The theorem-proving method DPLL($\Gamma+\mathcal{T}$)[1]

Maria Paola Bonacina

Dipartimento di Informatica
Università degli Studi di Verona
Verona, Italy, EU

Formal Topics Series
Computer Science Laboratory, SRI International, Menlo Park, California, USA
February and March 2016

(It subsumes a talk given at the Annual Meeting of the IFIP Working Group 1.6 on Term Rewriting held at the Federated Logic Conference (FLoC), Edinburgh, Scotland, UK, 10 July 2010)

[1] Joint work with Leo de Moura and Chris Lynch

Introduction

DPLL($\Gamma+\mathcal{T}$) is a theorem-proving method that

- Integrates SMT-solver DPLL($\mathcal{T}$) and first-order inference system Γ
- Combines built-in and axiomatized theories
- Makes first-order inferences model-driven by the candidate model built by the SMT-solver
- Yields some decision procedures for satisfiability of first-order formulæ

Motivation

- Formulæ from applications (e.g., verification) involve
  - Background theories (e.g., linear arithmetic, data structures)
  - Quantifiers to write, e.g.,
    - Invariants
    - Axioms of application-specific theories without decision procedure
- Objective: have both theory reasoning and reasoning about quantifiers
- Not even semi-decidable in general

Preliminary assumptions

- Background theory $\mathcal{T}$
  - $\mathcal{T} = \bigcup_{i=1}^{n} \mathcal{T}_i$
- Set of formulæ: $\mathcal{R} \cup P$
  - $\mathcal{R}$: set of non-ground clauses without $\mathcal{T}$-symbols
  - P: set of ground clauses
    typically with both $\mathcal{T}$-symbols and $\mathcal{R}$-symbols
- Determine whether $\mathcal{R} \cup P$ is satisfiable modulo $\mathcal{T}$

Some key state-of-the-art reasoning methods

- DPLL-CDCL procedure for SAT
- $\mathcal{T}_i$-solvers: Satisfiability procedures for the $\mathcal{T}_i$'s
- Satisfiability procedure for $\mathcal{T}$ via combination by equality sharing (aka Nelson-Oppen) of the $\mathcal{T}_i$-satisfiability procedures
- DPLL($\mathcal{T}$)-based SMT-solver
- First-order engine Γ to handle $\mathcal{R}$ (additional theory): Resolution+Rewriting+Superposition: Superposition-based

Theory combination by equality sharing

- Theories $\mathcal{T}_1, \ldots, \mathcal{T}_n$ with $\mathcal{T}_i$-satisfiability procedures
- $\mathcal{T} = \bigcup_{i=1}^{n} \mathcal{T}_i$
- Disjoint: share only $\simeq$ and uninterpreted constants
- Mixed terms separated by introducing new constants (e.g., $f(g(a)) \simeq b$ becomes $f(c) \simeq b \wedge g(a) \simeq c$, with c new, if f and g belong to different theories)
- Need to agree on:
  - Shared constants
  - Cardinalities of shared sorts

Theory combination by equality sharing

- Compute arrangement: which shared constants are equal and which are not
- $\mathcal{T}_i$-solvers generate and propagate all entailed (disjunctions of) equalities between shared constants
- For cardinalities: assume stably infinite:
  every $\mathcal{T}_i$-satisfiable ground formula has $\mathcal{T}_i$-model with infinite cardinality

Superposition-based inference system Γ

- FOL+= clauses with universally quantified variables
- Axiomatized theories
- Deduce clauses from clauses (expansion)
- Remove redundant clauses (contraction)
- Well-founded ordering $\succ$ on terms and literals to restrict expansion and define contraction
- Semi-decision procedure for unsatisfiability
- No backtracking

Ordering-based inferences

Ordering $\succ$ on terms and literals to

- restrict expansion inferences
- define contraction inferences

Complete Simplification Ordering:

- stable: if $s \succ t$ then $s\sigma \succ t\sigma$
- monotone: if $s \succ t$ then $l[s] \succ l[t]$
- subterm property: $l[t] \succeq t$
- total on ground terms and literals

Inference system Γ

State of derivation: set of clauses S

- Expansion rules:
  - Resolution: resolve maximal complementary literals
  - Superposition: superpose maximal side of maximal equation into maximal side of maximal (in)equation
  - Paramodulation: superpose maximal side of maximal equation into maximal literal
  - Factoring rules
- Contraction rules:
  - Simplification by well-founded rewriting
  - Subsumption of less general clauses ($C\sigma \subseteq D$ as multisets)
  - Deletion of trivial clauses

