Efficient Instantiation Techniques in SMT (Work In Progress)

Haniel Barbosa

Inria Nancy – VeriDis team, LORIA
Université de Lorraine, FR
UFRN, BR

2016–07–02

Outline

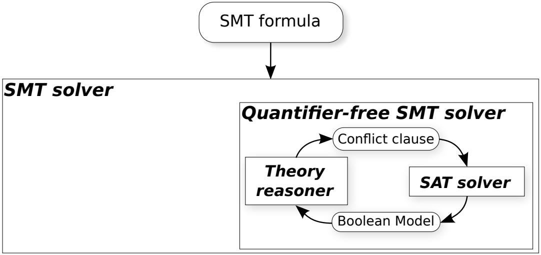
- SMT solving with quantifiers
- Instantiation framework
  - CCFV
  - Goal-oriented instantiation
  - Instances dismissal
- Conclusion and future work

SMT solving with quantifiers

SMT solving with quantifiers

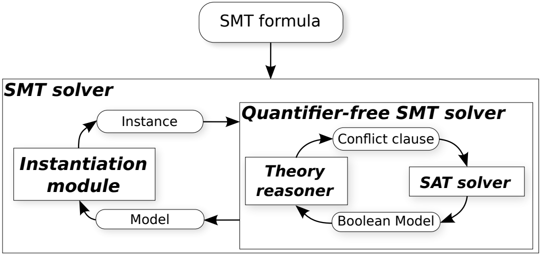
How to handle quantified formulas in the SMT context?

SMT solving with quantifiers



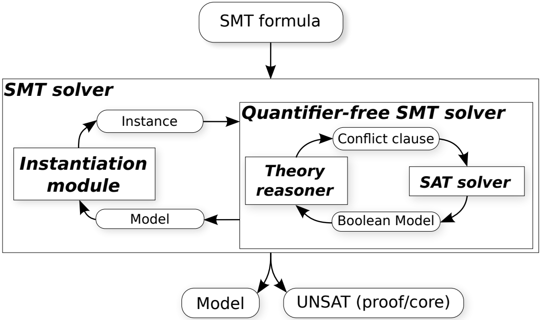
- Ground solver enumerates models $E \cup \mathcal{Q}$
  - E is a conjunctive set of ground equality literals
  - $\mathcal{Q}$ is a conjunctive set of quantified formulas

SMT solving with quantifiers



- ▷ Ground solver enumerates models $E \cup \mathcal{Q}$
  - E is a conjunctive set of ground equality literals
  - $\mathcal{Q}$ is a conjunctive set of quantified formulas
- ▷ Instead of evaluating consistency of $E \cup \mathcal{Q}$, one generally derives instantiations $\forall \mathbf{x}.\psi \rightarrow \psi\sigma$, for $\forall \mathbf{x}.\psi \in \mathcal{Q}$, and lets the ground solver sort it out.

SMT solving with quantifiers



- ▷ Ground solver enumerates models $E \cup \mathcal{Q}$
  - ▶ E is a conjunctive set of ground equality literals
  - ▶ $\mathcal{Q}$ is a conjunctive set of quantified formulas
- ▷ Instead of evaluating consistency of $E \cup \mathcal{Q}$, one generally derives instantiations $\forall \mathbf{x}.\psi \rightarrow \psi\sigma$, for $\forall \mathbf{x}.\psi \in \mathcal{Q}$, and lets the ground solver sort it out.

Triggers

- Models should be evaluated quickly
- With too many instances available, their selection becomes crucial

Triggers

- ▷ Models should be evaluated quickly
- ▷ With too many instances available, their selection becomes crucial

Triggers (matching triggers, matching patterns) [DNS05]

- ▷ Sets of terms and predicates which combined have all the bound variables of a quantifier
- ▷ Grounding the trigger yields a ground instantiation for the quantifier
- ▷ Instantiations are computed with *E*-matching:

$$E \models s_1\sigma \simeq t_1\sigma \wedge \cdots \wedge s_n\sigma \simeq t_n\sigma$$

E-matching

Example

Let $E \cup \mathcal{Q}$ be s.t. $\mathcal{Q} = \{\forall x, y.\ f(x) \simeq t \vee p(g(y))\}$

$\triangleright$ $T = \{f(x), g(y)\}$

E-matching

Example

Let $E \cup \mathcal{Q}$ be s.t. $\mathcal{Q} = \{\forall x, y.\ f(x) \simeq t \vee p(g(y))\}$

$\triangleright$ $T = \{f(x), g(y)\}$

Computes substitutions σ s.t.

