Proofs in Conflict-Driven Theory Combination[1]

Presentation Only[2]

Maria Paola Bonacina

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

16th Int. Workshop on Satisfiability Modulo Theories (SMT), satellite of the 9th Int. Joint Conf. on Automated Reasoning (IJCAR) during the 7th Federated Logic Conference (FLoC), Oxford, England, UK

13 July 2018

[1] Joint work with **Stéphane Graham-Lengrand** and **Natarajan Shankar**

[2] Paper in Proc. 7th ACM SIGPLAN Int. Conf. on **Certified Programs and Proofs** (**CPP**), LA, CA, USA, Jan. 2018

Conflict-driven reasoning

- Build candidate model
- Assignments to variables + propagation through constraints
- Conflict between model and constraints: explain by inferences
  $\Longrightarrow$ conflict-driven inferences on demand
- Lemma learning
- Solve conflict by fixing model

Conflict-driven satisfiability: State of the art

- The CDCL procedure: conflict-driven SAT-solving
- Conflict-driven $\mathcal{T}$-sat procedures for fragments $\mathcal{T}$ of arithmetic featuring:
  - Assignments to first-order variables
  - Explanation of conflicts with lemmas containing new (non-input) atoms
- Putting them together: the MCSAT [de Moura and Jovanović] calculus realizes conflict-driven satisfiability modulo one theory

Quest: conflict-driven theory combination

- $\mathcal{T}$ union of disjoint theories $\mathcal{T}_1, \ldots, \mathcal{T}_n$
- MCSAT as a formal system is not a combination calculus
- Equality sharing (aka Nelson-Oppen scheme): combination of $\mathcal{T}_k$-sat procedures as black-boxes
- Conflict-driven behavior and black-box integration are at odds: a conflict-driven $\mathcal{T}$-sat procedure needs to access the trail, post assignments, perform inferences, explain conflicts, export lemmas on a par with CDCL

Answer: CDSAT (Conflict-Driven SATisfiability)

What is CDSAT (Conflict-Driven SATisfiability)

- New method for SMT in a generic combination of disjoint theories $\mathcal{T}_1, \ldots, \mathcal{T}_n$
- Propositional logic (PL) as one of the combined theories
- Combines conflict-driven $\mathcal{T}_k$-sat procedures
- Accommodates non-conflict-driven black-box procedures
- Conflict-driven reasoning in the union of the theories
- Sound, complete, terminating, and it reduces to:
  - CDCL if there is only PL
  - Equality sharing if all $\mathcal{T}_k$-sat procedures are black-boxes
  - DPLL($\mathcal{T}$) with equality sharing if CDCL + black-box $\mathcal{T}_k$-sat procedures
  - MCSAT if CDCL + one conflict-driven $\mathcal{T}$-sat procedure

Assignments of values to terms

- CDSAT treats propositional and theory reasoning similarly: formulas as terms of sort prop (from proposition)
- Assignments take center stage:
  - Boolean assignments to formulas and first-order assignments to first-order terms
  - Problems are written as assignments: SMT and SMA problems
  - Mixed assignments: $(x > 1) \leftarrow \text{false}$,
    $(x > 1) \vee (y < 0) \leftarrow \text{true}$,
    $(store(a, i, v) \simeq b) \leftarrow \text{true}$,
    $y \leftarrow -1$,
    $select(a, j) \leftarrow 3$
- What are values? 3, $\sqrt{2}$ are not in the signature of any theory

Theory extensions to define values

- Theory $\mathcal{T}_k$
- Theory extension $\mathcal{T}_k^+$: add new constant symbols
- Example: add a constant symbol for every number (e.g., integers, rationals, algebraic reals) $\sqrt{2}$ is a constant symbol interpreted as $\sqrt{2}$
- Values in assignments are these constant symbols, called $\mathcal{T}_k$-values (*true* and *false* are values for all theories)
- Conservative theory extension: a $\mathcal{T}_k^+$-unsatisfiable set of $\mathcal{T}_k$-formulas is $\mathcal{T}_k$-unsatisfiable

