vee p_{a\neq \mathrm{g}(a)}$



Example

- $a = b \wedge \mathrm{g}(a) \neq b \wedge (\mathrm{f}(a) \neq \mathrm{f}(b) \vee \forall x.\, x = \mathrm{g}(x)))$
- abstract to $p_{a=b} \wedge p_{\mathrm{g}(a)\neq b} \wedge \left(p_{\mathrm{f}(a)\neq \mathrm{f}(b)} \vee p_{\forall x.x=\mathrm{g}(x)}\right)$
- SAT solver: $p_{a=b}$, $p_{\mathrm{g}(a)\neq b}$, $p_{\forall x.x=\mathrm{g}(x)}$ T-solver: ok, but what is $\forall$?
- instantiation module: add $(\forall x.x = \mathrm{g}(x)) \longrightarrow a = \mathrm{g}(a)$
- SAT solver: $p_{a=b}$, $p_{\mathrm{g}(a)\neq b}$, $p_{\forall x.x=\mathrm{g}(x)}$, $p_{a=\mathrm{g}(a)}$ T: $\neg p_{a=b} \vee p_{\mathrm{g}(a)=b} \vee p_{a\neq \mathrm{g}(a)}$

Instantiation

Definition (Instance)

$$(\forall \overline{x}\ \varphi(\overline{x})) \longrightarrow \varphi\sigma$$

is instance where $\overline{x}\sigma$ does not contain variables $\overline{x}$

Instantiation

Definition (Instance)

$$(\forall \overline{x}\ \varphi(\overline{x})) \longrightarrow \varphi\sigma$$

is instance where $\overline{x}\sigma$ does not contain variables $\overline{x}$

Example

$\forall x.\ H(x) \longrightarrow M(x)$ has instance $(\forall x.\ H(x) \longrightarrow M(x)) \longrightarrow (H(a) \longrightarrow M(a))$

Instantiation

Definition (Instance)

$$(\forall \overline{x}\ \varphi(\overline{x})) \longrightarrow \varphi\sigma$$

is instance where $\overline{x}\sigma$ does not contain variables $\overline{x}$

Example

$\forall x.\ H(x) \longrightarrow M(x)$ has instance $(\forall x.\ H(x) \longrightarrow M(x)) \longrightarrow (H(a) \longrightarrow M(a))$

Remarks

- as first-order logic formula, every instance is tautology

Instantiation

Definition (Instance)

$$(\forall \overline{x}\ \varphi(\overline{x})) \longrightarrow \varphi\sigma$$

is instance where $\overline{x}\sigma$ does not contain variables $\overline{x}$

Example

$\forall x.\ H(x) \longrightarrow M(x)$ has instance $(\forall x.\ H(x) \longrightarrow M(x)) \longrightarrow (H(a) \longrightarrow M(a))$

Remarks

- as first-order logic formula, every instance is tautology
- in SAT solver, $\forall \overline{x}\ \varphi(\overline{x})$ gets abstracted to propositional variable $p_{\forall \overline{x}\ \varphi(\overline{x})}$, which has meaning only for instantiation module

Instantiation

Definition (Instance)

$$(\forall \overline{x}\ \varphi(\overline{x})) \longrightarrow \varphi\sigma$$

is instance where $\overline{x}\sigma$ does not contain variables $\overline{x}$

Example

$\forall x.\ H(x) \longrightarrow M(x)$ has instance $(\forall x.\ H(x) \longrightarrow M(x)) \longrightarrow (H(a) \longrightarrow M(a))$

Remarks

- as first-order logic formula, every instance is tautology
- in SAT solver, $\forall \overline{x}\ \varphi(\overline{x})$ gets abstracted to propositional variable $p_{\forall \overline{x}\ \varphi(\overline{x})}$, which has meaning only for instantiation module
- $\varphi\sigma$ gets abstracted to propositional formula: involved variables have meaning for theory solver