$E \models f(x)\sigma \simeq f(t) \wedge g(y)\sigma \simeq g(t')$, for all $f(t)$ and $g(t')$ appearing in E

E-matching

Example

Let $E \cup \mathcal{Q}$ be s.t. $\mathcal{Q} = \{\forall x, y.\ f(x) \simeq t \vee p(g(y))\}$

$\triangleright$ $T = \{f(x), g(y)\}$

Computes substitutions σ s.t.

$E \models f(x)\sigma \simeq f(t) \wedge g(y)\sigma \simeq g(t')$, for all $f(t)$ and $g(t')$ appearing in E

yielding instantiations

$$\forall x, y.\ f(x) \simeq t \vee p(g(y)) \rightarrow (f(x) \simeq t \vee p(g(y)))\sigma$$

Instantiation framework

CCFV

CCFV: Congruence Closure with Free Variables

- A calculus to lift the ground Congruence Closure procedure to FOL
- Handles conjunctions of non-ground equality literals, yielding ground substitutions solving an *E*-unification problem:

$$E \models s_1\sigma \stackrel{.}{\simeq} t_1\sigma \wedge \cdots \wedge s_n\sigma \stackrel{.}{\simeq} t_n\sigma$$

- Provides a common framework for our instantiation techniques
- Amenable to the efficient implementation techniques of the classical algorithm

CCFV calculus

$$\frac{L,\ x \mathrel{\dot{\not\simeq}} y \parallel U}{L \parallel U \cup \{x \mathrel{\dot{\not\simeq}} y\}}\ (\text{RV})$$

(i) $\dot{\not\simeq} \in \{\simeq, \not\simeq\}$
(ii) x or y is free in U, or $E \cup U \models x \mathrel{\dot{\not\simeq}} y$

$$\frac{L,\ x \mathrel{\dot{\not\simeq}} t \parallel U}{L \parallel U \cup \{x \mathrel{\dot{\not\simeq}} t\}}\ (\text{RT})$$

(i) $\dot{\not\simeq} \in \{\simeq, \not\simeq\}$
(ii) either x is free in U or $E \cup U \models x \mathrel{\dot{\not\simeq}} t$

$$\frac{L, f(u) \simeq f(v) \parallel U}{L, u \simeq v \parallel U}\ (\textsc{Decompose})$$

$$\frac{L \parallel U}{\top}\ (\textsc{Yield})$$

(i) $L = \varnothing$ or $E \models L$

$$\frac{L,\ f(u) \mathrel{\dot{\not\simeq}} t \parallel U}{L,\ u \simeq t_1 \parallel U \quad \cdots \quad L,\ u \simeq t_n \parallel U}\ (\textsc{Ematch})$$

(i) $\dot{\not\simeq} \in \{\simeq, \not\simeq\}$
(ii) $f(t_i)$ are ground terms from E
(iii) $E \models t \mathrel{\dot{\not\simeq}} f(t_i)$, for $1 \leq i \leq n$

$$\frac{L,\ u \mathrel{\dot{\not\simeq}} f(u') \parallel U}{L,\ u \simeq t_{1,1},\ u' \simeq t'_1 \parallel U \quad \cdots \quad L,\ u \simeq t_{1,m_1},\ u' \simeq t'_1 \parallel U \quad \cdots \quad L,\ u \simeq t_{n,m_n},\ u' \simeq t'_n \parallel U}\ (\textsc{Euni})$$

