Satisfiability modulo theories

Verifying cyberphysical systems

Sayan Mitra

mitras@illinois.edu

Some of the slides and examples for this lecture are from Clark Barrett

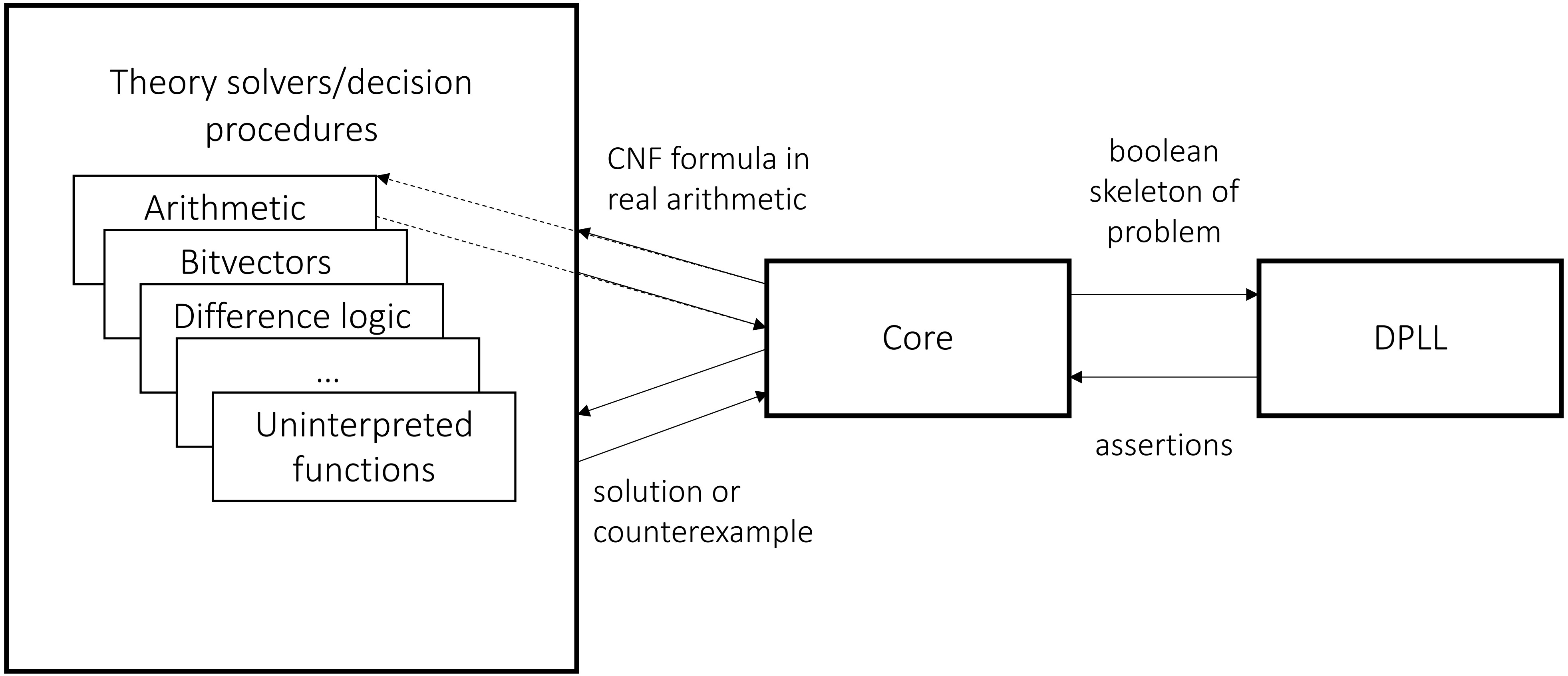
Today

- Satisfiability modulo theories (SMT)
  - Theories, models, decision procedures
  - Examples
- Brief z3 tutorial (see notebook)

Satisfiability modulo theories

- SAT: Given a *well-formed formula* in propositional logic, determine whether there exists a satisfying solution
- A *satisfiability modulo theory* (SMT) problem is a generalization of SAT in which some of the binary variables are replaced by predicates over a suitable set of non-binary variables
- $\phi_1(w,x,y,z) := (x - y = 5) \wedge (z - y \geq 2) \wedge (z - x > 2) \wedge (w - x = 2)$
- $\phi_2(x,y,z) := (3x^2 - 4y + 5z \leq 5) \wedge (-2x + 5z^3 \leq 7)$
- ϕ_1 is a predicate in *difference logic* in which the variables are real-valued, and the clauses are constructed with standard comparison operations >, >=, =$ and –(minus)
- ϕ_2 is a predicate in real arithmetic

Architecture of an SMT solver



$\langle logic \rangle$

A short overview of theories, models, decision procedures

What is a theory in mathematical logic?