Instantiation

Definition (Instance)

$$(\forall \overline{x}\ \varphi(\overline{x})) \longrightarrow \varphi\sigma$$

is instance where $\overline{x}\sigma$ does not contain variables $\overline{x}$

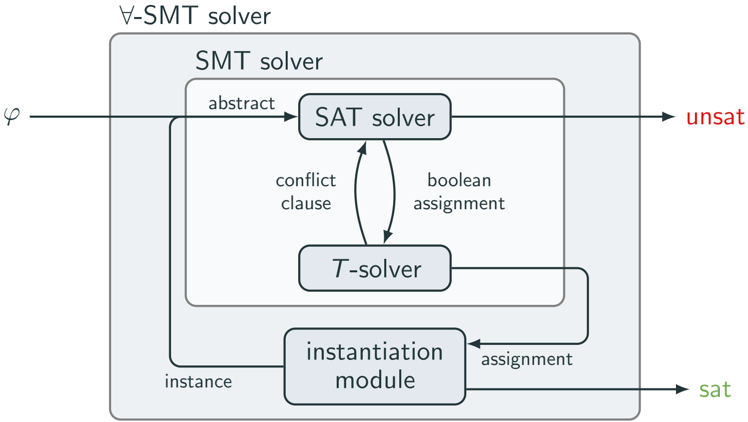
Example

$\forall x.\ H(x) \longrightarrow M(x)$ has instance $(\forall x.\ H(x) \longrightarrow M(x)) \longrightarrow (H(a) \longrightarrow M(a))$

Remarks

- as first-order logic formula, every instance is tautology
- in SAT solver, $\forall \overline{x}\ \varphi(\overline{x})$ gets abstracted to propositional variable $p_{\forall \overline{x}\ \varphi(\overline{x})}$, which has meaning only for instantiation module
- $\varphi\sigma$ gets abstracted to propositional formula: involved variables have meaning for theory solver
- idea: $\varphi\sigma$ gets "activated" if propositional variable $p_{\forall \overline{x}\ \varphi(\overline{x})}$ is assigned true

Instantiation Framework



- split φ into
  - literals φ_Q with quantifiers
  - literals φ_E without quantifiers
- instantiation module generates instances of φ_Q to extend φ_E

E-Matching

Example

$$\varphi_E : \neg P(a),\ \neg P(b),\ \neg R(b)$$
$$\varphi_Q : \forall x.\ P(x) \vee R(x)$$

E-Matching

Example

$$\varphi_E : \neg P(a),\ \neg P(b),\ \neg R(b)$$

$$\varphi_Q : \forall x.\ P(x) \vee R(x)$$

- assume literal $P(x)$ is instantiation pattern

trigger

E-Matching

Example

$$\varphi_E : \neg P(a),\ \neg P(b),\ \neg R(b)$$

$$\varphi_Q : \forall x.\ P(x) \vee R(x)$$

trigger

- assume literal $P(x)$ is instantiation pattern
- find substitutions σ such that $P(x)\sigma$ occurs in φ_E matching

E-Matching

Example

$$\varphi_E : \neg P(a),\ \neg P(b),\ \neg R(b)$$

$$\varphi_Q : \forall x.\ P(x) \lor R(x)$$

trigger

- assume literal $P(x)$ is instantiation pattern
- find substitutions σ such that $P(x)\sigma$ occurs in φ_E matching
- obtain $\{x \mapsto a\}$, $\{x \mapsto b\}$

E-Matching

Example

$$\varphi_E : \neg P(a),\ \neg P(b),\ \neg R(b)$$

$$\varphi_Q : \forall x.\ P(x) \vee R(x)$$

trigger

- assume literal $P(x)$ is instantiation pattern
- find substitutions σ such that $P(x)\sigma$ occurs in φ_E matching
- obtain $\{x \mapsto a\}$, $\{x \mapsto b\}$
- add $P(a) \vee R(a)$ and $P(b) \vee R(b)$ to φ_E