Combining strengths of different reasoning engines

- DPLL: SAT-problems; large clauses (also non-Horn)
- Theory solvers: e.g., ground equality, linear arithmetic
- DPLL($\mathcal{T}$)-based SMT-solver: efficient integration of the above
- Superposition-based inference system Γ:
  - Horn clauses, equalities, universal quantifiers (automated instantiation)
  - Satisfiability procedure for several theories of data structures

DPLL($\Gamma+\mathcal{T}$): integrate Γ in DPLL($\mathcal{T}$)

State of derivation $M \parallel F$

- Model-based deduction:
  literals in M as premises of Γ-inferences
- Stored as hypotheses in inferred clause
- Hypothetical clause: $(L_1 \wedge \ldots \wedge L_n) \triangleright (L'_1 \vee \ldots L'_m)$
  interpreted as $\neg L_1 \vee \ldots \vee \neg L_n \vee L'_1 \vee \ldots \vee L'_m$

Predecessor:
DPLL(Γ) [Leonardo de Moura and Nikolaj Bjørner, IJCAR 2008]

DPLL($\Gamma+\mathcal{T}$): integrate Γ in DPLL($\mathcal{T}$)

- Inferred clauses inherit hypotheses from premises
- Backjump: remove hypothetical clauses depending on undone assignments

DPLL(Γ+$\mathcal{T}$): division of labor

Use each engine for what is best at:

- DPLL($\mathcal{T}$) sees all and only ground clauses
- Γ sees all non-ground clauses and ground unit $\mathcal{R}$-clauses taken from M: Γ works on $\mathcal{R}$-satisfiability problem
- Both see the ground unit $\mathcal{R}$-clauses

DPLL($\Gamma+\mathcal{T}$): two modes

- Search mode: State of derivation $M \parallel F$
  - M sequence of ground literals: partial model
  - F set of hypothetical clauses
    $clauses(F)$ is the set of clauses in F stripped of the hypotheses
- Conflict resolution mode: State of derivation $M \parallel F \parallel C$
  - C ground conflict clause

Initial state: M empty, F is $\{\emptyset \triangleright C \mid C \in \mathcal{R} \cup P\}$

Model-based theory combination

A variant of equality sharing:

- Each $\mathcal{T}_i$-solver builds a candidate $\mathcal{T}_i$-model M_i
- Generate and propagate the equalities between shared constants that are true in M_i
- Less expensive than generating (disjunctions of) equalities true in all $\mathcal{T}_i$-models consistent with M
- Optimistic approach: if $t \simeq s$ inconsistent, retract, and fix M_i by backtracking
- Rationale: few equalities matter in practice

[Leonardo de Moura and Nikolaj Bjørner, SMT 2007]

Model-based theory combination in DPLL($\Gamma+\mathcal{T}$)

- PropagateEq: add to M ground $s \simeq t$ true in $\mathcal{T}_i$-model: if $M_i(t) = M_i(s)$, t and s occur in F,

$$M \parallel F \quad \Longrightarrow \quad M\ t \simeq s \parallel F$$

- Ground terms, not only shared constants, to serve next rule

DPLL($\Gamma+\mathcal{T}$): expansion inferences

- Say that non-ground clauses $C_1, \ldots, C_m$
  and ground $\mathcal{R}$-literals $L_{m+1}, \ldots, L_n$
  generate clause C
  by an expansion inference rule in Γ (e.g., superposition)
- Then if we have $H_1 \triangleright C_1, \ldots, H_m \triangleright C_m$ in F
  and $L_{m+1}, \ldots, L_n$ in M
  we can generate $H_1 \cup \ldots \cup H_m \cup \{L_{m+1}, \ldots, L_n\} \triangleright C$

DPLL($\Gamma+\mathcal{T}$): expansion inferences

- Deduce: given non-ground clauses $\{H_1 \triangleright C_1, \ldots, H_m \triangleright C_m\}$ in F and ground $\mathcal{R}$-literals $\{L_{m+1}, \ldots, L_n\}$ in M

$$M \parallel F \quad \Longrightarrow \quad M \parallel F, H \triangleright C$$

  where $H = H_1 \cup \ldots \cup H_m \cup \{L_{m+1}, \ldots, L_n\}$
  and a Γ-rule infers C from $\{C_1, \ldots, C_m, L_{m+1}, \ldots, L_n\}$
- Only $\mathcal{R}$-literals: Γ-inferences ignore $\mathcal{T}$-literals
- Take ground unit $\mathcal{R}$-clauses from M as PropagateEq puts them there

DPLL($\Gamma+\mathcal{T}$): contraction inferences

- Γ: generate and keep clause; delete redundant clauses; once redundant always redundant
- How to combine this with a system with backjumping, where clauses may disappear not because redundant, but because the hypotheses they depend on are gone from the trail due to backjumping?