(i) $\dot{\not\simeq} \in \{\simeq, \not\simeq\}$
(ii) $t_{i,j},\ f(t'_i)$ are ground terms from E
(iii) $E \models t_{i,j} \mathrel{\dot{\not\simeq}} f(t'_i)$, for $1 \leq i \leq n$, $1 \leq j \leq m_i$

$$\frac{L \parallel U}{\bot}\ (\textsc{Close})$$

(i) L is inconsistent modulo E or no other rule can be applied

CCFV algorithm

- Any derivation strategy based on the calculus yields a terminating procedure
- Backtracks may be necessary (non-proof confluent calculus)
- A successful run produces unifiers $U_1, \ldots, U_n$ representing sets of solutions

Computing ground substitutions

Unifiers U_i yields ground substitutions $\sigma_1, \ldots, \sigma_{k_i}$ s.t.

$$\sigma_j = \left\{ \; x \mapsto t \;\middle|\; \begin{array}{l} x \in X;\ U_i \models x \simeq t \text{ for some ground term } t, \text{ otherwise } t \text{ is a ground} \\ \text{term of the same sort as } x. \end{array} \right\}$$

and $E \models L\sigma_j$

CCFV algorithm

Example

Let $E \cup \mathcal{Q}$ be s.t. $\mathcal{Q} = \{\forall x, y.\ f(x) \simeq t \vee p(g(y))\}$

- $\triangleright$ $T = \{f(x), g(y)\}$

CCFV algorithm

Example

Let $E \cup \mathcal{Q}$ be s.t. $\mathcal{Q} = \{\forall x, y.\ f(x) \simeq t \vee p(g(y))\}$

$\rhd$ $T = \{f(x), g(y)\}$

Applies CCFV with E, $U = \varnothing$, $L = \{f(x) \simeq f(t), g(y) \simeq g(t')\}$, for all $f(t)$ and $g(t')$ appearing in E.

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{f(x) \simeq f(t), g(y) \simeq g(t') \parallel \varnothing}{x \simeq t, g(y) \simeq g(t') \parallel \varnothing}\ (\textsc{Decompose})}{g(y) \simeq g(t') \parallel \{x \simeq t\}}\ (\mathrm{RT})}{y \simeq t' \parallel \{x \simeq t\}}\ (\textsc{Decompose})}{\varnothing \parallel \{x \simeq t, y \simeq t'\}}\ (\mathrm{RT})}{\top}\ (\textsc{Yield})$$

The only ground substitution derivable from U is $\sigma = \{x \mapsto t, y \mapsto t'\}$

Implementation: Term Indexing

- Paramount for handling search space
- For now, top symbol indexing for ground terms:

$$f \rightarrow \begin{cases} f([t]_1, \dots, [t]_n) \\ \qquad \dots \\ f([t']_1, \dots, [t']_n) \end{cases}$$

- Either from signature table or SAT model
- Optimizations include minimizing model, bitmasks, sorting by congruence class for fast retrieval

Implementation: CCFV

- *Unifiers* data structure
  - embodies a congruence closure for free variables
  - array with each position representing a variable's valuation
  - Handled through UNION-FIND with path-compression

- Does *recursive descent E-unification* algorithm with the constraints described in the calculus

- Optimizations include memoization for avoiding recomputing expensive unifications

Impact of CCFV

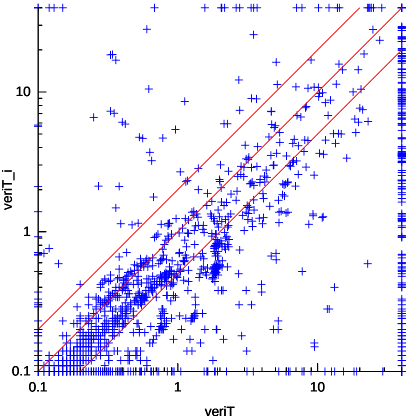


Figure : Impact of indexing and CCFV for trigger instantiation

* experiments in the "UF", "UFLIA", "UFLRA" and "UFIDL" categories of SMT-LIB, which have 10,495 benchmarks annotated as *unsatisfiable*, with 30s timeout.