E-Matching

Example

$$\varphi_E : \neg P(a),\ \neg P(b),\ \neg R(b)$$

$$\varphi_Q : \forall x.\ P(x) \vee R(x)$$

trigger

- assume literal $P(x)$ is instantiation pattern
- find substitutions σ such that $P(x)\sigma$ occurs in φ_E matching
- obtain $\{x \mapsto a\}$, $\{x \mapsto b\}$
- add $P(a) \vee R(a)$ and $P(b) \vee R(b)$ to φ_E

Instantiation via E-matching

for each $\forall \overline{x}.\psi$

- select set of instantiation patterns $\{t_1, \ldots, t_n\}$
- for each t_i let S_i be set of substitutions σ such that $t_i\sigma$ occurs in φ_E
- add $\{\psi\sigma \mid \sigma \in S_i\}$ to φ_E

Example

$$\forall x \forall y.\, \mathrm{sibling}(x, y) \longleftrightarrow \mathrm{mother}(x) = \mathrm{mother}(y) \land \mathrm{father}(x) = \mathrm{father}(y)$$

$$\mathrm{sibling}(\mathrm{adam}, \mathrm{bea})$$

$$\mathrm{sibling}(\mathrm{bea}, \mathrm{chris})$$

$$\neg \mathrm{sibling}(\mathrm{adam}, \mathrm{chris})$$

Example

$$\forall x \forall y.\, \text{sibling}(x, y) \longleftrightarrow \text{mother}(x) = \text{mother}(y) \land \text{father}(x) = \text{father}(y)$$

$$\text{sibling}(\text{adam}, \text{bea})$$

$$\text{sibling}(\text{bea}, \text{chris})$$

$$\neg\text{sibling}(\text{adam}, \text{chris})$$

- unsatisfiable 🪄

Example

$$\forall x \forall y.\, \mathsf{sibling}(x, y) \longleftrightarrow \mathsf{mother}(x) = \mathsf{mother}(y) \land \mathsf{father}(x) = \mathsf{father}(y)$$

$$\mathsf{sibling}(\mathsf{adam}, \mathsf{bea})$$

$$\mathsf{sibling}(\mathsf{bea}, \mathsf{chris})$$

$$\neg \mathsf{sibling}(\mathsf{adam}, \mathsf{chris})$$

- unsatisfiable 🪄
- suitable instantiation patterns?
  $\mathsf{sibling}(x, y)$ sufficient

Example

$$\forall x \forall y.\, \mathsf{sibling}(x, y) \longleftrightarrow \mathsf{mother}(x) = \mathsf{mother}(y) \land \mathsf{father}(x) = \mathsf{father}(y)$$

$$\mathsf{sibling}(\mathsf{adam}, \mathsf{bea})$$

$$\mathsf{sibling}(\mathsf{bea}, \mathsf{chris})$$

$$\neg\mathsf{sibling}(\mathsf{adam}, \mathsf{chris})$$

- unsatisfiable
- suitable instantiation patterns?
  $\mathsf{sibling}(x, y)$ sufficient

Remarks

- works as decision procedure for some theories (e.g., lists and arrays)
  but can easily omit necessary instances in other cases

Example

$$\forall x \forall y.\, \mathsf{sibling}(x, y) \longleftrightarrow \mathsf{mother}(x) = \mathsf{mother}(y) \land \mathsf{father}(x) = \mathsf{father}(y)$$

$$\mathsf{sibling}(\mathsf{adam}, \mathsf{bea})$$

$$\mathsf{sibling}(\mathsf{bea}, \mathsf{chris})$$

$$\neg\mathsf{sibling}(\mathsf{adam}, \mathsf{chris})$$

- unsatisfiable
- suitable instantiation patterns?
  $\mathsf{sibling}(x, y)$ sufficient

Remarks

- works as decision procedure for some theories (e.g., lists and arrays)
  but can easily omit necessary instances in other cases
- mostly efficient

