CS 511, Fall 2018, Handout 35

SMT Solver = SAT Solver + a theory

(continuation of *Handout 34: SAT Solvers*)

Assaf Kfoury

3 December 2018

PRELIMINARIES

- SMT = ***S**atisfiability **M**odulo a **T**heory*.
- Theory = *typically a quantifier-free fragment of a first-order theory*.[1]
- SMT Solver = *SAT solver* working with a *theory solver* (or *T-solver*).

[1] See Handout 18 for examples of first-order theories.

PRELIMINARIES

- SMT = ***S****atisfiability* ***M****odulo a* ***T****heory*.
- Theory = *typically a quantifier-free fragment of a first-order theory*.[1]
- SMT Solver = *SAT solver* working with a *theory solver* (or *T-solver*).
- Examples of first-order theories considered in SMT solvers, in each case limited to the quantifier-free fragment:
  - Equality with Uninterpreted Functions (EUF) – Handout 18, pp 3-4
  - Linear Integer Arithmetic (LIA) – Handout 18, page 19
  - Linear Real Arithmetic (LRA) – similar to LIA, except that the domain is $\mathbb{Q}$ (set of rationals) or $\mathbb{R}$ (set of reals)
  - Difference Logic (DL), which is a fragment of LRA
  - other theories: Arrays, Bit-Vectors, Tuples and Records, Algebraic Datatypes, etc.

[1] See Handout 18 for examples of first-order theories.

PRELIMINARIES

- SMT = ***S**atisfiability **M**odulo a **T**heory*.
- Theory = *typically a quantifier-free fragment of a first-order theory*.[1]
- SMT Solver = *SAT solver* working with a *theory solver* (or *T-solver*).
- Examples of first-order theories considered in SMT solvers, in each case limited to the quantifier-free fragment:
  - Equality with Uninterpreted Functions (EUF) – Handout 18, pp 3-4
  - Linear Integer Arithmetic (LIA) – Handout 18, page 19
  - Linear Real Arithmetic (LRA) – similar to LIA, except that the domain is $\mathbb{Q}$ (set of rationals) or $\mathbb{R}$ (set of reals)
  - Difference Logic (DL), which is a fragment of LRA
  - other theories: Arrays, Bit-Vectors, Tuples and Records, Algebraic Datatypes, etc.
- Reason for the restriction to quantifier-free fragments: Given a theory T, we need an efficient decision procedure to decide validity relative to T, *i.e.*, to "quickly" decide whether $T \vdash \varphi$.

[1] See Handout 18 for examples of first-order theories.

Two General Approaches to SMT Solving

[2] CDCL SAT-solver = Conflict-Driven Clause-Learning SAT-solver, a variant of the DPLL procedure.

Two General Approaches to SMT Solving

- Eager Methods

- Lazy Methods

[2] CDCL SAT-solver = Conflict-Driven Clause-Learning SAT-solver, a variant of the DPLL procedure.

Two General Approaches to SMT Solving

- Eager Methods
  - Convert SMT problem into an equisatisfiable SAT problem.
  - Example theories for which eager methods work well:
    Equality, Difference Logic, Bit-Vectors.
- Lazy Methods

[2] CDCL SAT-solver = Conflict-Driven Clause-Learning SAT-solver, a variant of the DPLL procedure.

Two General Approaches to SMT Solving

- Eager Methods
  - Convert SMT problem into an equisatisfiable SAT problem.
  - Example theories for which eager methods work well:
    Equality, Difference Logic, Bit-Vectors.
- Lazy Methods
  - Interleave SAT-solver steps with T-solver steps,
    but keep the two separate.
  - More widely applicable than eager methods.
  - Most common approach:
    CDCL SAT-solver combined with a T-solver.[2]

[2] CDCL SAT-solver = Conflict-Driven Clause-Learning SAT-solver, a variant of the DPLL procedure.

(THIS PAGE INTENTIONALLY LEFT BLANK)