Satisfiability Modulo Theories

Tjark Weber

webertj@in.tum.de

TECHNISCHE UNIVERSITÄT MÜNCHEN

Oberseminar Statische Analyse

November 11, 2004

Goal

To decide the satisfiability of formulas with respect to decidable background theories …

$$\phi ::= \mathcal{A} \mid \neg\phi \mid \phi \vee \phi \mid \phi \wedge \phi$$

Applications:

- Formal verification
- Scheduling
- Compiler optimization
- …

Goal

To decide the satisfiability of formulas with respect to decidable background theories ...

$$\phi ::= \mathcal{A} \mid \neg\phi \mid \phi \vee \phi \mid \phi \wedge \phi$$

... using a *combination* of SAT solving and theory-specific decision procedures.

Applications:

- Formal verification
- Scheduling
- Compiler optimization
- ...

People

- Armando, Alessandro (U. of Genova)
- Barrett, Clark (New York U.)
- Berezin, Sergey (Stanford U.)
- Castellini, Claudio (U. of Genova)
- Cimatti, Alessandro (IRST-ITC)
- Cok, David (Eastman Kodak Company)
- Flanagan, Cormac (UC Santa Cruz)
- Fontaine, Pascal (U. of Liège)
- Ganesh, Vijay (Stanford U.)
- Giunchiglia, Enrico (U. of Genova)
- Kiniry, Joseph (U. of Nijmegen and KindSoftware LCC)
- Krstic, Sava (Strategic CAD Labs, Intel Corporation)
- Harrison, John (Intel)
- Janicic, Predrag (U. of Belgrade)
- Lahiri, Shuvendu (Carnegie Mellon U.)

People, cntd.

- Joshi, Rajeev (NASA JPL)
- de Moura, Leonardo (SRI International)
- Nelson, Greg (HP Laboratories)
- Ranise, Silvio (INRIA-Lorraine)
- Ringeissen, Christophe (INRIA-Lorraine)
- Ruess, Harald (SRI International)
- Saxe, Jim (Compaq SRC)
- Sebastiani, Roberto (U. of Trento)
- Seshia, Sanjit (Carnegie Mellon U.)
- Shankar, Natarajan (SRI International)
- Strichman, Ofer (Technion U.)
- Stump, Aaron (Washington U.)
- Tinelli, Cesare (U. of Iowa)
- Zarba, Calogero (INRIA-Lorraine)

Source: `http://goedel.cs.uiowa.edu/smtlib/group.html`

Some SMT Systems

- Current:
  - Argo-lib
  - DPLL(T)
  - CVC Lite
  - haRVey
  - ICS
  - Math-SAT
  - Tsat++
  - UCLID
- Old:
  - CVC
  - LPSAT
  - RDL
  - Simplify
  - STeP
  - SVC
  - Tsat

Source: `http://goedel.cs.uiowa.edu/smtlib/solvers.html`

Combining Decision Procedures

Theories:

- $\mathcal{R}$: theory of rationals
  $\Sigma_{\mathcal{R}} = \{\leq, +, -, 0, 1\}$
- $\mathcal{L}$: theory of lists
  $\Sigma_{\mathcal{L}} = \{=, \mathrm{hd}, \mathrm{tl}, \mathrm{nil}, \mathrm{cons}\}$
- $\mathcal{E}$: theory of equality
  Σ: free function and predicate symbols

Problem: Is

$$x \leq y \wedge y \leq x + \mathrm{hd}(\mathrm{cons}(0, \mathrm{nil})) \wedge P(h(x) - h(y)) \wedge \neg P(0)$$

satisfiable in $\mathcal{R} \cup \mathcal{L} \cup \mathcal{E}$?

The Nelson-Oppen Procedure

G. Nelson and D.C. Oppen: *Simplification by cooperating decision procedures*, ACM Trans. on Programming Languages and Systems, 1(2):245-257, 1979.

Given:

- $\mathcal{T}_1$, $\mathcal{T}_2$ first-order theories with signatures Σ_1, Σ_2
- $\Sigma_1 \cap \Sigma_2 = \emptyset$
- ϕ quantifier-free formula over $\Sigma_1 \cup \Sigma_2$

Obtain a decision procedure for satisfiability in $\mathcal{T}_1 \cup \mathcal{T}_2$ from decision procedures for satisfiability in $\mathcal{T}_1$ and $\mathcal{T}_2$.

