Reasoning with speculative inferences[1]

Maria Paola Bonacina

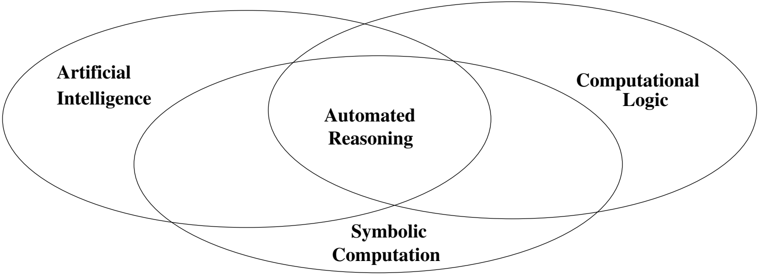
Visiting: Computer Science Laboratory, SRI International, Menlo Park, CA, USA
Affiliation: Dipartimento di Informatica, Università degli Studi di Verona, Verona, Italy, EU

Talk given at Nuance Communications
Sunnyvale, California, USA

September 2016

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

Automated reasoning



- Automated or symbolic reasoning:
- Logico-deductive, probabilistic ...

Theorem proving

- Assumptions: H
- Conjecture: φ
- Problem: $H \models^? \varphi$
  Refutation: is $H \cup \{\neg\varphi\}$ unsatisfiable?
- $H \cup \{\neg\varphi\} \leadsto S$ set of clauses
- Yes, with proof $S \vdash \bot$
  $\neg\varphi$ unsatisfiable in H, φ valid in H
- No, with model of S, counter-example for φ
  $\neg\varphi$ satisfiable in H, φ invalid in H

Model building/Constraint solving

- Assumptions: H
- Constraint: φ
- Problem: is there a model/solution of $H \cup \{\varphi\}$?
- $H \cup \{\varphi\} \leadsto S$ set of clauses
- Yes, with model of S
  φ satisfiable in H, $\neg\varphi$ invalid in H
- No, with proof $S \vdash \bot$
  φ unsatisfiable in H, $\neg\varphi$ valid in H

Two sides of the same coin

- Theorem proving and model building/constraint solving
- Proofs and models
- Are two sides of the same coin
- Both involve both inference and search:
  - Inference towards a proof, search as deciding which inference
  - Search towards a model, inference to repair a conflict between candidate model and the set S to be satisfied

Sample applications

- Verification: a program state is a model, proof of verification conditions
- Testing: models as "moles" in automated test generation
- Synthesis: proof of synthesis conditions, models as examples in example-driven synthesis
- Reasoning support to model checkers (e.g., abstraction refinement), static analyzers (e.g., invariant generation)
- Reasoning as a back-end enabling technology

Expressivity: Formulating the problem

- Propositional logic: P, $\neg Q$, $P \vee Q$, $\neg P \wedge Q$
- First-order logic: $P(a)$, $\neg R(x,x) \vee R(x,f(x))$
- Free symbols: P, Q, a, R, f
- Equality: $x \simeq y \supset f(x) \simeq f(y)$ (defined symbol)
- Arithmetic: $2 \leq a \wedge a \leq 3$, $a \simeq 2 \vee a \simeq 3$
- Data structures: $car(cons(x,y)) \simeq x$, $select(store(x,z,v),z) \simeq v$ (defined symbols)
- Quantifiers: invariants, theory axioms: $\forall x,z,v.\ select(store(x,z,v),z) \simeq v$

Decidability

A procedure that

- Takes the input set of clauses S and returns
  - Either satisfiable with a model
  - Or unsatisfiable with a proof
- Is a decision procedure for satisfiability/validity
- Decision procedures do exist (e.g., propositional logic, fragments of arithmetic, quantifier-free fragments of first-order theories)

Expressivity vs. decidability

- First-order logic: unsatisfiability (hence validity) is only semi-decidable; satisfiability is not even semi-decidable
- First-order formulæ of linear arithmetic with free function symbols: not even unsatisfiability is semi-decidable
- We cannot have decision procedures for all problems in a highly expressive language
- In practice we often need less than a general solution!