Goal-oriented instantiation

Conflicting instances

- ▷ Define the refutation of the current model as a goal for instantiations

Ground conflicting instances [Reynolds et al., 2014]

- ▷ SMT solver enumerates models $E \cup \mathcal{Q}$
- ▷ Derive, for some $\forall \mathbf{x}.\psi \in \mathcal{Q}$, ground substitutions σ s.t. $E \models \neg\psi\sigma$
- ▷ Instantiations $\forall \mathbf{x}.\psi \rightarrow \psi\sigma$ refute $E \cup \mathcal{Q}$

Conflicting instances

- ▷ Define the refutation of the current model as a goal for instantiations

Ground conflicting instances [Reynolds et al., 2014]

- ▷ SMT solver enumerates models $E \cup \mathcal{Q}$
- ▷ Derive, for some $\forall \mathbf{x}.\psi \in \mathcal{Q}$, ground substitutions σ s.t. $E \models \neg\psi\sigma$
- ▷ Instantiations $\forall \mathbf{x}.\psi \rightarrow \psi\sigma$ refute $E \cup \mathcal{Q}$

Particular case of Rigid *E*-unification [TBR00]

For $\forall \mathbf{x}.\psi \in \mathcal{Q}$ in CNF and $\neg\psi = s_1 \not\doteq t_1 \wedge \cdots \wedge s_n \not\doteq t_n$, solve

$$E \models (s_1 \not\doteq t_1)\sigma \wedge \cdots \wedge (s_n \not\doteq t_n)\sigma$$

Finding conflicting instances

Finding conflicting instantiations

Apply CCFV over $\neg\psi = l_1 \wedge \cdots \wedge l_n$ and compute, **if any**, sequences of substitutions $\sigma_0, \ldots, \sigma_n$ such that

$$\sigma_0 = \varnothing;\ \sigma_{i-1} \subseteq \sigma_i \text{ and } E \models l_i\sigma_i$$

which guarantees that $E \models \neg\psi\sigma_n$

Finding conflicting instances

Example

$$
\begin{array}{rcl}
E & = & \{f(c,c) \simeq d,\ f(c,b) \simeq d\} \\
\neg\psi & = & \{g(y) \simeq g(b),\ f(x,a) \simeq f(y,z)),\ f(c,x) \simeq d\}
\end{array}
$$

Finding conflicting instances

Example

$$
\begin{array}{rcl}
E & = & \{f(c,c) \simeq d,\ f(c,b) \simeq d\} \\
\neg\psi & = & \{g(y) \simeq g(b),\ f(x,a) \simeq f(y,z)),\ f(c,x) \simeq d\}
\end{array}
$$

$$
\dfrac{g(y) \simeq g(b), f(x,a) \simeq f(y,z), f(c,x) \simeq d \parallel \varnothing}{\dfrac{f(x,a) \simeq f(y,z), f(c,x) \simeq d \parallel \{y \simeq b\}}{\dfrac{x \simeq y, z \simeq a, f(c,x) \simeq d \parallel \{y \simeq b\}}{f(c,x) \simeq d \parallel \{x \simeq y, y \simeq b, z \simeq a\}}\ (\textsc{RV, RT})}\ (\textsc{Decompose})}\ (\textsc{Decompose, RT})
$$

$$
\dfrac{f(c,x) \simeq d \parallel \{x \simeq y, y \simeq b, z \simeq a\}}{\begin{array}{ll} f(c,x) \simeq f(c,c) \parallel \{x \simeq y, y \simeq b, z \simeq a\} & (\Pi_1) \\ f(c,x) \simeq f(c,b) \parallel \{x \simeq y, y \simeq b, z \simeq a\} & (\Pi_2) \end{array}}\ (\textsc{Ematch})
$$

Finding conflicting instances

Example

$$
\begin{array}{rcl}
E & = & \{f(c,c) \simeq d,\ f(c,b) \simeq d\} \\
\neg\psi & = & \{g(y) \simeq g(b),\ f(x,a) \simeq f(y,z)),\ f(c,x) \simeq d\}
\end{array}
$$