- When we talk about well-formed formulas with non-binary variables, we have to say exactly what type of formulas are allowed
- and, what it means for assignments to *satisfy such formulas*
- This brings us to the notions *theory* and *models* in mathematical logic

Building up a theory

- First we define the syntax for writing formulas
- A *signature* $\Sigma = (\Sigma_F, \Sigma_P, V)$
  - set of *function symbols* $\Sigma_F, e.g., \{+, -, f, g, sin, \ldots\}$
  - set of *predicate symbols* Σ_P
  - *arity* of each function: $arity: \Sigma_F \to \mathbb{N}$
  - *0 arity* functions are constants
  - V: set of *variables*
- $Terms(\Sigma, V)$
  - Elements of V are terms
  - If $t_1, \ldots, t_k \in Terms(\Sigma, V)$ and $f \in \Sigma_F$ with arity k, then $f(t_1, \ldots, t_k) \in Terms(\Sigma, V)$
  - *Ground terms* are terms without variables

- $\Sigma_F = \{0, +\}, \Sigma_P = \{<\}$
- $arity(0) = 0$
- $arity(+) = 2$
- $arity(<) = 2$
- $V = \{x, y, z\}$

- Terms defined by this signature are $x, y, z, +(x, y), +(+(x, y), 0), 0, \ldots$

Terms to Formulas

- Atomic formulas AF
  - True, False
  - If $t_1, \dots, t_k \in Terms(\Sigma, V)$ and p $\in \Sigma_P$ with arity k, then $\mathrm{p}(t_1, \dots, t_k) \in AF(\Sigma, V)$
  - A *literal* is an AF or its negation
  - Set of all atomic formulas $AF(\Sigma, V)$
- Quantifier free formulas $QFF(\Sigma, V)$
  - AF
  - if $\phi_1, \phi_2 \in QFF$ then
    - $\neg\phi_1 \in QFF, \phi_1 \wedge \phi_2 \in QFF, \phi_1 \vee \phi_2 \in QFF, \phi_1 \to \phi_2 \in QFF$
  - Set of all quantifier free formulas $QFF(\Sigma, V)$
- First order formulas is the set of quantifier free formulas under universal and existential quantifiers
  - Bound variables are those that are attached to quantifiers
  - Free variables: variables not bound
- *Sentence:* First order formula with no free variables
- $\mathrm{Theory}(\Sigma, V)$ set of all sentences over (Σ, V)

- $x < y$
- $+(x, y) = +(y, x)$

- $+(x, y) = 0 \wedge x > y$

- $\forall x, \exists y: +(x, y) = 0$
- $\forall x, \exists y: x < y$
- $\forall x, \exists y: +(x, y) = x$

- $\exists x: +(x, c) = x$

Models for theories

This notion of model from mathematical logic is not to be confused with the notion of a model for a computational or physical process

- A *model* gives meanings or *interpretations* to formulas in theory T
- A model M for $\mathrm{T} = \ \mathrm{Theory}(\Sigma, V)$ has to define
  - A domain |M|
  - interpretations of all functions and predicate symbols
  - $M(f)$: $|M|^n \to |M|$ if arity$(f) = n$
  - $M(p) \subseteq |M|^n$ if $arity(p) = n$
  - Assignment $M(x) \in |M|$ for every variable $x \in V$
- A formula ϕ is true in M if it evaluates to true under the given interpretations over domain M

Example model for $\Sigma = \{0, +, <\}$
$|M| = \{a, b, c\}$
$M(0) = a$

$M(<) = \{\langle a, b\rangle, \langle a, c\rangle, \langle b, c\rangle\}$

$M(+)$	a	b	$\cancel{b}$ c
a	a	b	c
b	b	c	a
c	c	a	b

if $M(x) = a, M(y) = b$

then $M(+(x, y))$ is $M(+)\big(M(x), M(y)\big) =$
$M(+)(a, b) = b$
$M(+(+(x, y), y) = c$
$M \vDash \forall x\, \exists y + (x, y) = x$

We say that the model M *T-satisfies* the formula ϕ

Decision procedures

Given a theory T a theory solver or a decision procedure for T takes as input a set of literals ϕ (atomic propositions) and determines whether ϕ is T-satisfiable, that is,

$\exists$ a model M such that $M \vDash \phi$?