Speculative inferences

Symbolic reasoning applied to mathematics

- Most conjectures are true: we expect a proof $S \vdash \perp$
- Sacrifice completeness: for some unsatisfiable inputs we won't find a refutation
- In favor of efficiency: find faster the proofs we find
- Retain soundness: if a proof is found, input S is indeed unsatisfiable

Example: Deletion by weight

- Associate weights to symbols
- If all weigh 1 same as symbol count
- Delete clauses of weight larger than k (heuristic threshold)
- Not complete, still sound, faster

Symbolic reasoning applied to verification

- Most conjectures are false (bugs, wrong specs): we expect a model of input S
- Imperil soundness: if a proof were found, S may not be unsatisfiable
- In favor of termination when S is satisfiable
- Retain completeness: if proof not found, S is indeed satisfiable

Speculative inferences

- Add to the problem S an arbitrary clause C
- Not a logical consequence of S: not sound

Speculative inferences: example

- Suppose we want to suppress the literals D in $C \vee D$
- Add C
- Subsume $C \vee D$

Speculative inferences: example

- Suppose a clause $C[t]$ contains a deep term t
- Add $t \simeq a$ where a is a constant
- Simplify $C[t]$ to $C[a]$

Speculative inferences for model building

- We expect S to have a model
- But the reasoner may not terminate
- We add C
- If $S \cup \{C\}$ has a model, it is also a model of S
- If adding C enforces termination, we find a model (we only need one!)
- We may need to try more than one C, preferably a few

Example

Let $\sqsubseteq$ be a subtype relation and f a type constructor

- Transitivity:
  $\neg(x \sqsubseteq y) \vee \neg(y \sqsubseteq z) \vee x \sqsubseteq z$
- Monotonicity:
  $\neg(x \sqsubseteq y) \vee f(x) \sqsubseteq f(y)$

Example

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

In practice $f(a) \sqsubseteq f(b)$ or $f^2(a) \sqsubseteq f^2(b)$ often suffice to show satisfiability of the input set

Example

- $\neg(x \sqsubseteq y) \vee f(x) \sqsubseteq f(y)$
- $a \sqsubseteq b$
- $a \sqsubseteq f(c)$
- $\neg(a \sqsubseteq c)$

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

How to avoid unsoundness

- Adding C may make unsatisfiable a problem that was satisfiable
- Detect it as a conflict between the candidate model we are building and the set of clauses
- Recover by backtracking
- The overall derivation is still sound!
- Requires a model-based reasoner, that operates by building a candidate model

Model-based reasoning: DPLL($\Gamma+\mathcal{T}$)

Shape of the input problem

- Background theory $\mathcal{T}$
  - $\mathcal{T} = \bigcup_{i=1}^{n} \mathcal{T}_i$ (e.g., linear arithmetic)
- Set of formulas: $\mathcal{R} \cup P$
  - $\mathcal{R}$: set of non-ground clauses without $\mathcal{T}$-symbols (e.g., invariant, frame condition, axioms of another theory)
  - P: set of ground clauses with $\mathcal{T}$-symbols
- Is $\mathcal{R} \cup P$ satisfiable in $\mathcal{T}$?

What do we need

- DPLL (Davis-Putnam-Logemann-Loveland) procedure with CDCL (Conflict-Driven Clause Learning) for SAT
- $\mathcal{T}_i$-solvers: decision procedures for the $\mathcal{T}_i$'s
- Equality sharing (Nelson-Oppen) combination of the $\mathcal{T}_i$-solvers to yield a $\mathcal{T}$-solver
- DPLL($\mathcal{T}$): DPLL-CDCL with $\mathcal{T}$-solver built-in
- First-order engine Γ to handle $\mathcal{R}$: resolution, subsumption, paramodulation, superposition, simplification

A taste of DPLL

$S \supseteq \{\neg a \vee b,\ \neg c \vee d,\ \neg e \vee \neg f,\ f \vee \neg e \vee \neg b\}$

1. Decide: a is true; Propagate: b must be true
2. Decide: c is true; Propagate: d must be true
3. Decide: e is true; Propagate: $\neg f$ must be true

- Conflict: $f \vee \neg e \vee \neg b$ is all false
- Backtrack: undo $\neg f$ and set $\neg e$ true
- Continue until it finds a satisfying assignment (model) or none can be found (backtrack to level 0)

A taste of CDCL

$S \supseteq \{\neg a \vee b,\ \neg c \vee d,\ \neg e \vee \neg f,\ f \vee \neg e \vee \neg b\}$
$M = a\ b\ c\ d\ e\ \neg f$

1. Conflict: $f \vee \neg e \vee \neg b$
2. Explain by resolving $f \vee \neg e \vee \neg b$ with $\neg e \vee \neg f$: $\neg e \vee \neg b$
3. Learn $\neg e \vee \neg b$: no model with e and b true
4. Backjump to earliest state with $\neg b$ false and $\neg e$ unassigned: $M = a\ b\ \neg e$

From now on: DPLL stands for DPLL-CDCL

A taste of DPLL($\mathcal{T}$)

Let $\mathcal{T}$ be the quantifier-free fragment of the theory of equality

$\mathcal{T}$-literals replaced by proxy propositional atoms (abstraction)

