Lecturecise 23: Satisfiability Modulo Theory Solvers

2013

Satisfiability Modulo Theories

SAT = Satisfiability for Propositional Logic

- formula: $p \land (\neg q \lor r) \land s$
- Do there exist truth values p, q, r, s that make formula true?

Satisfiability Modulo Theories

SAT = Satisfiability for Propositional Logic

- formula: $p \wedge (\neg q \vee r) \wedge s$
- Do there exist truth values p, q, r, s that make formula true? yes, e.g., $p \mapsto 1, q \mapsto 0, s \mapsto 1$ (r is any)

Satisfiability Modulo Theories

SAT = Satisfiability for Propositional Logic

- formula: $p \wedge (\neg q \vee r) \wedge s$
- Do there exist truth values p, q, r, s that make formula true? yes, e.g., $p \mapsto 1, q \mapsto 0, s \mapsto 1$ (r is any)

SMT = Satisfiability Modulo Theories (e.g. Z3 solver)

- formula: $a = b \wedge (\neg(f(a) = f(b)) \vee b = c) \wedge \neg(f(a) = f(c))$
- Do there exist values of a, b, c that makes formula true?

Satisfiability Modulo Theories

SAT = Satisfiability for Propositional Logic

- formula: $p \wedge (\neg q \vee r) \wedge s$
- Do there exist truth values p, q, r, s that make formula true? yes, e.g., $p \mapsto 1, q \mapsto 0, s \mapsto 1$ (r is any)

SMT = Satisfiability Modulo Theories (e.g. Z3 solver)

- formula: $a = b \wedge (\neg(f(a) = f(b)) \vee b = c) \wedge \neg(f(a) = f(c))$
- Do there exist values of a, b, c that makes formula true? No. We have: $a = b$, $f(a) = f(b)$, $b = c$, $a = c$, $f(a) = f(c)$, $\neg(f(a) = f(c))$ ↯

Satisfiability Modulo Theories

SAT = Satisfiability for Propositional Logic

- formula: $p \wedge (\neg q \vee r) \wedge s$
- Do there exist truth values p, q, r, s that make formula true? yes, e.g., $p \mapsto 1, q \mapsto 0, s \mapsto 1$ (r is any)

SMT = Satisfiability Modulo Theories (e.g. Z3 solver)

- formula: $a = b \wedge (\neg(f(a) = f(b)) \vee b = c) \wedge \neg(f(a) = f(c))$
- Do there exist values of a, b, c that makes formula true? No. We have: $a = b$, $f(a) = f(b)$, $b = c$, $a = c$, $f(a) = f(c)$, $\neg(f(a) = f(c))$ ↯
- Another example: $x = y \wedge f(x) < f(y)$

Satisfiability Modulo Theories

SAT = Satisfiability for Propositional Logic

- formula: $p \wedge (\neg q \vee r) \wedge s$
- Do there exist truth values p, q, r, s that make formula true? yes, e.g., $p \mapsto 1, q \mapsto 0, s \mapsto 1$ (r is any)

SMT = Satisfiability Modulo Theories (e.g. Z3 solver)

- formula: $a = b \wedge (\neg(f(a) = f(b)) \vee b = c) \wedge \neg(f(a) = f(c))$
- Do there exist values of a, b, c that makes formula true? No. We have: $a = b$, $f(a) = f(b)$, $b = c$, $a = c$, $f(a) = f(c)$, $\neg(f(a) = f(c))$ ↯
- Another example: $x = y \wedge f(x) < f(y)$: unsatisfiable

Satisfiability Modulo Theories

SAT = Satisfiability for Propositional Logic

- formula: $p \wedge (\neg q \vee r) \wedge s$
- Do there exist truth values p, q, r, s that make formula true? yes, e.g., $p \mapsto 1, q \mapsto 0, s \mapsto 1$ (r is any)

SMT = Satisfiability Modulo Theories (e.g. Z3 solver)