$\langle\backslash logic\rangle$

A short overview of theories and models in mathematical logic

Example theories

- Uninterpreted functions (UF) $\Sigma_F := \{f, g, \dots\}, \Sigma_P := \{=\}, V := \{x_i\}$
  - $x_1 = x_2 \ \wedge \ x_3 \neq x_2 \ \wedge \ f(x_3) \neq f(x_2)$
- Difference logic $\Sigma_F := \{1,2, .., -\}, \Sigma_P := \{<, \leq, =, >, \geq\}$
  - $x_1 - x_2 \gtreqless k$, where $\gtreqless \in \{<, \leq, =, >, \geq\}$
- Linear arithmetic
  - $4x - 3y + 6z \leq 10$
- Real arithmetic (nonlinear)
  - $4x^2 + 6y - 9z^3 \leq 5$
- Bit vectors
- Arrays
  - $x'[i] = x[i] + 1$

Example decision procedure 1: Difference logic

$$\phi = (x - y = 5) \land (z - y \geq 2) \land (z - x > 2) \land (w - x = 2) \land (z - w < 0)$$

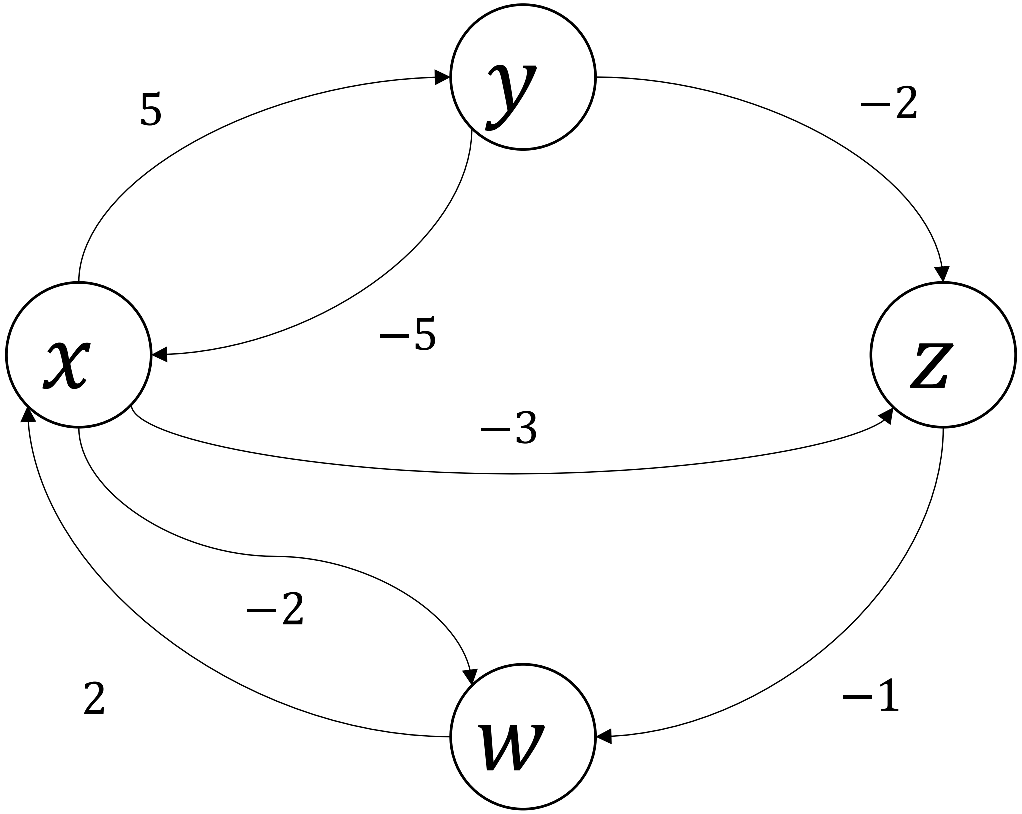
Decision procedure:

Convert each literal (AF) to $x_1 - x_2 \leq c$ form:

$$\begin{gathered}\phi' = (x - y \leq 5) \land (y - x \leq -5) \\ \land (y - z \leq -2) \land \\ (x - z \leq -3) \land \\ (w - x \leq 2) \land (x - w \leq -2) \\ (z - w \leq {\color{red}-1})\end{gathered}$$

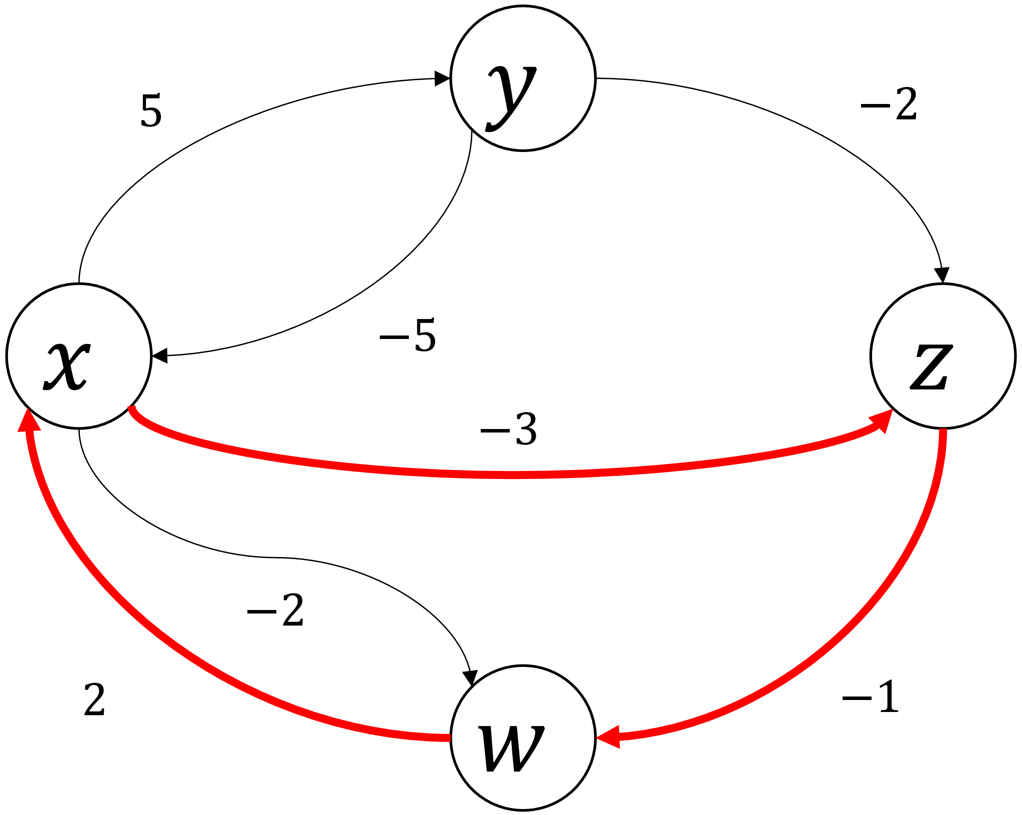
$$\phi' = (x - y \leq 5) \wedge (y - x \leq -5)$$
$$\wedge (y - z \leq -2) \wedge$$
$$(x - z \leq -3) \wedge$$
$$(w - x \leq 2) \wedge (x - w \leq -2)$$
$$(z - w \leq -1)$$

Construct a graph with edge from $x \rightarrow^{c} y$ for each literal ϕ'



$$\phi' = (x - y \leq 5) \wedge (y - x \leq -5) \\ \wedge (y - z \leq -2) \wedge \\ (x - z \leq -3) \wedge \\ (w - x \leq 2) \wedge (x - w \leq -2) \\ (z - w \leq -1)$$

Construct a graph $G_{\phi'}$ with edge from $x \rightarrow^c y$ for each literal ϕ'



Proposition. ϕ is satisfiable iff $G_{\phi'}$ is negative cycle free.