$M = a\ b\ c\ d\ e\ \neg f$

- $\mathcal{T}$-Propagate: if a stands for $p \simeq q$ and c stands for $q \simeq r$ add to M a propositional variable that stands for $p \simeq r$
- $\mathcal{T}$-Conflict: if a stands for $p \simeq q$ and $\neg f$ stands for $g(p) \not\simeq g(q)$ detect a theory conflict

A taste of Γ

$$S \supseteq \{f(x) \simeq g(a,x),\ P(f(b)),\ \neg P(g(y,b))\}$$

$\succ$: recursive path ordering based on precedence $f > g > a$

1. Simplification:
   $f(x) \simeq g(a,x)$ simplifies $P(f(b))$ into $P(g(a,b))$ with matching substitution $\sigma = \{x \leftarrow b\}$ and because $f(b) \succ g(a,b)$
2. Resolution:
   $P(g(a,b))$ and $\neg P(g(y,b))$ resolve with most general unifier $\sigma = \{y \leftarrow a\}$ to yield $\Box$

A taste of Γ

$$S \supseteq \{f(z,e) \simeq z,\ f(I(x,y),y) \simeq x\}$$

$\succ$: any simplification ordering (subterm property)

- Superposition:
  $f(z,e) \simeq z$ superposes into $f(I(x,y),y) \simeq x$ with most general unifier $\sigma = \{z \leftarrow I(x,e),\ y \leftarrow e\}$ and because $f(I(x,e),e) \succ I(x,e)$ and $f(I(x,e),e) \succ x$ it yields $I(x,e) \simeq x$
- Expansion: e.g., Resolution, Factoring, Superposition, Paramodulation
- Contraction: e.g., Simplification, Subsumption, Tautology deletion

How to combine the strengths of these reasoning engines?

- DPLL-CDCL: SAT-problems; large non-Horn clauses
- Theory solvers: e.g., ground equality, linear arithmetic
- DPLL($\mathcal{T}$)-based SMT-solver: efficient, scalable, integrated theory reasoning
- Superposition-based inference system Γ: equality, Horn clauses, universally quantified variables

Division of labor

Recall the assumption that input non-ground clauses do not contain $\mathcal{T}$-symbols

Use each reasoning engine for what is best at:

- Non-ground clauses: seen only by Γ
- Ground non-unit clauses: seen only by DPLL($\mathcal{T}$)
- Ground unit clauses: seen by both

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

- Let ground literals from the candidate model M built by DPLL($\mathcal{T}$) be available as premises of Γ-inferences
- Model-driven Γ-inferences
- Stored as hypotheses in inferred clause
- Hypothetical clause: $H \triangleright C$ (equivalent to $\neg H \vee C$)
- Inferred clauses inherit hypotheses from premises

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

State of derivation: $M \,\|\, S$

- Expansion: take as premises non-ground clauses from S and $\mathcal{R}$-literals (unit clauses) from M and add result to S
- Remove hypothetical clauses depending on literals removed from M upon Backjump
- Contraction: as above + scope level to prevent situation where clause is deleted, but clauses that make it redundant are gone because of backjumping

Example

- Reconsider simplifying $P(f(b))$ into $P(g(a,b))$ by $f(x) \simeq g(a,x)$
- Say we have $a \simeq b \triangleright f(x) \simeq g(a,x)$ and $f(f(b)) \simeq f(b) \triangleright P(f(b))$
- Where $a \simeq b$ comes from level k in M and $f(f(b)) \simeq f(b)$ comes from level q in M
- $k \leq q$: delete $f(f(b)) \simeq f(b) \triangleright P(f(b))$ and replace it with $a \simeq b, f(f(b)) \simeq f(b) \triangleright P(g(a,b))$
- $k > q$: disable $f(f(b)) \simeq f(b) \triangleright P(f(b))$ and add $a \simeq b, f(f(b)) \simeq f(b) \triangleright P(g(a,b))$
- Backjump to a level smaller than k: re-enable $f(f(b)) \simeq f(b) \triangleright P(f(b))$ in place of $a \simeq b, f(f(b)) \simeq f(b) \triangleright P(g(a,b))$

DPLL($\Gamma+\mathcal{T}$) as a transition system

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

- Search mode: State of derivation $M \parallel S$
  - M sequence of assigned ground literals: partial model
  - S set of hypothetical clauses
- Conflict resolution mode: State of derivation $M \parallel S \parallel C$
  - C ground conflict clause (including $\neg H \vee C$)
  - CDCL applies

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

- Refutational completeness of the inference system:
  - From that of Γ, DPLL($\mathcal{T}$) and equality sharing
  - Under suitable hypotheses (e.g., disjoint theories)
- Fairness of the search plan:
  - Depth-first search fair only for ground problems
  - Add iterative deepening on inference depth