Nelson-Oppen: Example

Variable abstraction + *equality propagation*:

$$x \leq y \wedge y \leq x + \mathrm{hd}(\mathrm{cons}(0, \mathrm{nil})) \wedge P(h(x) - h(y)) \wedge \neg P(0)$$

Nelson-Oppen: Example

Variable abstraction + *equality propagation*:

$$x \leq y \wedge y \leq x + \underbrace{\mathrm{hd}(\mathrm{cons}(0, \mathrm{nil}))}_{v_1} \wedge P(\underbrace{\underbrace{h(x)}_{v_3} - \underbrace{h(y)}_{v_4}}_{v_2}) \wedge \neg P(\underbrace{0}_{v_5})$$

Nelson-Oppen: Example

Variable abstraction + *equality propagation*:

$$x \leq y \wedge y \leq x + \underbrace{\mathrm{hd}(\mathrm{cons}(0, \mathrm{nil}))}_{v_1} \wedge P(\underbrace{\underbrace{h(x)}_{v_3} - \underbrace{h(y)}_{v_4}}_{v_2}) \wedge \neg P(\underbrace{0}_{v_5})$$

$\mathcal{R}$	$\mathcal{L}$	$\mathcal{E}$
$x \leq y$		$P(v_2)$
$y \leq x + v_1$		$\neg P(v_5)$

Nelson-Oppen: Example

Variable abstraction + *equality propagation*:

$$x \leq y \wedge y \leq x + \underbrace{\mathrm{hd}(\mathrm{cons}(0, \mathrm{nil}))}_{v_1} \wedge P(\underbrace{\underbrace{h(x)}_{v_3} - \underbrace{h(y)}_{v_4}}_{v_2}) \wedge \neg P(\underbrace{0}_{v_5})$$

$\mathcal{R}$	$\mathcal{L}$	$\mathcal{E}$
$x \leq y$		$P(v_2)$
$y \leq x + v_1$		$\neg P(v_5)$
$v_2 = v_3 - v_4$	$v_1 = \mathrm{hd}(\mathrm{cons}(v_5, \mathrm{nil}))$	$v_3 = h(x)$
$v_5 = 0$		$v_4 = h(y)$

Nelson-Oppen: Example

Variable abstraction + *equality propagation*:

$$x \leq y \wedge y \leq x + \underbrace{\mathrm{hd}(\mathrm{cons}(0, \mathrm{nil}))}_{v_1} \wedge P(\underbrace{\underbrace{h(x)}_{v_3} - \underbrace{h(y)}_{v_4}}_{v_2}) \wedge \neg P(\underbrace{0}_{v_5})$$

$\mathcal{R}$	$\mathcal{L}$	$\mathcal{E}$
$x \leq y$		$P(v_2)$
$y \leq x + v_1$		$\neg P(v_5)$
$v_2 = v_3 - v_4$	$v_1 = \mathrm{hd}(\mathrm{cons}(v_5, \mathrm{nil}))$	$v_3 = h(x)$
$v_5 = 0$		$v_4 = h(y)$
	$v_1 = v_5$	

Nelson-Oppen: Example

Variable abstraction + *equality propagation*:

$$x \leq y \wedge y \leq x + \underbrace{\mathrm{hd}(\mathrm{cons}(0, \mathrm{nil}))}_{v_1} \wedge P(\underbrace{\underbrace{h(x)}_{v_3} - \underbrace{h(y)}_{v_4}}_{v_2}) \wedge \neg P(\underbrace{0}_{v_5})$$

$\mathcal{R}$	$\mathcal{L}$	$\mathcal{E}$
$x \leq y$		$P(v_2)$
$y \leq x + v_1$		$\neg P(v_5)$
$v_2 = v_3 - v_4$	$v_1 = \mathrm{hd}(\mathrm{cons}(v_5, \mathrm{nil}))$	$v_3 = h(x)$
$v_5 = 0$		$v_4 = h(y)$
$x = y$	$v_1 = v_5$	

Nelson-Oppen: Example

Variable abstraction + *equality propagation*:

$$x \le y \wedge y \le x + \underbrace{\mathrm{hd}(\mathrm{cons}(0, \mathrm{nil}))}_{v_1} \wedge P(\underbrace{\underbrace{h(x)}_{v_3} - \underbrace{h(y)}_{v_4}}_{v_2}) \wedge \neg P(\underbrace{0}_{v_5})$$