- formula: $a = b \wedge (\neg(f(a) = f(b)) \vee b = c) \wedge \neg(f(a) = f(c))$
- Do there exist values of a, b, c that makes formula true? No. We have: $a = b$, $f(a) = f(b)$, $b = c$, $a = c$, $f(a) = f(c)$, $\neg(f(a) = f(c))$ ↯
- Another example: $x = y \wedge f(x) < f(y)$: unsatisfiable
- Another example: $1 \leq x \wedge x \leq 2 \wedge f(x) \neq f(1)$

Satisfiability Modulo Theories

SAT = Satisfiability for Propositional Logic

- formula: $p \wedge (\neg q \vee r) \wedge s$
- Do there exist truth values p, q, r, s that make formula true? yes, e.g., $p \mapsto 1, q \mapsto 0, s \mapsto 1$ (r is any)

SMT = Satisfiability Modulo Theories (e.g. Z3 solver)

- formula: $a = b \wedge (\neg(f(a) = f(b)) \vee b = c) \wedge \neg(f(a) = f(c))$
- Do there exist values of a, b, c that makes formula true? No. We have: $a = b$, $f(a) = f(b)$, $b = c$, $a = c$, $f(a) = f(c)$, $\neg(f(a) = f(c))$ ↯
- Another example: $x = y \wedge f(x) < f(y)$: unsatisfiable
- Another example: $1 \leq x \wedge x \leq 2 \wedge f(x) \neq f(1)$: $x \mapsto 2$

Satisfiability Modulo Theories

SAT = Satisfiability for Propositional Logic

- formula: $p \wedge (\neg q \vee r) \wedge s$
- Do there exist truth values p, q, r, s that make formula true? yes, e.g., $p \mapsto 1, q \mapsto 0, s \mapsto 1$ (r is any)

SMT = Satisfiability Modulo Theories (e.g. Z3 solver)

- formula: $a = b \wedge (\neg(f(a) = f(b)) \vee b = c) \wedge \neg(f(a) = f(c))$
- Do there exist values of a, b, c that makes formula true? No. We have: $a = b$, $f(a) = f(b)$, $b = c$, $a = c$, $f(a) = f(c)$, $\neg(f(a) = f(c))$ ↯
- Another example: $x = y \wedge f(x) < f(y)$: unsatisfiable
- Another example: $1 \leq x \wedge x \leq 2 \wedge f(x) \neq f(1)$: $x \mapsto 2$

Large formulas with few no or few quantifiers (unlike pure FOL provers)

- propositional structure explored using SAT solver
- function and relation symbols come from decidable theories (quantifier-free linear arithmetic, algebraic data types)
- atomic formulas solved using decision procedures (theory solvers)
- quantifiers handled mostly by instantiation

Flattening and Extracting Propositional Structure

$$a = b \wedge (f(a) \neq f(b) \vee b = c) \wedge f(a) \neq f(c)$$

for each atomic formula introduce propositional variable:

Flattening and Extracting Propositional Structure

$$a = b \wedge (f(a) \neq f(b) \vee b = c) \wedge f(a) \neq f(c)$$

for each atomic formula introduce propositional variable:

$$\begin{array}{l} p \wedge (\neg q \vee r) \wedge \neg s \\ p \Leftrightarrow a = b \\ q \Leftrightarrow f(a) = f(b) \\ r \Leftrightarrow b = c \\ s \Leftrightarrow f(a) = f(c) \end{array}$$

flatten: give name to each subterm, e.g. f_a denotes $f(a)$:

Flattening and Extracting Propositional Structure

$$a = b \wedge (f(a) \neq f(b) \vee b = c) \wedge f(a) \neq f(c)$$

for each atomic formula introduce propositional variable:

$$\begin{array}{l} p \wedge (\neg q \vee r) \wedge \neg s \\ p \Leftrightarrow a = b \\ q \Leftrightarrow f(a) = f(b) \\ r \Leftrightarrow b = c \\ s \Leftrightarrow f(a) = f(c) \end{array}$$

flatten: give name to each subterm, e.g. f_a denotes $f(a)$:

$p \wedge (\neg q \vee r) \wedge \neg s\}$ give to SAT solver, who returns e.g. $p \wedge \neg q \wedge s$

Flattening and Extracting Propositional Structure

$$a = b \land (f(a) \neq f(b) \lor b = c) \land f(a) \neq f(c)$$

for each atomic formula introduce propositional variable:

$$\begin{array}{l} p \land (\neg q \lor r) \land \neg s \\ p \Leftrightarrow a = b \\ q \Leftrightarrow f(a) = f(b) \\ r \Leftrightarrow b = c \\ s \Leftrightarrow f(a) = f(c) \end{array}$$

flatten: give name to each subterm, e.g. f_a denotes $f(a)$:

$p \land (\neg q \lor r) \land \neg s\}$ give to SAT solver, who returns e.g. $p \land \neg q \land s$

$$\left.\begin{array}{l} p \Leftrightarrow a = b \\ q \Leftrightarrow f_a = f_b \\ r \Leftrightarrow b = c \\ s \Leftrightarrow f_a = f_c \end{array}\right\}$$

maps prop. assignment to conjunction of literals

$$a = b \land f_a \neq f_b \land f_a \neq f_c$$

Flattening and Extracting Propositional Structure

$$a = b \land (f(a) \neq f(b) \lor b = c) \land f(a) \neq f(c)$$

for each atomic formula introduce propositional variable:

$$\begin{array}{l} p \land (\neg q \lor r) \land \neg s \\ p \Leftrightarrow a = b \\ q \Leftrightarrow f(a) = f(b) \\ r \Leftrightarrow b = c \\ s \Leftrightarrow f(a) = f(c) \end{array}$$

flatten: give name to each subterm, e.g. f_a denotes $f(a)$:

$p \land (\neg q \lor r) \land \neg s\}$ give to SAT solver, who returns e.g. $p \land \neg q \land s$

$$\left.\begin{array}{l} p \Leftrightarrow a = b \\ q \Leftrightarrow f_a = f_b \\ r \Leftrightarrow b = c \\ s \Leftrightarrow f_a = f_c \end{array}\right\}$$

maps prop. assignment to conjunction of literals

$a = b \land f_a \neq f_b \land f_a \neq f_c$

$$\left.\begin{array}{l} f_a = f(a) \\ f_b = f(b) \\ f_c = f(c) \end{array}\right\}$$

each theory uses its conjuncts and definitions

$... \land a = b \land f_a \neq f_b \land f_a \neq f_c$

UNSAT, give to SAT solver : $\neg(p \land \neg q \land s)$

Formula containing function symbols and arithmetic

- f is **uninterpreted symbol** (as in FOL)
- $+, <, \leq, 1, 3, 5$ are as in linear integer arithmetic; x is of type integer

$$\underbrace{1 \leq x}_{p} \wedge \underbrace{x < 3}_{q} \wedge \Big((\underbrace{f(1) + 1 \leq f(x)}_{r} \wedge \underbrace{f(x) < f(2)}_{s}) \vee \underbrace{4 = 2x}_{t} \Big)$$

Formula containing function symbols and arithmetic

- f is **uninterpreted symbol** (as in FOL)
- $+, <, \leq, 1, 3, 5$ are as in linear integer arithmetic; x is of type integer

$$\underbrace{1 \leq x}_{p} \wedge \underbrace{x < 3}_{q} \wedge \Big((\underbrace{f(1) + 1 \leq f(x)}_{r} \wedge \underbrace{f(x) < f(2)}_{s}) \vee \underbrace{4 = 2x}_{t} \Big)$$