Theory view of an assignment

- Theory $\mathcal{T}_k$
- The $\mathcal{T}_k$-assignments: those that assign $\mathcal{T}_k$-values
- $u \simeq t$ if there are $u \leftarrow \mathfrak{c}$ and $t \leftarrow \mathfrak{c}$ by any theory
- $u \not\simeq t$ if there are $u \leftarrow \mathfrak{c}$ and $t \leftarrow \mathfrak{q}$ by any theory
- $H = \{x > 1, store(a,i,v) \simeq b, select(a,j) \leftarrow \mathrm{red}, y \leftarrow -1, z \leftarrow 2\}$
  - Boolean view: $\{x > 1,\ store(a,i,v) \simeq b\}$
  - Arrays-view: $\{x > 1,\ store(a,i,v) \simeq b,\ select(a,j) \leftarrow \mathrm{red}\}$
  - LRA-view: $\{x > 1,\ store(a,i,v) \simeq b,\ y \leftarrow -1,\ z \leftarrow 2,\ y \neq z\}$
  - Global view: $H \cup \{y \neq z\}$

Assignments and models: endorsement

- $\mathcal{M} \models (u \leftarrow \mathfrak{c})$: $\mathcal{M}$ interprets u and $\mathfrak{c}$ as the same element
- Theory view and endorsement work together
- $u \leftarrow \mathfrak{c},\ t \leftarrow \mathfrak{c}$: $\mathcal{M} \models\ u \simeq t$
- $u \leftarrow \mathfrak{c},\ t \leftarrow \mathfrak{q}$: $\mathcal{M} \models\ u \not\simeq t$
- $\mathcal{T}_k$-satisfiable: a $\mathcal{T}_k$-model satisfies the $\mathcal{T}_k$-view
- Satisfiable: a $\mathcal{T}$-model satisfies the global view (global endorsement)

Theory modules

- CDSAT combines inference systems called theory modules $\mathcal{I}_1, \ldots, \mathcal{I}_n$ for $\mathcal{T}_1, \ldots, \mathcal{T}_n$
- Inferences deduce Boolean assignments from assignments of any kind (design choice)
- Theory modules for PL, LRA, EUF, Arrays
- CDSAT treats a non-conflict-driven $\mathcal{T}_k$-satisfiability procedure as a theory module whose only inference rule invokes the procedure to detect $\mathcal{T}_k$-unsatisfiability: $l_1 \leftarrow \mathfrak{b}_1, \ldots, l_m \leftarrow \mathfrak{b}_m \vdash_{\mathcal{T}} \bot$

CDSAT trail

- Sequence of assignments: decision or justified assignment
- Decision: either Boolean or first-order; opens the next level
- Justification of A: set H of assignments that appear before A
  - Due to inferences, e.g., $J \vdash_{\mathcal{I}_k} A$
  - Input assignments (empty justification)
  - Due to conflict-solving transitions
  - Boolean or input first-order assignment in SMA
- Level of A: maximum among those of the elements of H
- A justified assignment of level 5 may appear after a decision of level 6

The CDSAT transition system: Decide

Decide: $\Gamma \longrightarrow \Gamma, {}_?(u \leftarrow \mathfrak{c})$ adds a decision
if $u \leftarrow \mathfrak{c}$ is an acceptable $\mathcal{T}_k$-assignment for $\mathcal{I}_k$ in $\Gamma_{\mathcal{T}_k}$:

- $\Gamma_{\mathcal{T}_k}$ does not already assign a $\mathcal{T}_k$-value to u
- $u \leftarrow \mathfrak{c}$ first-order: it does not happen $J \cup \{u \leftarrow \mathfrak{c}\} \vdash_{\mathcal{I}_k} L$, where $J \subseteq \Gamma_{\mathcal{T}_k}$ and $\overline{L} \in \Gamma_{\mathcal{T}_k}$
- u is relevant to $\mathcal{T}_k$:
  either u occurs in $\Gamma_{\mathcal{T}_k}$ and $\mathcal{T}_k$ has $\mathcal{T}_k$-values for its sort;
  or u is an equality whose sides occur in $\Gamma_{\mathcal{T}_k}$,
  $\mathcal{T}_k$ has their sort, but does not have $\mathcal{T}_k$-values for that sort