$$
\dfrac{\dfrac{\dfrac{\dfrac{g(y) \simeq g(b), f(x,a) \simeq f(y,z), f(c,x) \simeq d \parallel \varnothing}{f(x,a) \simeq f(y,z), f(c,x) \simeq d \parallel \{y \simeq b\}}\,(\textsc{Decompose},\ \text{RT})}{x \simeq y, z \simeq a, f(c,x) \simeq d \parallel \{y \simeq b\}}\,(\textsc{Decompose})}{f(c,x) \simeq d \parallel \{x \simeq y, y \simeq b, z \simeq a\}}\,(\text{RV},\ \text{RT})}{\begin{array}{ll} f(c,x) \simeq f(c,c) \parallel \{x \simeq y, y \simeq b, z \simeq a\} & (\Pi_1) \\ f(c,x) \simeq f(c,b) \parallel \{x \simeq y, y \simeq b, z \simeq a\} & (\Pi_2) \end{array}}\,(\textsc{Ematch})
$$

$$
\dfrac{\Pi_1}{\bot}\,(\textsc{Decompose},\ \textsc{Close}) \qquad \dfrac{\dfrac{\Pi_2}{\varnothing \parallel \{x \simeq y, x \simeq b, y \simeq b, z \simeq a\}}\,(\textsc{Decompose},\ \text{RT})}{\top}\,(\textsc{Yield})
$$

The single conflicting instantiation is $\{x \mapsto b, y \mapsto b, z \mapsto a\}$.

Impact of goal conflicting instantiation

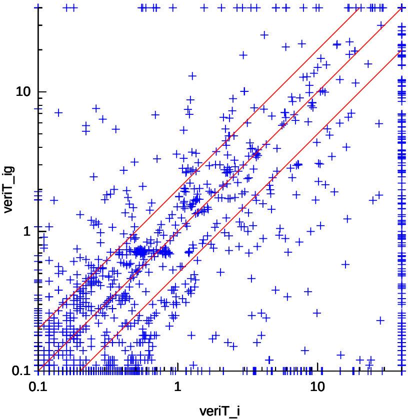


Figure : Impact of goal-oriented along with trigger instantiation

* experiments in the "UF", "UFLIA", "UFLRA" and "UFIDL" categories of SMT-LIB, which have 10,495 benchmarks annotated as *unsatisfiable*, with 30s timeout.

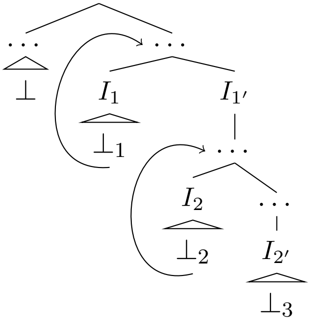
Instances dismissal

Instance dismissal

- ▷ Trigger instantiation is quite fast, but quite chaotic
- ▷ No straightforward redundancy criteria for removal of instances in SMT
- ▷ We propose a lightweight approach combining heuristic deletion [dMB07] and instantiation levels [GBT07]

Instance dismissal

- ▷ Trigger instantiation is quite fast, but quite chaotic
- ▷ No straightforward redundancy criteria for removal of instances in SMT
- ▷ We propose a lightweight approach combining heuristic deletion [dMB07] and instantiation levels [GBT07]



- ▷ SAT activity as criterion
- ▷ Only instances from previous rounds plus promoted ones are considered
- ▷ Avoids deleting instances

Instances dismissal

Example

Assume that a given instantiation $\forall \mathbf{x}.\psi \rightarrow \psi\sigma$ is derived at level 2:

$$\forall \mathbf{x}.\psi \quad \rightarrow \quad \overbrace{C_1 \wedge \cdots \wedge C_n}^{\psi\sigma}$$

$$\downarrow$$

promoted

Terms appearing in C_1 are indexed at any instantiation level, while those from the other clauses would be so only at level at least 3.

Impact of instances dismissal

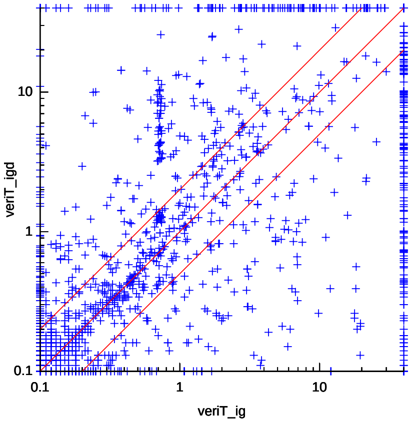


Figure : Comparison of the two main strategies

* experiments in the "UF", "UFLIA", "UFLRA" and "UFIDL" categories of SMT-LIB, which have 10,495 benchmarks annotated as *unsatisfiable*, with 30s timeout.