$$\begin{aligned}
& p \wedge q \wedge ((r \wedge s) \vee t) \wedge \\
& \quad p \Leftrightarrow 1 \leq x \wedge \\
& \quad q \Leftrightarrow x < 3 \wedge \\
& \quad r \Leftrightarrow u_1 \leq u_2 \wedge \quad u_1 = u_3 + 1 \wedge u_3 = f(u_4) \wedge u_2 = f(x) \wedge u_4 = 1 \\
& \quad s \Leftrightarrow u_2 < u_5 \wedge \quad u_5 = f(u_6) \wedge u_6 = 2 \\
& \quad t \Leftrightarrow u_7 = u_8 \wedge \quad u_7 = 4 \wedge u_8 = 2x
\end{aligned}$$

Formula containing function symbols and arithmetic

- f is **uninterpreted symbol** (as in FOL)
- $+, <, \leq, 1, 3, 5$ are as in linear integer arithmetic; x is of type integer

$$\underbrace{1 \leq x}_{p} \wedge \underbrace{x < 3}_{q} \wedge \Big(\big(\underbrace{f(1) + 1 \leq f(x)}_{r} \wedge \underbrace{f(x) < f(2)}_{s} \big) \vee \underbrace{4 = 2x}_{t} \Big)$$

$$\begin{aligned}
& p \wedge q \wedge ((r \wedge s) \vee t) \wedge \\
& \quad p \Leftrightarrow 1 \leq x \ \wedge \\
& \quad q \Leftrightarrow x < 3 \ \wedge \\
& \quad r \Leftrightarrow u_1 \leq u_2 \ \wedge \quad u_1 = u_3 + 1 \wedge u_3 = f(u_4) \wedge u_2 = f(x) \wedge u_4 = 1 \\
& \quad s \Leftrightarrow u_2 < u_5 \ \wedge \quad u_5 = f(u_6) \wedge u_6 = 2 \\
& \quad t \Leftrightarrow u_7 = u_8 \ \wedge \quad u_7 = 4 \wedge u_8 = 2x
\end{aligned}$$

Who handles which part in this example:

propositional formula	SAT solver
pure equalities ($u_7 = u_8$)	both theory solvers
highlighted formulas	solver for theory of uninterpreted functions
remaining ones	solver for theory of integer linear arithmetic

Completeness for Combination of Theories

Suppose that we have conjuncts that talk about two different theories, e.g.

- integers
- algebraic data types on some infinite set (ADTs)

Group conjuncts into those for integers and those for ADTs: $F_1 \land F_2$

- If F_1 is unsat in theory of integers, then $F_1 \land F_2$ is unsat
- If F_2 is unsat in the theory of ADTs, then $F_1 \land F_2$ is unsat

Completeness for Combination of Theories

Suppose that we have conjuncts that talk about two different theories, e.g.

- integers
- algebraic data types on some infinite set (ADTs)

Group conjuncts into those for integers and those for ADTs: $F_1 \wedge F_2$

- If F_1 is unsat in theory of integers, then $F_1 \wedge F_2$ is unsat
- If F_2 is unsat in the theory of ADTs, then $F_1 \wedge F_2$ is unsat
- What if F_1 has a model and F_2 has a model?

Can two models can be merged?

- If yes, we have complete combination of two decision procedures

Basic idea: two theories can build models as long as the parts of models that overlap are isomorphic (so they can be merged)

In practice, this works because operations are mostly **disjoint**.

ADTs have constructors and selectores, integers have $+$

Merging models is like merging graphs with disjoint edges. Must sure:

- distinct variables are distinct in both models (share equalities!)
- models can be made to have same cardinality (often: require each model can be made infinite)

Theory of Uninterpreted Function Symbols

Quantifier-free first-order logic with equality
Assume it is interpreted over an infinite domain
Assume no relation symbols: replace $R(t_1, \ldots, t_n)$ with $f_R(t_1, \ldots, t_n) = T$ for some fresh constant T
SAT solver handles disjunctions: assume conjunction of equalities and disequalities
Key inference rule, for each function symbol f of n arguments:

$$\frac{t_1 = t_1' \quad \ldots \quad t_n = t_n'}{f(t_1, \ldots, t_n) = f(t_1', \ldots, t_n')}$$