Relevance: subdivision of work among theories

- $H = \{x \leftarrow 5,\ f(x) \leftarrow 2,\ f(y) \leftarrow 3\}$
- Rational variables x and y are LRA-relevant, not EUF-relevant
- $x \simeq y$ is EUF-relevant (assume EUF has sort Q), not LRA-relevant
- LRA can make x and y equal/different by assigning them the same/different value
- EUF can make x and y equal/different by assigning a truth value to $x \simeq y$
- Two ways to communicate equalities

The CDSAT transition system: Deduce

- Deduce: $\Gamma \longrightarrow \Gamma, {}_{J\vdash}L$ adds a justified assignment
  - $J \vdash_{\mathcal{I}_k} L$, for some k, $1 \leq k \leq n$, $J \subseteq \Gamma$, and $L \notin \Gamma$
  - $\overline{L} \notin \Gamma$
  - L is $l{\leftarrow}\mathfrak{b}$ for some $l \in \mathcal{B}$
  - $\mathcal{B}$: finite global basis to draw new terms from for the purpose of termination
- Both theory propagation and theory explanation of $\mathcal{T}_k$-conflict

The CDSAT transition system: Fail and ConflictSolve

- Conflict: an unsatisfiable assignment
- $J \vdash_{\mathcal{I}_k} L$, for some k, $1 \leq k \leq n$, $J \subseteq \Gamma$, $L \notin \Gamma$
- $\overline{L} \in \Gamma$: $J \cup \{\overline{L}\}$ is a conflict
- Fail: $\Gamma \longrightarrow$ unsat declares unsatisfiability
  if $\mathrm{level}_\Gamma(J \cup \{\overline{L}\}) = 0$
- ConflictSolve: $\Gamma \longrightarrow \Gamma'$
  solves the conflict by calling the conflict-state rules
  if $\mathrm{level}_\Gamma(J \cup \{\overline{L}\}) > 0$ and $\langle \Gamma; J \cup \{\overline{L}\} \rangle \Longrightarrow^* \Gamma'$

The CDSAT transition system: conflict-state rules

- The conflict contains an assignment that stands out because its level is maximum in the conflict:
  - If this outstanding assignment is Boolean: Backjump rule
  - If this outstanding assignment is first-order: UndoClear rule
- Otherwise:
  - Explain the conflict by resolving upon a Boolean assignment: Resolve rule

The CDSAT transition system: UndoClear

UndoClear: $\langle \Gamma; E \uplus \{A\} \rangle \Longrightarrow \Gamma^{\leq m-1}$

- A is a first-order decision of level $m > \mathrm{level}_\Gamma(E)$
- Removes A and all assignments of level $\geq m$
- $\Gamma^{\leq m-1}$: the restriction of trail Γ to its elements of level at most $m-1$

Example of UndoClear

$\Gamma \;=\; -2x - y < 0,\;\; x + y < 0,\;\; x < -1$ (level 0)

1. Decide $y \leftarrow 0$ (level 1)
2. LRA-conflict is $\{-2{\cdot}x{-}y{<}0,\; x{<}{-}1,\; y{\leftarrow}0\}$
3. Deduce $-y < -2$ from $-2x - y < 0$ and $x < -1$ (level 0)
4. LRA-conflict is $\{y \leftarrow 0,\; -y < -2\}$
5. UndoClear removes $y \leftarrow 0$ resulting in
   $\Gamma \;=\; -2x - y < 0,\;\; x + y < 0,\;\; x < -1,\;\; -y < -2$ (level 0)

Example of Resolve

Γ_0 includes: $(\neg l_4 \vee l_5)$, $(\neg l_2 \vee \neg l_4 \vee \neg l_5)$ (level 0)

1. Decide: A_1 (level 1)
2. Decide: l_2 (level 2)
3. Decide: A_3 (level 3)
4. Decide: l_4 (level 4)
5. Deduce: l_5 with justification $\{\neg l_4 \vee l_5,\ l_4\}$ (level 4)
6. Conflict: $\{\neg l_2 \vee \neg l_4 \vee \neg l_5,\ l_2,\ l_4,\ l_5\}$
7. Resolve: $\{\neg l_2 \vee \neg l_4 \vee \neg l_5,\ l_2,\ l_4,\ \neg l_4 \vee l_5\}$