DPLL($\Gamma+\mathcal{T}$): contraction inferences

- Single premise (e.g., tautology deletion):
  apply to $H \triangleright C$ if it applies to C
- Multiple premises (e.g., subsumption, simplification):
  prevent situation where clause is deleted, but clauses that make it redundant are gone because of backjumping

Scope level

- Scope level of a literal in M: its decision level:
  $level(L)$ in $M\ L\ M'$: number of decided literals in $M\ L$
- Scope level of a set of literals: the maximum:
  $level(H) = max\{level(L) \mid L \in H\}$ and 0 for $\emptyset$

DPLL($\Gamma+\mathcal{T}$): contraction inferences

- Say we have non-ground clauses $H \triangleright C$, $H_2 \triangleright C_2, \ldots, H_m \triangleright C_m$ in F and ground $\mathcal{R}$-literals $L_{m+1}, \ldots, L_n$ in M
- $C_2, \ldots, C_m, L_{m+1}, \ldots, L_n$ simplify C to C' or subsume it
- Let $H' = H_2 \cup \ldots \cup H_m \cup \{L_{m+1}, \ldots, L_n\}$
- Simplification: replace $H \triangleright C$ by $(H \cup H') \triangleright C'$
- Subsumption: delete $H \triangleright C$
- Both: if $\mathit{level}(H') \leq \mathit{level}(H)$: delete
  if $\mathit{level}(H') > \mathit{level}(H)$: disable
  (re-enable when backjumping $\mathit{level}(H')$)

DPLL($\Gamma+\mathcal{T}$): DPLL-CDCL rules

- Decide: guess ground L true, add it to M (decided literal)

$$M \parallel F \quad \Longrightarrow \quad M\ L \parallel F$$

- UnitPropagate consequence of assignment (implied literal):
  $C \vee L$ ground clause
  if $M \models_P \neg C$ (all lits in C false)

$$M \parallel F, H \triangleright (C \vee L) \quad \Longrightarrow \quad M\ L_{H \triangleright (C \vee L)} \parallel F, H \triangleright (C \vee L)$$

Literals in H are immaterial here because they come from M

DPLL($\Gamma+\mathcal{T}$): DPLL-CDCL rules

- Conflict: C ground clause
  if $M \models_P \neg C$

$$M \parallel F, H \triangleright C \quad \Longrightarrow \quad M \parallel F, H \triangleright C \parallel \neg H \vee C$$

Conflict clauses are ground

DPLL($\Gamma+\mathcal{T}$): DPLL-CDCL rules

- Explain: unfold by resolution implied literal: if $L_{H \triangleright (D \vee L)} \in M$

$$M \parallel F \parallel C \vee \neg L \quad \Longrightarrow \quad M \parallel F \parallel \neg H \vee D \vee C$$

- Learn conflict clause $C \notin clauses(F)$

$$M \parallel F \parallel C \quad \Longrightarrow \quad M \parallel F, C \parallel C$$

DPLL($\Gamma+\mathcal{T}$): DPLL-CDCL rules

- Backjump:

$$M\ L'\ M' \parallel F \parallel C \vee L \quad \Longrightarrow \quad M\ L_{C \vee L} \parallel F'$$

where L' is the least recently decided literal such that
$M \models_P \neg C$ and L undefined in M
F' is F minus clauses whose hypothesis intersects $L'\ M'$

- Unsat: conflict clause is $\Box$

$$M \parallel F \parallel \Box \quad \Longrightarrow \quad unsat$$

DPLL($\Gamma+\mathcal{T}$): DPLL($\mathcal{T}$) rules

- $\mathcal{T}$-Propagate: add ground L that is $\mathcal{T}$-consequence of M: if $L_1, \ldots, L_n \in M$ and $L_1, \ldots, L_n \models_{\mathcal{T}} L$

$$M \parallel F \quad \Longrightarrow \quad M \; L_{(\neg L_1 \vee \ldots \vee \neg L_n \vee L)} \parallel F$$

- $\mathcal{T}$-Conflict: detect that $L_1, \ldots, L_n$ in M are $\mathcal{T}$-inconsistent: if $L_1, \ldots, L_n \in M$ and $L_1, \ldots, L_n \models_{\mathcal{T}} \perp$

$$M \parallel F \quad \Longrightarrow \quad M \parallel F \parallel \neg L_1 \vee \ldots \vee \neg L_n$$

DPLL($\Gamma+\mathcal{T}$): Summary

Use each engine for what is best at:

- DPLL($\mathcal{T}$) works on ground clauses and built-in theories
- Γ works on non-ground clauses and ground unit $\mathcal{R}$-clauses taken from M
- Γ works on $\mathcal{R}$-satisfiability problem
- Γ seen as $\mathcal{R}$-solver in a Nelson-Oppen combination
- Γ-inferences guided by current partial model