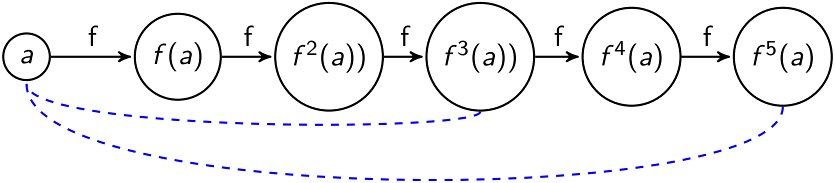
Also: "$=$" is equivalence relation and $t \neq t$ is contradictory
Apply these rules only to those terms that occur in the formula
Implementation: E-graph stores congruence relation computed so far.
Applying rules: merging nodes in this graph

Example of Running the Algorithm

Let $f^k(a)$ denote $f(\dots f(a)\dots)$ with k-fold application of f. Consider

$$f^3(a) = a \wedge f^5(a) = a \wedge f^2(a) \neq a$$

Apply the congruence closure algorithm to check its satisfiability.
Initial graph of all ground terms and the given equalities:



Congruence rule in this case: $x = y \to f(x) = f(y)$
Equivalence maintained using union-find algorithm
Conjunction is satisfiable $\Leftrightarrow$ there is literal $t_1 \neq t_2$ where t_1, t_2 are merged
$\Leftarrow$): by properties of equality, conclusions are sound
$\Rightarrow$): computed congruence extends to congruence on the Herbrand model

Herbrand-Like Theorem for Equality

For every set of formulas with equality S the following are equivalent

- S has a model
- $S' \cup AxEq$ has a model (where $AxEq$ are Axioms for Equality: congruence+equivalence), and S' is result of replacing in S '=' with 'eq' symbol that is axiomatized;
- S has a model whose domain is the quotient $[GT]$ of the set of ground terms under some congruence.

Given Herbrand model (GT, α) where eq satisfies axioms of equality, we define quotient of Herbrand model. For each element $x \in GT$, define
$[x] = \{y \mid (x, y) \in \alpha(eq)\}$ and $[GT] = \{[x] \mid x \in GT\}$
The constructed model is $I_Q = ([GT], \alpha_Q)$ where

$$\alpha_Q(R) = \{([x_1], \ldots, [x_n]) \mid (x_1, \ldots, x_n) \in \alpha(R)\}$$

In particular, when R is eq we have

$$alpha_Q(eq) = \{([x_1], [x_2]) \mid (x_1, x_2) \in \alpha(eq)\} = \{(a, a) \mid a \in D\}$$

Functions are a special case of relations (note that result is unique):

$$\alpha_Q(f) = \{([x_1], \ldots, [x_n], [x_{n+1}]) \mid (x_1, \ldots, x_n, x_{n+1}) \in \alpha(f)\}$$

Quantifier Instantiation During SMT Solving Process

$$G \wedge \forall x.F(x) \;\; \rightsquigarrow \;\; G \wedge F(t) \wedge \forall x.F(x)$$

where t is a term occurring in G

- this can go on forever
- in general this is incomplete: may need to invent terms that do not occur
- even in the limit it is not complete with respect to the ideal semantics of e.g. integers (theory of quantified integers is not even enumerable)

Controlling the instantiation process using **triggers**

- for each quantified formula $\forall \bar{x}.F(\bar{x})$ require a pattern $P(\bar{x})$ that contains all free variables in $F(\bar{x})$
- instantiate $F(\bar{x})$ only if the the pattern $P(x)$ occurs in the ground formula so far
- introduced in Simplify: a theorem prover for program checking

More information in these papers

- Solving Quantified Verification Conditions using Satisfiability Modulo Theories
- Efficient E-matching for SMT solvers