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

- Speculative inference: add arbitrary clause C
- To induce termination on satisfiable input
- What if it makes problem unsatisfiable?!
- Detect conflict and backjump:
  - Keep track by adding $\lceil C \rceil \triangleright C$
  - $\lceil C \rceil$: new propositional variable (a "name" for C)
  - Speculative inferences are reversible

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

State of derivation: $M \parallel S$

Inference rule:

- SpeculativeIntro: add $\lceil C \rceil \triangleright C$ to S and $\lceil C \rceil$ to M
- Also bounded by iterative deepening

Example as done by system

- $\neg(x \sqsubseteq y) \vee f(x) \sqsubseteq f(y)$
- $a \sqsubseteq b$
- $a \sqsubseteq f(c)$
- $\neg(a \sqsubseteq c)$

Example as done by system

- $\neg(x \sqsubseteq y) \vee f(x) \sqsubseteq f(y)$
- $a \sqsubseteq b$
- $a \sqsubseteq f(c)$
- $\neg(a \sqsubseteq c)$

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

Example as done by system

- $\neg(x \sqsubseteq y) \vee f(x) \sqsubseteq f(y)$
- $a \sqsubseteq b$
- $a \sqsubseteq f(c)$
- $\neg(a \sqsubseteq c)$

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 $\neg\lceil f(x) \simeq x \rceil$

Example as done by system

- $\neg(x \sqsubseteq y) \vee f(x) \sqsubseteq f(y)$
- $a \sqsubseteq b$
- $a \sqsubseteq f(c)$
- $\neg(a \sqsubseteq c)$

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 $\neg \lceil f(x) \simeq x \rceil$
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 "speculative axioms" U
- Show that there exists k s.t. k-bounded DPLL($\Gamma+\mathcal{T}$) is guaranteed to terminate
  - With Unsatisfiable if $\mathcal{R} \cup P$ is unsatisfiable in $\mathcal{T}$
  - In a state which is not stuck at k if $\mathcal{R} \cup P$ is satisfiable in $\mathcal{T}$
- Being stuck at k means halting not because done, but by hitting the limit k in iterative deepening

Decision procedures: essentially finite theories

- $\mathcal{R}$ has single monadic function symbol f
- Essentially finite: if $\mathcal{R} \cup P$ is satisfiable, has model where range of f is finite
- Such a model satisfies $f^j(x) \simeq f^k(x)$ for some $j \neq k$
- SpeculativeIntro adds "pseudo-axioms" $f^j(x) \simeq f^k(x)$, $j > k$
- Use $f^j(x) \simeq f^k(x)$ as rewrite rule to limit term depth
- Clause length limited by other properties of Γ and $\mathcal{R}$
- Only finitely many clauses generated: termination guaranteed without getting stuck

Situations where clause length is limited

Γ: Superposition, Resolution + negative selection, Simplification

Negative selection: only positive literals in positive clauses are active

- $\mathcal{R}$ is Horn
- $\mathcal{R}$ is ground-preserving: variables in positive literals appear also in negative literals;
  the only positive clauses are ground

Axiomatizations of type systems

$$\begin{array}{rll}
\text{Reflexivity} & x \sqsubseteq x & (1) \\
\text{Transitivity} & \neg(x \sqsubseteq y) \vee \neg(y \sqsubseteq z) \vee x \sqsubseteq z & (2) \\
\text{Anti-Symmetry} & \neg(x \sqsubseteq y) \vee \neg(y \sqsubseteq x) \vee x \simeq y & (3) \\
\text{Monotonicity} & \neg(x \sqsubseteq y) \vee f(x) \sqsubseteq f(y) & (4) \\
\text{Tree-Property} & \neg(z \sqsubseteq x) \vee \neg(z \sqsubseteq y) \vee x \sqsubseteq y \vee y \sqsubseteq x & (5)
\end{array}$$

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

Concrete examples of decision procedures

DPLL($\Gamma+\mathcal{T}$) with SpeculativeIntro adding $f^j(x) \simeq f^k(x)$ for $j > k$ decides the satisfiability modulo $\mathcal{T}$ of problems

- $\mathrm{MI} \cup P$
- $\mathrm{SI} \cup P$
- $\mathrm{MI} \cup \mathrm{TR} \cup P$ and $\mathrm{SI} \cup \mathrm{TR} \cup P$

where $\mathrm{TR} = \{\neg(g(x) \simeq \mathit{null}),\ h(g(x)) \simeq x\}$

Thanks

Thanks to all my co-authors

and

Thank you!