Exercise.

Example decision procedure 2: Uninterpreted functions (UF)

$$\phi = \; x_1 = x_2 \wedge (x_2 = x_3) \; \wedge (x_4 = x_5) \; \wedge (x_5 \neq x_1) \wedge \;\; (F(x_1) \neq F(x_3))$$

Decision procedure

1. Put all variables and function instances in their own classes
2. If $t_1 = t_2$ is a literal then merge the classes containing them; do this repeatedly
3. If t_1 and t_2 are terms in the same class then merge classes containing $F(t_1)$ and $F(t_2)$; repeat
4. If $t_1 \neq t_2$ is a literal in ϕ and they belong to the same class then return unsat else return sat t_1 and t_2

Example decision procedure 2: Uninterpreted functions (UF)

Initial classes $\phi = \; x_1 = x_2 \wedge (x_2 = x_3) \; \wedge (x_4 = x_5) \; \wedge (x_5 \neq x_1) \wedge \; (F(x_1) \neq F(x_3))$

Classes $\{x_1\}\,\{x_2\}\{x_3\}\{x_4\}\{x_5\}\{F(x_1)\}\{F(x_3)\}$

$\{x_1, x_2, x_3\}\,\{x_4, x_5\}\{F(x_1)\}\{F(x_3)\}$

$\{x_1, x_2, x_3\}\,\{x_4, x_5\}\{F(x_1), F(x_3)\}$

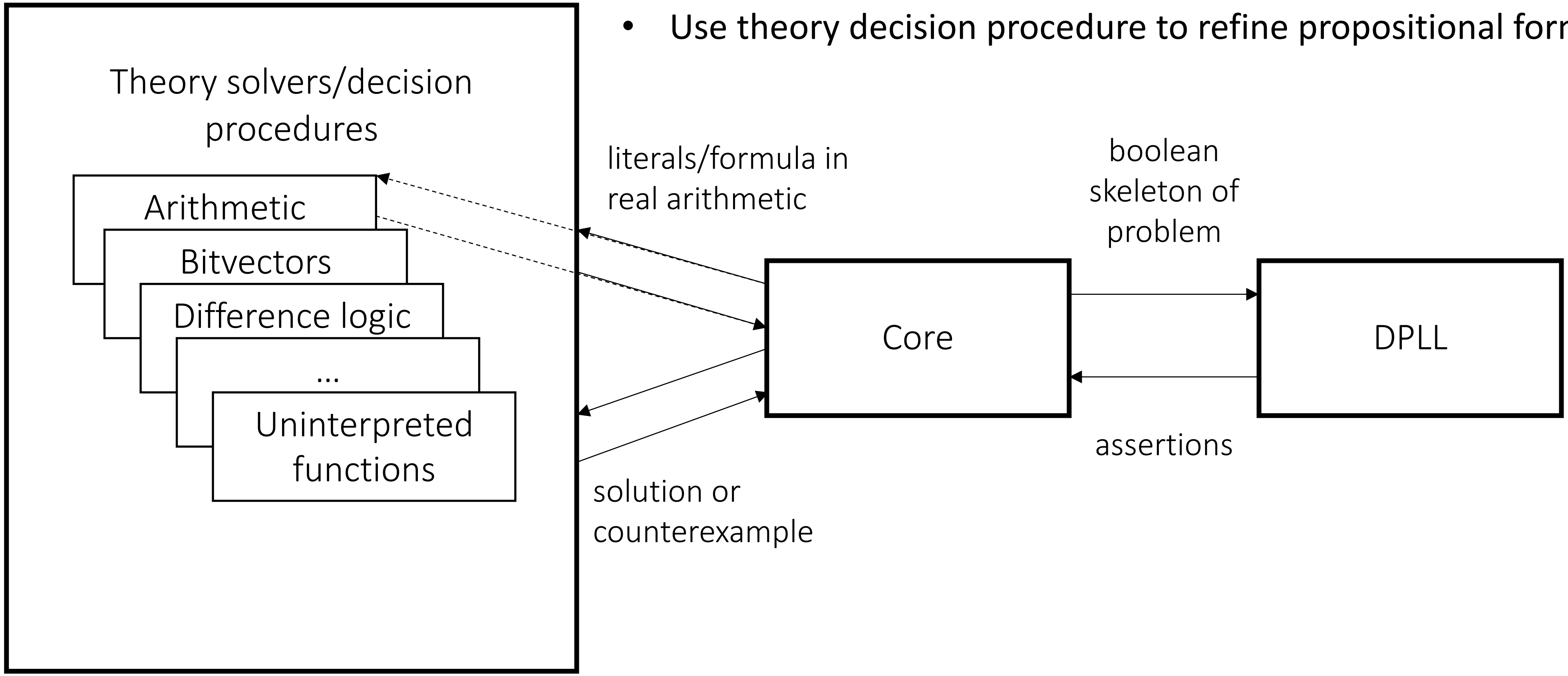
$Unsat$

Return to SMT

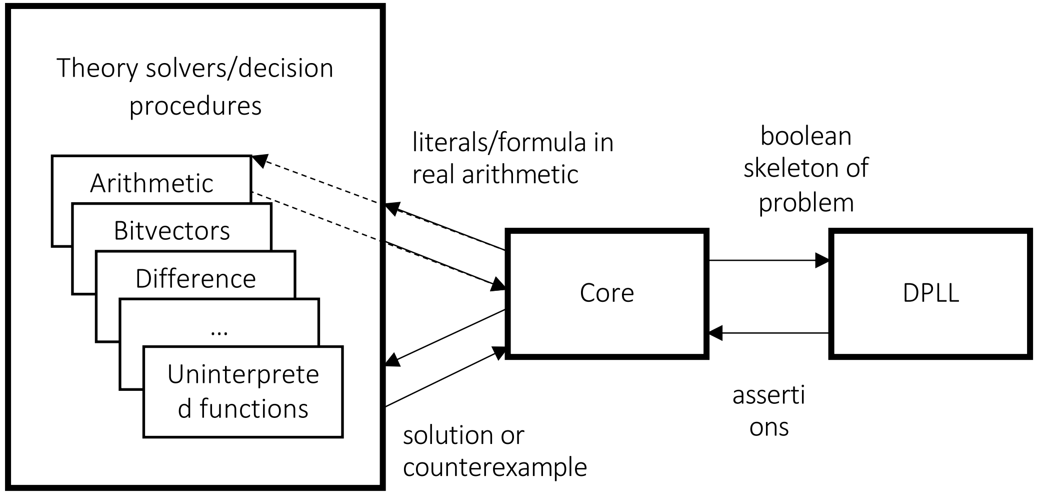