Issues about completeness

- Γ is refutationally complete
  Since Γ does not see all the clauses, DPLL($\Gamma+\mathcal{T}$) does not inherit refutational completeness trivially
- Equality sharing is complete for Nelson-Oppen built-in theories: how to extend to a combination with an axiomatized theory $\mathcal{R}$?
- DPLL($\mathcal{T}$) uses depth-first search: complete for ground SMT problems, not with non-ground inferences

From rewriting-based theorem proving

- N: set of ground clauses
- I_N: candidate model
- Counterexample: $I_N \not\models C$
- Reduction property for counterexamples: for all N, I_N, and counterexample $C \in N$, Γ infers a counterexample $D \prec C$
- Theorem: if N Γ-saturated, then unsatisfiable iff $\Box \in N$
- Proof: show that if $\Box \notin N$ then satisfiable

From rewriting-based theorem proving

- Proof: show that if $\Box \notin N$ then satisfiable
  BWOC: Assume that it is not
  For all candidate model I_N there is a counterexample $C \in N$
  Let C be the $\prec$-smallest
  By the reduction property for counterexamples, Γ can generate a counterexample $D \prec C$
  Either $D \in N$ and then C is not the smallest
  Or $D \notin N$ and then N is not Γ-saturated
  Either way we have a contradiction

Γ as decision procedure

- Termination results by analysis of inferences: Γ as an $\mathcal{R}$-satisfiability procedure
- Covered theories include: lists, arrays and records with or without extensionality, recursive data structures

Joint works with Alessandro Armando, Mnacho Echenim, Michaël Rusinowitch, Silvio Ranise, and Stephan Schulz

Variable-inactivity

- Clause C: variable-inactive if no maximal literal has the form $t \simeq x$ where $x \notin Var(t)$
  (Intuition: no paramodulation/superposition from variables
  the case $x \in Var(t)$ is blocked by the ordering as $t[x] \succ x$ by the subterm property)
- Set of clauses: variable-inactive if all its clauses are

[Alessandro Armando, Maria Paola Bonacina, Silvio Ranise, Stephan Schulz, FroCoS 2005, ACM TOCL 2009]

Variable-inactivity

- $S_0 = \mathcal{R} \cup S$ where S is any set of ground $\mathcal{R}$-literals
- Γ-derivation: $S_0 \vdash S_1 \vdash \ldots S_i \vdash S_{i+1}$
- Fairness of Γ: no irredundant Γ-inference indefinitely postponed
- Limit: $S_\infty = \bigcup_{j\geq 0} \bigcap_{i\geq j} S_i$ (persistent clauses)
- Theory $\mathcal{R}$: variable-inactive if limit S_∞ of fair Γ-derivation from $S_0 = \mathcal{R} \cup S$ is variable-inactive
- Persistent clauses are variable-inactive

[Alessandro Armando, Maria Paola Bonacina, Silvio Ranise, Stephan Schulz, FroCoS 2005, ACM TOCL 2009]

Modularity of termination

- Theorem: if Γ terminates on $\mathcal{R}_i$-satisfiability problems, it terminates also on $\mathcal{R}$-satisfiability problems for $\mathcal{R} = \bigcup_{i=1}^{n} \mathcal{R}_i$, if the $\mathcal{R}_i$'s are disjoint and variable-inactive
- Proof: (assume $t \succ c$ for all compound term t and constant c) the only inferences across theories are superpositions/paramodulations from shared constants replacing constant with constant: only finitely many (informally: correspond to equalities between shared constants in equality sharing)

[Alessandro Armando, Maria Paola Bonacina, Silvio Ranise, Stephan Schulz, FroCoS 2005 and ACM TOCL 2009]

Variable inactivity implies stable infiniteness

- Lemma: if S_0 is satisfiable, it admits no infinite model iff S_∞ contains a cardinality constraint
  (e.g., $x \simeq y \vee x \simeq z \vee z \simeq y$: not variable-inactive)
- Theorem: if $\mathcal{R}$ is variable-inactive, then it is stably infinite
  Proof: by the lemma, not stably infinite implies not variable-inactive
- In practice Γ reveals lack of infinite model by generating a cardinality constraint

[Maria Paola Bonacina, Silvio Ghilardi, Enrica Nicolini, Silvio Ranise, and Daniele Zucchelli, IJCAR 2006]

Requirements for DPLL($\Gamma+\mathcal{T}$): $\mathcal{T}$-smooth set

$\mathcal{R} \cup P$ is $\mathcal{T}$-smooth, for $\mathcal{T} = \bigcup_{i=1}^{n} \mathcal{T}_i$, if

