

SAT and SMT Solving

Sarah Winkler

KRDB

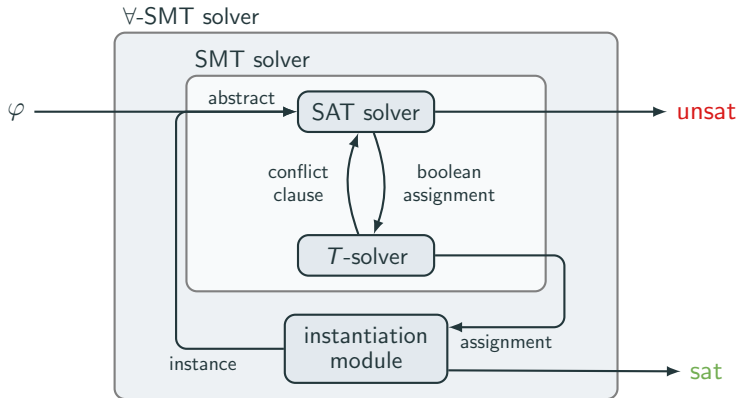
Department of Computer Science
Free University of Bozen-Bolzano

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 \forall quantifiers

Skolemization

- 1 bring formula into prenex form
- 2 replace $\forall x_1, \dots, x_k \exists y \psi[y]$ by $\forall x_1, \dots, x_k \psi[f(x_1, \dots, x_k)]$ for fresh f until no existential quantifiers left

Theorem

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

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

Definition

set of function symbols and constants

Herbrand instance of Skolem formula $\forall x_1, \dots, x_n \varphi[x_1, \dots, x_n]$ is $\varphi[t_1, \dots, t_n]$ 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 \iff

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.

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

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

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

- ▶ already unsatisfiable when replacing quantified formulas by **Herbrand instances**

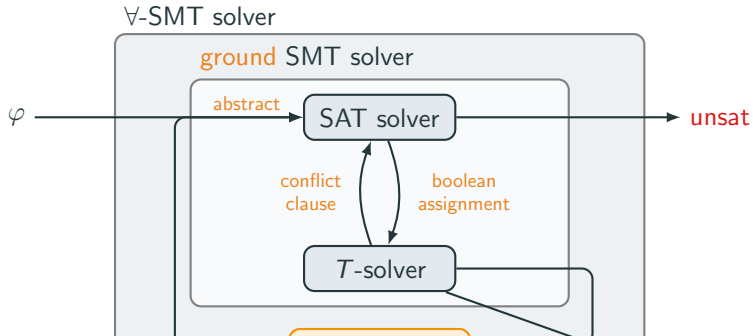
$$H(a) \longrightarrow M(a), \quad G(a) \longrightarrow H(a), \quad G(a) \wedge \neg M(a)$$

cannot be answered by SMT solver

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

Outline

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



Example

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

Instantiation

Definition (Instance)

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

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

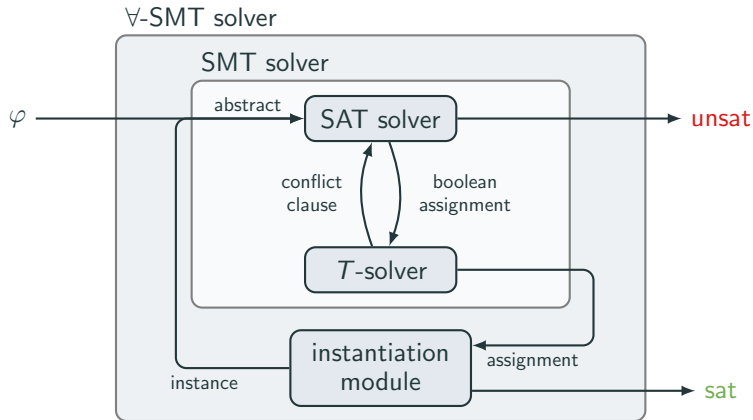
Example

$\forall x. H(x) \longrightarrow M(x)$ 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, $\forall \bar{x} \varphi(\bar{x})$ gets abstracted to propositional variable $p_{\forall \bar{x} \varphi(\bar{x})}$, 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 $p_{\forall \bar{x} \varphi(\bar{x})}$ 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

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

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

trigger

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

matching

Instantiation via E-matching

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

- ▶ select set of instantiation patterns $\{t_1, \dots, t_n\}$
- ▶ 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

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

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

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

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

- ▶ unsatisfiable ✨
- ▶ suitable instantiation patterns?
 $\text{sibling}(x, y)$ 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

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)

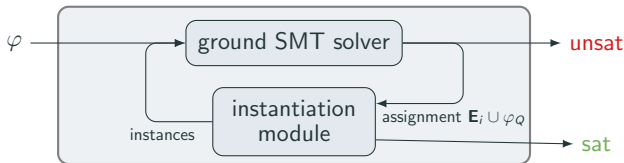
$\varphi_E \wedge \varphi_Q$ is unsatisfiable if and only if there exist infinite series

- ▶ E_i of finite literals sets
- ▶ Q_i of finite sets of φ_Q instances

such that

- ▶ $Q_i \subseteq \{\psi\sigma \mid \forall \bar{x}. \psi \text{ occurs in } \varphi_Q \text{ and } \text{dom}(\sigma) = \bar{x} \text{ and } \text{ran}(\sigma) \subseteq \mathcal{T}(E_i)\}$
- ▶ $E_0 = \varphi_E$ and $E_{i+1} = E_i \cup Q_i$
- ▶ some E_n is unsatisfiable

Direct application in \forall -SMT solver



- ▶ ground solver enumerates assignments $E_i \cup \varphi_Q$
- ▶ instantiation returns $\forall \bar{x} \psi(\bar{x}) \rightarrow Q$ for all $Q \in Q_i$ generated from $\forall \bar{x} \psi(\bar{x})$

Lemma

Instantiation via enumeration

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

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

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

If $\mathbf{Q}_i = \emptyset$ 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

- ▶ **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