The CDSAT transition system: Resolve

Resolve: $\langle \Gamma; E \uplus \{A\} \rangle \Longrightarrow \langle \Gamma; E \cup H \rangle$

- A is a justified assignment ${}_{H\vdash} A$
- Replace A by its justification H
- Provided H does not contain a first-order decision A' whose level is $\mathrm{level}_{\Gamma}(E \uplus \{A\})$ (i.e., maximum)
- Avoiding a "Resolve, UndoClear, Decide" loop (first-order decisions do not have complement)
- And what if there is such an A'? UndoDecide rule

The CDSAT transition system: UndoDecide

UndoDecide: $\langle \Gamma; E \uplus \{L\} \rangle \Longrightarrow \Gamma^{\leq m-1}, {}_{?}\overline{L}$

- L is a Boolean justified assignment ${}_{H\vdash} L$ with $m = \mathrm{level}_{\Gamma}(E) = \mathrm{level}_{\Gamma}(L)$
- Neither Backjump nor UndoClear apply
- H contains a first-order decision A' of level m: Resolve does not apply
- UndoDecide removes A' and decides $\overline{L}$

Example of UndoDecide

$\Gamma \;=\; x > 1 \vee y < 0, \;\; x < -1 \vee y > 0$ (level 0)

1. Decide: $x \leftarrow 0$ (level 1)
2. Deduce: $(x > 1) \leftarrow \mathit{false}$ (level 1)
   $(x < -1) \leftarrow \mathit{false}$ (level 1)
   $y < 0$ (level 1)
   $y > 0$ (level 1)
3. LRA-conflict: $\{y{<}0, \; y{>}0\}$
4. Resolve: $\{x > 1 \vee y < 0, \; x < -1 \vee y > 0, \; x > 1 \leftarrow \mathit{false},$
   $x < -1 \leftarrow \mathit{false}\}$
5. UndoDecide: $x > 1$ (level 1)

The CDSAT transition system: LearnBackjump

LearnBackjump: $\langle \Gamma; E \uplus H \rangle \Longrightarrow \Gamma^{\leq m}, {}_{E\vdash} F$

- H contains only Boolean assignments: H as $L_1 \wedge \ldots \wedge L_k$
- Since $E \uplus H \models \perp$, it is $E \models \overline{L_1} \vee \ldots \vee \overline{L_k}$
- Learned lemma: $F = \overline{L_1} \vee \ldots \vee \overline{L_k}$ (clausal form of H)
- Provided $F \notin \Gamma$, $\overline{F} \notin \Gamma$, $F \in \mathcal{B}$
- Choice of level where to backjump to: $\mathrm{level}_\Gamma(E) \leq m < \mathrm{level}_\Gamma(H)$

Examples of learning and backjumping (continued)

Conflict: $\{\neg l_2 \vee \neg l_4 \vee \neg l_5,\ l_2,\ l_4,\ \neg l_4 \vee l_5\}$

- LearnBackjump with $H = \{l_2, l_4\}$: learns the first assertion clause $\neg l_2 \vee \neg l_4$ with justification $\{\neg l_2 \vee \neg l_4 \vee \neg l_5,\ \neg l_4 \vee l_5\}$ (level 0)
- With destination level $m = 0$: restart
- With destination level $m = 2$: Deduce: $\overline{l_4}$ with justification $\{\neg l_2 \vee \neg l_4,\ l_2\}$

Proofs in CDSAT

- Proof objects in memory (checkable by proof checker)
  - The theory modules produce proofs
  - Proof-carrying CDSAT transition system
  - Proof reconstruction: from proof terms to proofs (e.g., resolution proofs)
- LCF style as in ITP (correct by construction)
  - Trusted kernel of primitives

Implementation

- MCSAT as add-on in DPLL(T)-based solvers Z3, CVC4, Yices
- MCSAT/CDSAT with the E-graph at the center:
  paper by François Bobot, Stéphane Graham-Lengrand, Bruno Marre and Guillaume Bury at this workshop
- CDSAT in C++: forthcoming SMT solver Eos
  (by Giulio Mazzi at U. Verona)
- Several issues, e.g.:
  - Heuristic strategies to make decisions and prioritize theory inferences
  - Efficient techniques to detect the applicability of theory inference rules and the acceptability of assignments