- $\mathcal{T}_1, \ldots, \mathcal{T}_n$ and $\mathcal{R}$ are disjoint
- $\mathcal{T}_1, \ldots, \mathcal{T}_n$ are stably infinite
- $\mathcal{R}$ is variable-inactive
- P is $P_1 \cup P_2$
  - P_1: ground $\mathcal{R}$-clauses
  - P_2: ground $\mathcal{T}$-clauses

Fairness for DPLL($\Gamma+\mathcal{T}$)

- Γ-based transitions: Deduce transitions and contraction transitions
- Fairness: all applicable transitions applied eventually except redundant Γ-based transitions
- Saturated state:
  - Either $M \parallel F \parallel \Box$
  - Or $M \parallel F$ such that the only applicable inferences are redundant Γ-based transitions
- Fair derivation yields saturated state eventually

Refutational completeness of DPLL($\Gamma+\mathcal{T}$)

- Theorem: if input $S = \mathcal{R} \cup P$ is $\mathcal{T}$-smooth, whenever DPLL($\Gamma+\mathcal{T}$) reaches a saturated state $M \parallel F$, S is $\mathcal{T}$-satisfiable.
- Proof: we need to show that *clauses*$(F) \cup M$ is $\mathcal{T}$-satisfiable
  - For each ground non-unit clause C in *clauses*(F) there is a literal of C in M by saturation w.r.t. Decide: ground non-unit clause are redundant in *clauses*$(F) \cup M$
  - Thus, the fact that Γ does not see ground non-unit $\mathcal{R}$-clauses is immaterial, because they are satisfied by M

Refutational completeness of DPLL($\Gamma+\mathcal{T}$)

Proof: (continues)

- Non-ground $\mathcal{R}$-clauses in $clauses(F)$ and ground $\mathcal{R}$-literals in M: Γ-saturated, hence satisfiable by the reduction property for counterexamples
- All $\mathcal{T}$-clauses: $\mathcal{T}$-satisfiable by saturation w.r.t. $\mathcal{T}$-conflict
- Combination: by completeness of a Nelson-Oppen combination of stably infinite theories by $\mathcal{T}$-smoothness

How to ensure fairness of DPLL($\Gamma+\mathcal{T}$)?

Example:

1. $\neg p(x, y) \vee p(f(x), f(y)) \vee p(g(x), g(y))$: seen by Γ
2. $p(a, b)$
3. $g(x) \not\simeq x$: seen by Γ
4. $g(c) \simeq c \vee g(d) \simeq d$

Unsatisfiable because of clauses (3) and (4).
Initially Γ sees only clauses (1) and (3) because M is empty.

Example continued

1. $\neg p(x,y) \vee p(f(x),f(y)) \vee p(g(x),g(y))$: seen by Γ
2. $p(a,b)$
3. $g(x) \not\simeq x$: seen by Γ
4. $g(c) \simeq c \vee g(d) \simeq d$

1. Decide adds $p(a,b)$ to M: seen by Γ
2. Resolution generates $p(f(a),f(b)) \vee p(g(a),g(b))$
3. Decide adds $p(f(a),f(b))$ to M: seen by Γ
4. Resolution generates
   $p(f(f(a)),f(f(b))) \vee p(g(f(a)),g(f(b)))$ ...
5. ... infinite unfair derivation that does not detect unsat!

Answer: iterative deepening

Inference depth:

- Clause: $\mathit{infDepth}(C) =$ depth of inference tree producing C
- Implied literal: $\mathit{infDepth}(L) =$ depth of clause that implied L
- Decided literal: $\mathit{infDepth}(L) =$ min inference depth of clause including L

k-bounded DPLL($\Gamma+\mathcal{T}$): Deduce restricted to premises C with $\mathit{infDepth}(C) < k$

Same example with iterative deepening

1. $\neg p(x, y) \vee p(f(x), f(y)) \vee p(g(x), g(y))$: seen by Γ
2. $p(a, b)$
3. $g(x) \not\simeq x$: seen by Γ
4. $g(c) \simeq c \vee g(d) \simeq d$

1. The bound on inference depth prevents the infinite alternation of Decide and Resolution steps
2. Decide adds $g(c) \simeq c$ to M: seen by Γ
3. Resolution generates $\square$
4. Decide adds $g(d) \simeq d$ to M: seen by Γ
5. Resolution generates $\square$
6. Unsat

Termination

- Theorem: k-bounded DPLL($\Gamma+\mathcal{T}$) terminates:
  DPLL($\mathcal{T}$) does + finitely many Deduce steps within k
- DPLL($\Gamma+\mathcal{T}$) stuck at k if only Deduce applies and only to premises excluded by bound k
- Three outcomes: sat, unsat, stuck (don't know)
- Decision procedure: sat, unsat

How to get decision procedures?