Example

$$\forall x \forall y.\, \mathsf{sibling}(x, y) \longleftrightarrow \mathsf{mother}(x) = \mathsf{mother}(y) \wedge \mathsf{father}(x) = \mathsf{father}(y)$$

$$\mathsf{sibling}(\mathsf{adam}, \mathsf{bea})$$

$$\mathsf{sibling}(\mathsf{bea}, \mathsf{chris})$$

$$\neg \mathsf{sibling}(\mathsf{adam}, \mathsf{chris})$$

- unsatisfiable 🪄
- suitable instantiation patterns?
  $\mathsf{sibling}(x, y)$ sufficient

Remarks

- works as decision procedure for some theories (e.g., lists and arrays)
  but can easily omit necessary instances in other cases
- mostly efficient
- requires instantiation patterns (manually or heuristically determined)

Example

$$\forall x \forall y.\, \mathsf{sibling}(x, y) \longleftrightarrow \mathsf{mother}(x) = \mathsf{mother}(y) \land \mathsf{father}(x) = \mathsf{father}(y)$$

$$\mathsf{sibling}(\mathsf{adam}, \mathsf{bea})$$

$$\mathsf{sibling}(\mathsf{bea}, \mathsf{chris})$$

$$\neg \mathsf{sibling}(\mathsf{adam}, \mathsf{chris})$$

- unsatisfiable 🪄
- suitable instantiation patterns?
  $\mathsf{sibling}(x, y)$ sufficient

Remarks

- works as decision procedure for some theories (e.g., lists and arrays)
  but can easily omit necessary instances in other cases
- mostly efficient
- requires instantiation patterns (manually or heuristically determined)
- instantiation patterns can be specified in SMT-LIB 🪄

Outline

- Summary of Last Week
- Instantiation Techniques
  - E-Matching
  - Enumerative Instantiation
- More on SAT and SMT
- Test
- Evaluation

Enumerative Instantiation

Why not use Herbrand's theorem directly?

Theorem (Herbrand)

Skolem formula φ *is unsatisfiable* $\iff$
there exists finite unsatisfiable set of Herbrand instances of φ

Enumerative Instantiation

Why not use Herbrand's theorem directly?

Theorem (Herbrand)

Skolem formula φ is unsatisfiable $\Longleftrightarrow$
there exists finite unsatisfiable set of Herbrand instances of φ

Early days of theorem proving

- first theorem provers enumerated Herbrand instances, looked for refutation
- infeasible in practice
- approach was forgotten

Enumerative Instantiation

Why not use Herbrand's theorem directly?

Theorem (Herbrand)

Skolem formula φ is unsatisfiable $\Longleftrightarrow$
there exists finite unsatisfiable set of Herbrand instances of φ

Early days of theorem proving

- first theorem provers enumerated Herbrand instances, looked for refutation
- infeasible in practice
- approach was forgotten

Enumerative instantiation

- instantiation module based on stronger version of Herbrand's theorem
- efficient implementation techniques

Theorem (Stronger Herbrand)

$\varphi_E \wedge \varphi_Q$ is unsatisfiable if and only if there exist infinite series

- $\mathbf{E}_i$ *of finite literals sets*
- $\mathbf{Q}_i$ *of finite sets of φ_Q instances*

such that

Theorem (Stronger Herbrand)

$\varphi_E \wedge \varphi_Q$ *is unsatisfiable if and only if there exist infinite series*

- $\mathbf{E}_i$ *of finite literals sets*
- $\mathbf{Q}_i$ *of finite sets of* φ_Q *instances*

such that

- $\mathbf{Q}_i \subseteq \{\psi\sigma \mid \forall\overline{x}.\, \psi$ *occurs in* φ_Q

Theorem (Stronger Herbrand)

$\varphi_E \wedge \varphi_Q$ *is unsatisfiable if and only if there exist infinite series*