$\mathcal{R}$	$\mathcal{L}$	$\mathcal{E}$
$x \le y$		$P(v_2)$
$y \le x + v_1$		$\neg P(v_5)$
$v_2 = v_3 - v_4$	$v_1 = \mathrm{hd}(\mathrm{cons}(v_5, \mathrm{nil}))$	$v_3 = h(x)$
$v_5 = 0$		$v_4 = h(y)$
$x = y$	$v_1 = v_5$	$v_3 = v_4$

Nelson-Oppen: Example

Variable abstraction + *equality propagation*:

$$x \le y \wedge y \le x + \underbrace{\mathrm{hd}(\mathrm{cons}(0, \mathrm{nil}))}_{v_1} \wedge P(\underbrace{\underbrace{h(x)}_{v_3} - \underbrace{h(y)}_{v_4}}_{v_2}) \wedge \neg P(\underbrace{0}_{v_5})$$

$\mathcal{R}$	$\mathcal{L}$	$\mathcal{E}$
$x \le y$		$P(v_2)$
$y \le x + v_1$		$\neg P(v_5)$
$v_2 = v_3 - v_4$	$v_1 = \mathrm{hd}(\mathrm{cons}(v_5, \mathrm{nil}))$	$v_3 = h(x)$
$v_5 = 0$		$v_4 = h(y)$
$x = y$	$v_1 = v_5$	$v_3 = v_4$
$v_2 = v_5$		

Nelson-Oppen: Example

Variable abstraction + *equality propagation*:

$$x \le y \wedge y \le x + \underbrace{\mathrm{hd}(\mathrm{cons}(0, \mathrm{nil}))}_{v_1} \wedge P(\underbrace{\underbrace{h(x)}_{v_3} - \underbrace{h(y)}_{v_4}}_{v_2}) \wedge \neg P(\underbrace{0}_{v_5})$$

$\mathcal{R}$	$\mathcal{L}$	$\mathcal{E}$
$x \le y$		$P(v_2)$
$y \le x + v_1$		$\neg P(v_5)$
$v_2 = v_3 - v_4$	$v_1 = \mathrm{hd}(\mathrm{cons}(v_5, \mathrm{nil}))$	$v_3 = h(x)$
$v_5 = 0$		$v_4 = h(y)$
$x = y$	$v_1 = v_5$	$v_3 = v_4$
$v_2 = v_5$		$\perp$

Extensions and Related Work

- Relaxations of the *disjointness* requirement
- Nelson-Oppen is sound for combinations of *stably-infinite* theories

- R.E. *Shostak*: *Deciding Combinations of Theories*. J. of the ACM, 31(1):1-12, 1984
- Combinations of *unification* algorithms [F. Baader, K. Schulz]

SAT Solving: DPLL

M. Davis, G. Logemann, D. Loveland: *A machine program for theorem-proving*. Communications of the ACM, 5(7):394-397, 1962.

```
dpll(φ:Boolean formula, θ:partial assignment) {
   θ' := deduce(φ, θ);
   φ' := eval(φ, θ');
   if φ'=True then return θ'
   else if φ'=False then return UNSATISFIABLE
   else {
      x := choose_fresh_variable(φ', θ');
      result := dpll(φ', θ' ∪ {x ↦ True});
      if result=UNSATISFIABLE then
         return dpll(φ', θ' ∪ {x ↦ False})
      else return result
   }
}
```

Combining Nelson-Oppen and DPLL

```
satisfy(ϕ:formula) {
    create mapping Γ from Boolean variables to
      atomic formulas;
    while True {
      θ := dpll(Γ⁻¹(ϕ), ∅);
      if θ = UNSATISFIABLE then return θ
      else {
        Θ := Γ(θ);
        if n-o(Θ) = SATISFIABLE then return Θ
        else ϕ := ϕ ∧ ¬Θ
      }
    }
}
```

Optimizations and Variants

- Gilles Audemard, Piergiorgio Bertoli, Alessandro Cimatti, Artur Kornilowicz, Roberto Sebastiani: *A SAT Based Approach for Solving Formulas over Boolean and Linear Mathematical Propositions*. 18th International Conference on Automated Deduction (CADE 2002), Copenhagen, Denmark, July 2002. *Math-SAT*
- Cormac Flanagan, Rajeev Joshi, Xinming Ou, James B. Saxe: *Theorem Proving using Lazy Proof Explication*. 15th International Conference on Computer Aided Verification (CAV 2003), Boulder, USA, July 2003. *Verifun*
- Harald Ganzinger, George Hagen, Robert Nieuwenhuis, Albert Oliveras, Cesare Tinelli: *DPLL(T): Fast Decision Procedures*. 16th International Conference on Computer Aided Verification (CAV 2004), Boston, USA, July 2004. *DPLL(T)*

Optimizations: Math-SAT

- Preprocessing atoms
  Atoms are rewritten into *normal form*, using theory-specific facts (associativity, commutativity, ...).