- Need theorem prover that terminates on satisfiable inputs
- Not possible in general:
  - FOL is only semi-decidable
  - First-order formulæ of linear arithmetic with uninterpreted functions: not even semi-decidable

However we need less than a general solution.

Problematic axioms do occur

Example:

1. $\neg(x \sqsubseteq y) \vee f(x) \sqsubseteq f(y)$ (Monotonicity)
2. $a \sqsubseteq b$ generates by resolution
3. $\{f^i(a) \sqsubseteq f^i(b)\}_{i\geq 0}$

When $f(a) \sqsubseteq f(b)$ or $f^2(a) \sqsubseteq f^2(b)$ often suffice to show satisfiability

The idea of speculative inferences

- Speculative inference: adds arbitrary clause C
- To induce termination on satisfiable inputs
- In order to detect satisfiability it suffices to find one model
- If we can find a model that satisfies both the input set of clauses and those added by speculative inferences, we do not worry that the latter may not be true in all models

Speculative inferences: example

1. $\neg(x \sqsubseteq y) \vee f(x) \sqsubseteq f(y)$
2. $a \sqsubseteq b$
3. $a \sqsubseteq f(c)$
4. $\neg(a \sqsubseteq c)$

Speculative inferences: example

1. $\neg(x \sqsubseteq y) \vee f(x) \sqsubseteq f(y)$
2. $a \sqsubseteq b$
3. $a \sqsubseteq f(c)$
4. $\neg(a \sqsubseteq c)$

1. Add $f(x) \simeq x$
2. Rewrite $a \sqsubseteq f(c)$ into $a \sqsubseteq c$ and get $\Box$: backtrack!

Speculative inferences: example

1. $\neg(x \sqsubseteq y) \vee f(x) \sqsubseteq f(y)$
2. $a \sqsubseteq b$
3. $a \sqsubseteq f(c)$
4. $\neg(a \sqsubseteq c)$

1. Add $f(x) \simeq x$
2. Rewrite $a \sqsubseteq f(c)$ into $a \sqsubseteq c$ and get $\Box$: backtrack!
3. Add $f(f(x)) \simeq x$
4. $a \sqsubseteq b$ yields only $f(a) \sqsubseteq f(b)$
5. $a \sqsubseteq f(c)$ yields only $f(a) \sqsubseteq c$
6. Terminate and detect satisfiability

Speculative inferences in DPLL($\Gamma+\mathcal{T}$)

- Speculative inference: add arbitrary clause C
- What if it makes the problem unsatisfiable?
- Detect conflict and backjump:
  - $\lceil C \rceil$: new propositional variable (a "name" for C)
  - Use hypothetical clauses: Add $\lceil C \rceil \triangleright C$ to F
  - Add $\lceil C \rceil$ to M to memorize this assumption in the trail
  - Speculative inferences are reversible, as the system can remove $\lceil C \rceil$ from M and $\lceil C \rceil \triangleright C$ from F by backjumping

Speculative inferences in DPLL($\Gamma+\mathcal{T}$)

State of derivation: $M \parallel F$

Transition rule:

- SpeculativeIntro: add $\lceil C \rceil \triangleright C$ to F and $\lceil C \rceil$ to M

$$M \parallel F \quad \Longrightarrow \quad M\ \lceil C \rceil \parallel F,\ \lceil C \rceil \triangleright C$$

Speculative inferences in DPLL($\Gamma+\mathcal{T}$)

- Also SpeculativeIntro is bounded by iterative deepening for termination:
  (k, u)-bounded DPLL($\Gamma+\mathcal{T}$)
  with bound k on inference depth for Deduce
  and bound u on number of applications of SpeculativeIntro
- DPLL($\Gamma+\mathcal{T}$) stuck at (k, u) if the only applicable transitions are Deduce beyond k or SpeculativeIntro beyond u

The example again

1. $\neg(x \sqsubseteq y) \vee f(x) \sqsubseteq f(y)$
2. $a \sqsubseteq b$
3. $a \sqsubseteq f(c)$
4. $\neg(a \sqsubseteq c)$

The example again

1. $\neg(x \sqsubseteq y) \vee f(x) \sqsubseteq f(y)$
2. $a \sqsubseteq b$
3. $a \sqsubseteq f(c)$
4. $\neg(a \sqsubseteq c)$

1. Add $\lceil f(x) \simeq x \rceil \triangleright f(x) \simeq x$
2. Rewrite $a \sqsubseteq f(c)$ into $\lceil f(x) \simeq x \rceil \triangleright a \sqsubseteq c$

The example again

1. $\neg(x \sqsubseteq y) \vee f(x) \sqsubseteq f(y)$
2. $a \sqsubseteq b$
3. $a \sqsubseteq f(c)$
4. $\neg(a \sqsubseteq c)$