- $\mathbf{E}_i$ *of finite literals sets*
- $\mathbf{Q}_i$ *of finite sets of* φ_Q *instances*

such that

- $\mathbf{Q}_i \subseteq \{\psi\sigma \mid \forall\overline{x}.\, \psi$ *occurs in* φ_Q *and* $\mathrm{dom}(\sigma) = \overline{x}$

Theorem (Stronger Herbrand)

$\varphi_E \wedge \varphi_Q$ *is unsatisfiable if and only if there exist infinite series*

- $\mathbf{E}_i$ *of finite literals sets*
- $\mathbf{Q}_i$ *of finite sets of* φ_Q *instances*

such that

- $\mathbf{Q}_i \subseteq \{\psi\sigma \mid \forall\overline{x}.\,\psi$ *occurs in* φ_Q *and* $\mathrm{dom}(\sigma) = \overline{x}$ *and* $\mathrm{ran}(\sigma) \subseteq \mathcal{T}(\mathbf{E}_i)\}$

Theorem (Stronger Herbrand)

$\varphi_E \wedge \varphi_Q$ *is unsatisfiable if and only if there exist infinite series*

- $\mathbf{E}_i$ *of finite literals sets*
- $\mathbf{Q}_i$ *of finite sets of* φ_Q *instances*

such that

- $\mathbf{Q}_i \subseteq \{\psi\sigma \mid \forall \overline{x}.\, \psi$ *occurs in* φ_Q *and* $\mathrm{dom}(\sigma) = \overline{x}$ *and* $\mathrm{ran}(\sigma) \subseteq \mathcal{T}(\mathbf{E}_i)\}$
- $\mathbf{E}_0 = \varphi_E$ *and* $\mathbf{E}_{i+1} = \mathbf{E}_i \cup \mathbf{Q}_i$

Theorem (Stronger Herbrand)

$\varphi_E \wedge \varphi_Q$ *is unsatisfiable if and only if there exist infinite series*

- $\mathbf{E}_i$ *of finite literals sets*
- $\mathbf{Q}_i$ *of finite sets of* φ_Q *instances*

such that

- $\mathbf{Q}_i \subseteq \{\psi\sigma \mid \forall \overline{x}.\, \psi \text{ occurs in } \varphi_Q \text{ and } \mathrm{dom}(\sigma) = \overline{x} \text{ and } \mathrm{ran}(\sigma) \subseteq \mathcal{T}(\mathbf{E}_i)\}$
- $\mathbf{E}_0 = \varphi_E$ *and* $\mathbf{E}_{i+1} = \mathbf{E}_i \cup \mathbf{Q}_i$
- *some* $\mathbf{E}_n$ *is unsatisfiable*

Theorem (Stronger Herbrand)

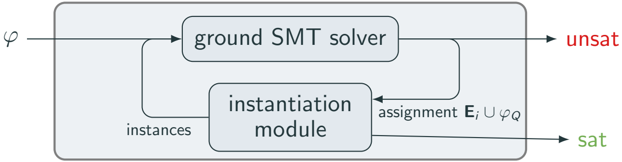
$\varphi_E \wedge \varphi_Q$ is unsatisfiable if and only if there exist infinite series

- $\mathbf{E}_i$ *of finite literals sets*
- $\mathbf{Q}_i$ *of finite sets of φ_Q instances*

such that

- $\mathbf{Q}_i \subseteq \{\psi\sigma \mid \forall \overline{x}.\, \psi$ *occurs in* φ_Q *and* $\mathrm{dom}(\sigma) = \overline{x}$ *and* $\mathrm{ran}(\sigma) \subseteq \mathcal{T}(\mathbf{E}_i)\}$
- $\mathbf{E}_0 = \varphi_E$ *and* $\mathbf{E}_{i+1} = \mathbf{E}_i \cup \mathbf{Q}_i$
- *some* $\mathbf{E}_n$ *is unsatisfiable*