Optimizations: Math-SAT

- Preprocessing atoms
  Atoms are rewritten into *normal form*, using theory-specific facts (associativity, commutativity, . . .).

- Several layers of decision procedures
  More powerful procedures are *invoked only when weaker ones fail* to show unsatisfiability.

Optimizations: Math-SAT

- Preprocessing atoms
  Atoms are rewritten into *normal form*, using theory-specific facts (associativity, commutativity, . . .).

- Several layers of decision procedures
  More powerful procedures are *invoked only when weaker ones fail* to show unsatisfiability.

- Early pruning
  Partial Boolean assignments are tested by the theory-specific decision procedure.

Optimizations: Math-SAT

- Preprocessing atoms
  Atoms are rewritten into *normal form*, using theory-specific facts (associativity, commutativity, . . .).
- Several layers of decision procedures
  More powerful procedures are *invoked only when weaker ones fail* to show unsatisfiability.
- Early pruning
  Partial Boolean assignments are tested by the theory-specific decision procedure.
- Enhanced early pruning
  Information gained from partial assignments is *passed back* to the SAT solver.

Optimizations: Math-SAT, Verifun

- Online SAT solving

  The SAT solver *continues* its search after accepting additional clauses (rather than to restart from scratch).

Optimizations: Math-SAT, Verifun

- Online SAT solving
  The SAT solver *continues* its search after accepting additional clauses (rather than to restart from scratch).
- Proof explication/mathematical learning
  The theory-specific decision procedures generate *lemmas*.

Optimizations: Math-SAT, Verifun

- Online SAT solving
  The SAT solver *continues* its search after accepting additional clauses (rather than to restart from scratch).
- Proof explication/mathematical learning
  The theory-specific decision procedures generate *lemmas*.
  - Lazy/eager

Optimizations: Math-SAT, Verifun

- Online SAT solving
  The SAT solver *continues* its search after accepting additional clauses (rather than to restart from scratch).
- Proof explication/mathematical learning
  The theory-specific decision procedures generate *lemmas*.
  - Lazy/eager
  - Fine-grain/coarse-grain

Optimizations: Math-SAT, Verifun

- Online SAT solving
  The SAT solver *continues* its search after accepting additional clauses (rather than to restart from scratch).
- Proof explication/mathematical learning
  The theory-specific decision procedures generate *lemmas*.
  - Lazy/eager
  - Fine-grain/coarse-grain
  - Hiding of new proxy variables

Optimizations: Math-SAT, Verifun

- Online SAT solving

  The SAT solver *continues* its search after accepting additional clauses (rather than to restart from scratch).

- Proof explication/mathematical learning

  The theory-specific decision procedures generate *lemmas*.

  - Lazy/eager
  - Fine-grain/coarse-grain
  - Hiding of new proxy variables

- Mathematical backjumping

  The solver jumps back to the deepest branching point in which a literal *contributing to a conflict* was assigned a value.

Optimizations: DPLL(T)

- *Tight integration* of the theory-specific decision procedure with the DPLL framework:
  - `Initialize(`$\mathcal{L}$`:literal set)`
  - `SetTrue(l:`$\mathcal{L}$`-literal):`$\mathcal{L}$`-literal set`
  - `IsTrue?(l:`$\mathcal{L}$`-literal):bool`
  - `Backtrack(n:`$\mathbb{N}$`)`
  - `Explanation(l:`$\mathcal{L}$`-literal):`$\mathcal{L}$`-literal set`

  The solver maintains a stack of all $\mathcal{L}$-literals that are true in a partial interpretation.

Future Work

- Better (theory-dependent) *heuristics* for . . .
  - lemma management
  - literal selection
  - restarting
- Extension of existing SMT systems with decision procedures for *other theories*