1. Add $\lceil f(x) \simeq x \rceil \triangleright f(x) \simeq x$
2. Rewrite $a \sqsubseteq f(c)$ into $\lceil f(x) \simeq x \rceil \triangleright a \sqsubseteq c$
3. Generate $\lceil f(x) \simeq x \rceil \triangleright \Box$; Backtrack, learn $\neg\lceil f(x) \simeq x \rceil$

The example again

1. $\neg(x \sqsubseteq y) \vee f(x) \sqsubseteq f(y)$
2. $a \sqsubseteq b$
3. $a \sqsubseteq f(c)$
4. $\neg(a \sqsubseteq c)$

1. Add $\lceil f(x) \simeq x \rceil \triangleright f(x) \simeq x$
2. Rewrite $a \sqsubseteq f(c)$ into $\lceil f(x) \simeq x \rceil \triangleright a \sqsubseteq c$
3. Generate $\lceil f(x) \simeq x \rceil \triangleright \square$; Backtrack, learn $\neg \lceil f(x) \simeq x \rceil$
4. Add $\lceil f(f(x)) \simeq x \rceil \triangleright f(f(x)) \simeq x$
5. $a \sqsubseteq b$ yields only $f(a) \sqsubseteq f(b)$
6. $a \sqsubseteq f(c)$ yields only $\lceil f(f(x)) = x \rceil \triangleright f(a) \sqsubseteq c$
7. Terminate and detect satisfiability

Decision procedures with speculative inferences

To decide satisfiability modulo $\mathcal{T}$ of $\mathcal{R} \cup P$:

- Find sequence of clauses $U = C_1, C_2 \ldots C_i, \ldots$ such that
- If SpeculativeIntro adds the clauses in U there exist k and u s.t. (k, u)-bounded DPLL($\Gamma+\mathcal{T}$) is guaranteed to terminate
  - returning Unsat if $\mathcal{R} \cup P$ is $\mathcal{T}$-unsatisfiable
  - in a state which is not stuck otherwise

Essentially finite theories

A weakening of the finite model property:

- A structure Φ is essentially finite w.r.t. a function symbol f if the range of $\Phi(f)$ is finite
- Theorem: If Φ is essentially finite w.r.t. a monadic function symbol f then $\Phi \models f^j(x) \simeq f^i(x)$ for some $j \neq i$
- Essentially finite $\mathcal{R}$:
  - signature has a single monadic function symbol f
  - whenever $\mathcal{R} \cup P$ is satisfiable, for P a set of ground $\mathcal{R}$-clauses, it has an essentially finite model w.r.t. f

Decision procedures for essentially finite theories

Theorem:

- $\mathcal{R}$ is essentially finite
- SpeculativeIntro adds $f^j(x) \simeq f^i(x)$, $j > i$, for increasing values of i and j
- If the number of literals in clauses is bounded by other properties of Γ and $\mathcal{R}$
- Then DPLL($\Gamma+\mathcal{T}$) is a decision procedure for $\mathcal{T}$-satisfiability of $\mathcal{R}$-smooth problems $\mathcal{R} \cup P$

Decision procedures for essentially finite theories

Proof:

- $\mathcal{R} \cup P$ $\mathcal{T}$-unsatisfiable: by refutational completeness DPLL(Γ+$\mathcal{T}$) reaches state unsat when the bound k on inference depth gets large enough
- $\mathcal{R} \cup P$ $\mathcal{T}$-satisfiable:
  - Bound u on SpeculativeIntro large enough to add $f^j(x) \simeq f^i(x)$ true in the model ($j > i$)
  - Rewriting by $f^j(x) \simeq f^i(x)$ limits term depth
  - Number of literals limited by hypothesis
  - Only finitely many clauses generated
  - Termination without getting stuck

Negative selection

A way to restrict Resolution and Paramodulation/Superposition:

- A clause can have one, some or all its negative literal selected depending on the chosen selection function
- The selection function is part of the search plan
- The negative literal resolved upon and the literal paramodulated/superposed into do not need to be maximal, must be selected instead
- The other premise must not contain any selected literal

Negative selection

- Some negative literal is selected for each clause containing one
- Then one premise for each Resolution and Paramodulation/Superposition inference will be positive: Positive Strategy
- If in addition the problem is Horn: (Positive) Unit Strategy
- Resolution with negative selection realizes (Positive) Hyperresolution

A situation where clause length is limited

Γ: Resolution and Paramodulation/Superposition with negative selection, Simplification

- $\mathcal{R}$ is Horn
- (Positive) Unit Strategy
- Unit Paramodulation/Superposition does not increase the number of literals
- Hyperresolution only generates positive unit clauses
- The number of literals in generated clauses is bounded