Direct application in $\forall$-SMT solver



- ground solver enumerates assignments $\mathbf{E}_i \cup \varphi_Q$
- instantiation returns $\forall \overline{x}\ \psi(\overline{x}) \longrightarrow Q$ for all $Q \in \mathbf{Q}_i$ generated from $\forall \overline{x}\ \psi(\overline{x})$

Theorem (Stronger Herbrand)

$\varphi_E \wedge \varphi_Q$ is unsatisfiable if and only if there exist infinite series

- $\mathbf{E}_i$ *of finite literals sets*
- $\mathbf{Q}_i$ *of finite sets of φ_Q instances*

such that

- $\mathbf{Q}_i \subseteq \{\psi\sigma \mid \forall\overline{x}.\,\psi$ *occurs in* φ_Q *and* $\mathrm{dom}(\sigma) = \overline{x}$ *and* $\mathrm{ran}(\sigma) \subseteq \mathcal{T}(\mathbf{E}_i)\}$
- $\mathbf{E}_0 = \varphi_E$ *and* $\mathbf{E}_{i+1} = \mathbf{E}_i \cup \mathbf{Q}_i$
- *some* $\mathbf{E}_n$ *is unsatisfiable*

Lemma

if there exist infinite series $\mathbf{E}_i$, $\mathbf{Q}_i$ *such that*

- $\mathbf{Q}_i \subseteq \{\psi\sigma \mid \forall\overline{x}.\,\psi$ *occurs in* φ_Q *and* $\mathrm{dom}(\sigma) = \overline{x}$ *and* $\mathrm{ran}(\sigma) \subseteq \mathcal{T}(\mathbf{E}_i)\}$
- $\mathbf{E}_0 = \varphi_E$ *and* $\mathbf{E}_{i+1} \models \mathbf{E}_i \cup \mathbf{Q}_i$
- *and all* $\mathbf{E}_i$ *are satisfiable*

Theorem (Stronger Herbrand)

$\varphi_E \land \varphi_Q$ is unsatisfiable if and only if there exist infinite series

- $\mathbf{E}_i$ *of finite literals sets*
- $\mathbf{Q}_i$ *of finite sets of φ_Q instances*

such that

- $\mathbf{Q}_i \subseteq \{\psi\sigma \mid \forall\overline{x}.\,\psi$ *occurs in* φ_Q *and* $\mathrm{dom}(\sigma) = \overline{x}$ *and* $\mathrm{ran}(\sigma) \subseteq \mathcal{T}(\mathbf{E}_i)\}$
- $\mathbf{E}_0 = \varphi_E$ *and* $\mathbf{E}_{i+1} = \mathbf{E}_i \cup \mathbf{Q}_i$
- *some* $\mathbf{E}_n$ *is unsatisfiable*

Lemma

if there exist infinite series $\mathbf{E}_i$, $\mathbf{Q}_i$ *such that*

- $\mathbf{Q}_i \subseteq \{\psi\sigma \mid \forall\overline{x}.\,\psi$ *occurs in* φ_Q *and* $\mathrm{dom}(\sigma) = \overline{x}$ *and* $\mathrm{ran}(\sigma) \subseteq \mathcal{T}(\mathbf{E}_i)\}$
- $\mathbf{E}_0 = \varphi_E$ *and* $\mathbf{E}_{i+1} \models \mathbf{E}_i \cup \mathbf{Q}_i$
- *and all* $\mathbf{E}_i$ *are satisfiable*

then $\varphi_E \land \varphi_Q$ is satisfiable

Instantiation via enumeration

Fix ordering $>$ on tuples of terms without quantified variables.

Instantiation via enumeration

Fix ordering $>$ on tuples of terms without quantified variables.