$$\phi \equiv g(a) = c \wedge f\big(g(a)\big) \neq f(c) \vee g(a) = d \wedge c \neq d$$

Several approaches, lazy approach:

- Abstract ϕ to propositional form
- Feed to DPLL
- Use theory decision procedure to refine propositional formula a guide SAT



- $\phi \equiv \underbrace{g(a) = c}_{1} \wedge \underbrace{f\big(g(a)\big) \neq f(c)}_{\bar{2}} \vee \underbrace{g(a) = d}_{3} \wedge \underbrace{c \neq d}_{\bar{4}}$



- send $\{1, \bar{2} \vee 3, \bar{4}\}$ to DPLL
- returns model $\{1, \bar{2}, \bar{4}\}$
- UF solver concretizes to $g(a) = c$, $f\big(g(a)\big) \neq f(c)$, $c \neq d$
- checks this as UNSAT
- send $\{1, \bar{2} \vee 3, \bar{4}, \bar{1} \vee 2 \vee 4\}$ to DPLL
- returns model $\{1, 2, 3, \bar{4}\}$
- UF solver concretizes and finds this to be UNSAT
- send $\{1, \bar{2} \vee 3, \bar{4}, \bar{1} \vee 2 \vee 4, \bar{1} \vee \bar{2} \vee \bar{3} \vee 4\}$ to DPLL
- returns UNSAT

Assignments

- HW1
- Learn z3
  - https://ericpony.github.io/z3py-tutorial/guide-examples.htm

Readings

- Chapter 7.5.3 of CPSBook on using SAT/SMT for verification
- Read chapter 4 for next week
- Reading more about decision procedures