Ground-preserving clauses

- A clause is ground-preserving if variables in positive literals appear also in negative literals
- A set of clauses is ground-preserving if all its clauses are
- In a ground-preserving set the only positive clauses are ground

Another terminating situation

Γ: Resolution and Paramodulation/Superposition with negative selection, Simplification

- $\mathcal{R}$ is ground-preserving
- Positive Strategy
- Hyperresolution only generates positive ground clauses
- Paramodulation/Superposition generates either ground clauses or non-ground ground-preserving clauses with fewer variable positions than the non-ground parent
- Simplification by $f^j(x) \simeq f^i(x)$ limits term depth
- Only finitely many clauses generated

Axiomatizations of type systems

$\sqsubseteq$: subtype relation, f: type constructor

Reflexivity	$x \sqsubseteq x$	(1)
Transitivity	$\neg(x \sqsubseteq y) \vee \neg(y \sqsubseteq z) \vee x \sqsubseteq z$	(2)
Anti-Symmetry	$\neg(x \sqsubseteq y) \vee \neg(y \sqsubseteq x) \vee x \simeq y$	(3)
Monotonicity	$\neg(x \sqsubseteq y) \vee f(x) \sqsubseteq f(y)$	(4)
Tree-Property	$\neg(z \sqsubseteq x) \vee \neg(z \sqsubseteq y) \vee x \sqsubseteq y \vee y \sqsubseteq x$	(5)

Multiple inheritance: $\mathrm{MI} = \{(1), (2), (3), (4)\}$
Single inheritance: $\mathrm{SI} = \mathrm{MI} \cup \{(5)\}$

These axiomatizations are essentially finite

- $\mathcal{R}$ has the finite model property: whenever $\mathcal{R} \cup P$ is satisfiable, for P a set of ground $\mathcal{R}$-clauses, it has a finite model
- Theorems: SI and MI have the finite model property and therefore they are essentially finite

Concrete examples of decision procedures

DPLL($\Gamma+\mathcal{T}$) with addition of $f^j(x) \simeq f^i(x)$ for $j > i$ decides the satisfiability modulo $\mathcal{T}$ of $\mathcal{T}$-smooth problems

- $\mathrm{MI} \cup P$
  because MI is essentially finite and Horn
- $\mathrm{SI} \cup P$
  because SI is essentially finite and ground-preserving
  (except for reflexivity which however does not affect termination by case analysis of the possible inferences)

More axioms for types

g: type representative

- $g(x) \not\simeq null$
- $h(g(x)) \simeq x$

Let $\mathrm{TR} = \{g(x) \not\simeq null,\ h(g(x)) \simeq x\}$
TR has only infinite models:

- g is injective, since it has left inverse
- g is not surjective, since there is no pre-image for *null*
- a set with an injective but not surjective function is infinite

A decision procedure for more than one function symbol

Theorem: DPLL($\Gamma+\mathcal{T}$) with addition of $f^j(x) \simeq f^i(x)$ for $j > i$ decides the satisfiability modulo $\mathcal{T}$ of $\mathcal{T}$-smooth problems $\mathrm{MI} \cup \mathrm{TR} \cup P$ and $\mathrm{SI} \cup \mathrm{TR} \cup P$.

A decision procedure for more than one function symbol

Proof:

- Γ terminates on TR-satisfiability problems by case analysis of the possible inferences
- MI and TR are disjoint and variable-inactive
- SI and TR are disjoint and variable-inactive
- Γ terminates on MI $\cup$ TR-satisfiability problems and SI $\cup$ TR-satisfiability problems
- Thus the addition of TR does not affect the previous results

Future work

- More decision procedures by speculative inferences?
- DPLL($\Gamma+\mathcal{T}$) detects the lack of infinite models if Γ generates a cardinality constraint, but does not have a general way to discover the lack of finite models (works on asymmetric combinations and superposition for bounded domains?)
- MCsat(Γ)?

Selected references

- M. P. Bonacina, C. A. Lynch and L. de Moura. On deciding satisfiability by theorem proving with speculative inferences. *Journal of Automated Reasoning*, 47(2):161–189, August 2011.
- A. Armando, M. P. Bonacina, S. Ranise and S. Schulz. New results on rewrite-based satisfiability procedures. *ACM Transactions on Computational Logic*, 10(1):129–179, January 2009.
- M. P. Bonacina and M. Echenim. On variable-inactivity and polynomial $\mathcal{T}$-satisfiability procedures. *Journal of Logic and Computation*, 18(1):77–96, February 2008.
- M. P. Bonacina, S. Ghilardi, E. Nicolini, S. Ranise and D. Zucchelli. Decidability and undecidability results for Nelson-Oppen and rewrite-based decision procedures. *Proc. of the 3rd IJCAR*, Springer, LNAI 4130, 513–527, 2006.