Given assignment $\mathbf{E}_i$ from T-solver

- for each $\forall\overline{x}.\psi$ in φ_Q
  - search minimal $\overline{x}\sigma$ with respect to $\succeq$ such that $\overline{x}\sigma \in \mathcal{T}(\mathbf{E}_i)$ and $\mathbf{E}_i \not\models \psi\sigma$
  - if exists, add $\{\psi\sigma\}$ to $\mathbf{Q}_i$

Instantiation via enumeration

Fix ordering $>$ on tuples of terms without quantified variables.

Given assignment $\mathbf{E}_i$ from T-solver

- for each $\forall \overline{x}.\psi$ in φ_Q
  - search minimal $\overline{x}\sigma$ with respect to $\succeq$ such that $\overline{x}\sigma \in \mathcal{T}(\mathbf{E}_i)$ and $\mathbf{E}_i \not\models \psi\sigma$
  - if exists, add $\{\psi\sigma\}$ to $\mathbf{Q}_i$

If $\mathbf{Q}_i = \varnothing$ then sat, otherwise return $\mathbf{Q}_i$

Instantiation via enumeration

Fix ordering $>$ on tuples of terms without quantified variables.

Given assignment $\mathbf{E}_i$ from T-solver

- for each $\forall \overline{x}.\psi$ in φ_Q
  - search minimal $\overline{x}\sigma$ with respect to $\succeq$ such that $\overline{x}\sigma \in \mathcal{T}(\mathbf{E}_i)$ and $\mathbf{E}_i \not\models \psi\sigma$
  - if exists, add $\{\psi\sigma\}$ to $\mathbf{Q}_i$

If $\mathbf{Q}_i = \varnothing$ then sat, otherwise return $\mathbf{Q}_i$

Example

$$\varphi_E : P(a) \vee a = b,\ \neg P(b),\ \neg P(g(b))$$

$$\varphi_Q : \forall x.\, P(x) \vee P(f(x)),\ \forall x.\, g(x) = f(x)$$

- suppose order $a < b < f(a) < f(b) < \dots$
- ground solver: model $P(a),\ \neg P(b),\ \neg P(g(b)$ (and φ_Q)
- instantiation: $\mathbf{Q}_1$ consists of $P(b) \vee P(f(b))$ and $f(a) = g(a)$

Instantiation via enumeration

Fix ordering $>$ on tuples of terms without quantified variables.

Given assignment $\mathbf{E}_i$ from T-solver

- for each $\forall \overline{x}.\psi$ in φ_Q
  - search minimal $\overline{x}\sigma$ with respect to $\succeq$ such that $\overline{x}\sigma \in \mathcal{T}(\mathbf{E}_i)$ and $\mathbf{E}_i \not\models \psi\sigma$
  - if exists, add $\{\psi\sigma\}$ to $\mathbf{Q}_i$

If $\mathbf{Q}_i = \varnothing$ then sat, otherwise return $\mathbf{Q}_i$

Example

$$\varphi_E : P(a) \vee a = b,\ \neg P(b),\ \neg P(g(b))$$

$$\varphi_Q : \forall x.\, P(x) \vee P(f(x)),\ \forall x.\, g(x) = f(x)$$

- suppose order $a < b < f(a) < f(b) < \ldots$
- ground solver: model $P(a),\ \neg P(b),\ \neg P(g(b)$ (and φ_Q)
- instantiation: $\mathbf{Q}_1$ consists of $P(b) \vee P(f(b))$ and $f(a) = g(a)$
- ground solver: model $P(a),\ \neg P(b),\ \neg P(g(b), f(a) = g(a), P(f(b))$ (and φ_Q)

Instantiation via enumeration

Fix ordering $>$ on tuples of terms without quantified variables.