Conclusion and future work

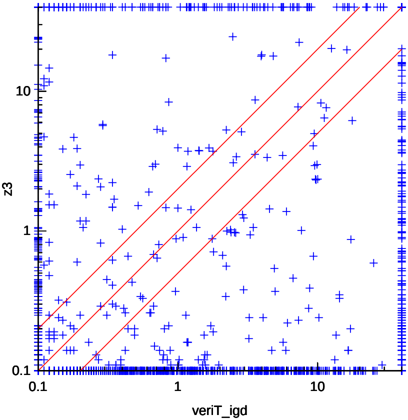
Comparison with other SMT solvers

Logic	Class	CVC4	Z3	veriT_igd	veriT_ig	veriT_i	veriT
UF	grasshopper	410	418	431	**437**	418	413
	sledgehammer	**1412**	1249	1293	1272	1134	1066
UFIDL	all	61	**62**	56	58	58	58
UFLIA	boogie	841	**852**	722	681	660	661
	sexpr	15	**26**	15	7	5	5
	grasshopper	320	341	356	**367**	340	335
	sledgehammer	**1892**	1581	1781	1778	1620	1569
	simplify	770	**831**	797	803	735	690
	simplify2	2226	**2337**	2277	2298	2291	2177
Total		**7947**	7697	7727	7701	7203	6916

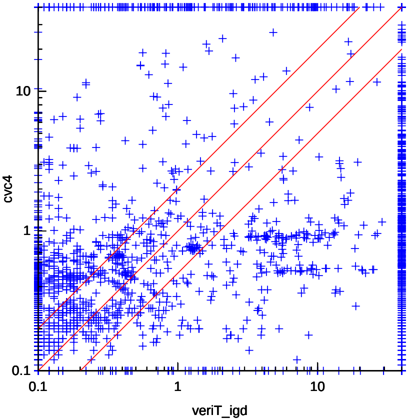
- ▷ Each veriT configuration solves ≈ 150 problems exclusively (in comparison with itself)
- ▷ $\mathrm{Z3}$ very good for arithmetic
- ▷ $\mathrm{CVC4}$ more robust, introduced goal-oriented inst in SMT

* experiments in the "UF", "UFLIA", "UFLRA" and "UFIDL" categories of SMT-LIB, which have 10,495 benchmarks annotated as *unsatisfiable*, with 30s timeout. Results over 8,701 problems which are not trivially solved by all systems.

Comparison with other SMT solvers



(a) Z3 vs veriT_igd



(b) CVC4 vs veriT_igd

* experiments in the "UF", "UFLIA", "UFLRA" and "UFIDL" categories of SMT-LIB, which have 10,495 benchmarks annotated as *unsatisfiable*, with 30s timeout.

Future work

- CCFV
  - Improve formalization
  - Learning dismatching constraints
  - Better indexing and incrementality
- Goal-oriented instantiation
  - *If you catch a tiger by the tail, don't fail*
  - Complete proof search
- Instances dismissal
  - Improve criteria (LBD, proof analysis)
  - Better promotion heuristics

Thanks

Questions?

References

Leonardo de Moura and Nikolaj Bjørner.
Efficient E-Matching for SMT Solvers.
In Frank Pfenning, editor, *Automated Deduction CADE-21*, volume 4603 of *Lecture Notes in Computer Science*, pages 183–198. Springer Berlin Heidelberg, 2007.

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

Yeting Ge, Clark Barrett, and Cesare Tinelli.
Solving Quantified Verification Conditions Using Satisfiability Modulo Theories.
In Frank Pfenning, editor, *Automated Deduction CADE-21*, volume 4603 of *Lecture Notes in Computer Science*, pages 167–182. Springer Berlin Heidelberg, 2007.

Ashish Tiwari, Leo Bachmair, and Harald Ruess.