Given assignment $\mathbf{E}_i$ from T-solver

- for each $\forall \overline{x}.\psi$ in φ_Q
  - search minimal $\overline{x}\sigma$ with respect to $\succeq$ such that $\overline{x}\sigma \in \mathcal{T}(\mathbf{E}_i)$ and $\mathbf{E}_i \not\models \psi\sigma$
  - if exists, add $\{\psi\sigma\}$ to $\mathbf{Q}_i$

If $\mathbf{Q}_i = \varnothing$ then sat, otherwise return $\mathbf{Q}_i$

Example

$$\varphi_E : P(a) \vee a = b,\ \neg P(b),\ \neg P(g(b))$$

$$\varphi_Q : \forall x.\, P(x) \vee P(f(x)),\ \forall x.\, g(x) = f(x)$$

- suppose order $a < b < f(a) < f(b) < \dots$
- ground solver: model $P(a),\ \neg P(b),\ \neg P(g(b)$ (and φ_Q)
- instantiation: $\mathbf{Q}_1$ consists of $P(b) \vee P(f(b))$ and $f(a) = g(a)$
- ground solver: model $P(a),\ \neg P(b),\ \neg P(g(b), f(a) = g(a), P(f(b))$ (and φ_Q)
- instantiation: $\mathbf{Q}_2$ consists of $P(f(a)) \vee P(f(f(a)))$ and $f(b) = g(b)$

Instantiation via enumeration

Fix ordering $>$ on tuples of terms without quantified variables.

Given assignment $\mathbf{E}_i$ from T-solver

- for each $\forall \overline{x}.\psi$ in φ_Q
  - search minimal $\overline{x}\sigma$ with respect to $\succeq$ such that $\overline{x}\sigma \in \mathcal{T}(\mathbf{E}_i)$ and $\mathbf{E}_i \not\models \psi\sigma$
  - if exists, add $\{\psi\sigma\}$ to $\mathbf{Q}_i$

If $\mathbf{Q}_i = \varnothing$ then sat, otherwise return $\mathbf{Q}_i$

Example

$$\varphi_E : P(a) \vee a = b,\ \neg P(b),\ \neg P(g(b))$$

$$\varphi_Q : \forall x.\ P(x) \vee P(f(x)),\ \forall x.\ g(x) = f(x)$$

- suppose order $a < b < f(a) < f(b) < \dots$
- ground solver: model $P(a),\ \neg P(b),\ \neg P(g(b)$ (and φ_Q)
- instantiation: $\mathbf{Q}_1$ consists of $P(b) \vee P(f(b))$ and $f(a) = g(a)$
- ground solver: model $P(a),\ \neg P(b),\ \neg P(g(b), f(a) = g(a), P(f(b))$ (and φ_Q)
- instantiation: $\mathbf{Q}_2$ consists of $P(f(a)) \vee P(f(f(a)))$ and $f(b) = g(b)$
- ground solver: unsat

Bibliography

David Detlefs, Greg Nelson, and James B. Saxe.
Simplify: A Theorem Prover for Program Checking.
J. ACM, 52(3):365-473, 2005.

Andrew Reynolds, Haniel Barbosa and Pascal Fontaine.
Revisiting Enumerative Instantiation.
Proc. TACAS, pp 112–131, 2018.

Slide material partially taken from Pascal Fontaine's talk at SMT Summer School 2018.

Outline

- Summary of Last Week
- Instantiation Techniques
- More on SAT and SMT
- Test
- Evaluation

Test

- February 3, 14:15
- open book
- material includes weeks 7–12
  - Simplex and Fourier-Motzkin elimination
  - Gomory cuts
  - Nelson-Oppen
  - bitvectors
- should take approx 60 minutes (but open end)
- see test of last year

Outline

- Summary of Last Week
- Instantiation Techniques
- More on SAT and SMT
- Test
- Evaluation

Evaluation

- **LV-Code: 703048**
- **perhaps topics for comments**
  (a) Should there be more/less theory, or more/fewer applications in the course?
  (b) Which topics/exercises were interesting, which not?
  (c) Do you think you might use a SAT/SMT solver in the future?
  (d) Difficulty level of exercises too easy/too hard?
  (e) Possible improvements for course organization


CAT