Lecture 5: Satisfiability modulo theories

Huan Zhang

huan@huan-zhang.com

Slides adapted from Prof. Sayan Mitra's slides in Fall 2021
Some of the slides for this lecture are adapted from slides by Clark Barrett

Previous lectures

- Boolean satisfiability problem

$$\alpha := (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_4) \wedge (\neg x_2 \vee \neg x_3 \vee \neg x_4)$$

DPLL algorithm (backtracking + unit propagation + pure literal assignment)

```
function DPLL(α)
    unit-propagate
    pure-literal-assign
    check-stopping-conditions
    l ← choose-literal(α);
    return (DPLL(α ∧ {l}) or
            DPLL(α ∧ {¬l}));
```

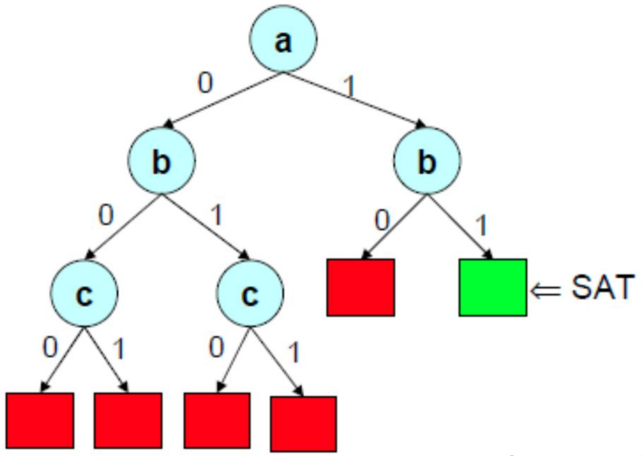


Image from Wikipedia

Today

- Satisfiability modulo theories (SMT): more general satisfiability problem
  - Theories, models, decision procedures
  - Uninterpreted Functions
  - Difference Logic
- Brief z3 tutorial (see notebook)

Satisfiability modulo theories

- SAT: Given a *well-formed formula* in propositional logic, determine whether there exists a satisfying solution

$$\alpha := (\neg x_1 \lor x_2) \land (\neg x_2 \lor x_4) \land (\neg x_2 \lor \neg x_3 \lor \neg x_4)$$

- 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

Satisfiability modulo theories

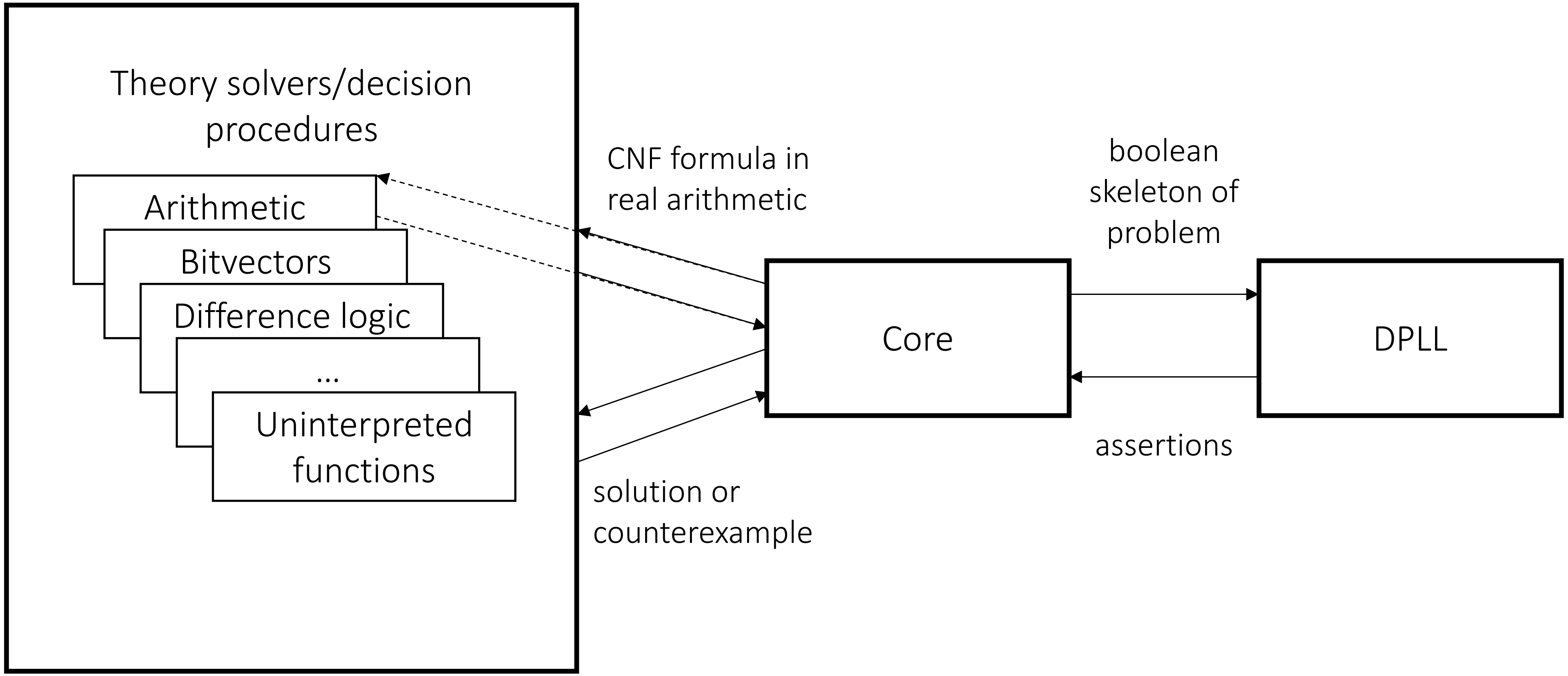
- SAT: Given a *well-formed formula* in propositional logic, determine whether there exists a satisfying solution

$$\alpha := (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_4) \wedge (\neg x_2 \vee \neg x_3 \vee \neg x_4)$$

- 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



$\langle model\ theory \rangle$

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 some basic notions in mathematical logic
  - *theory* --- what does a well-formed formula look like ?
  - *models* --- what does it mean to satisfy a formula?

Building up a theory

First, we define the syntax for writing **formulas**

A *signature* $\Sigma = (\Sigma_F, \Sigma_P, V)$

- Σ_F : set of *function symbols*, $e.g.,\ \{+, -, f, g, sin, \ldots\}$
- Σ_P : set of *predicate symbols*, *e.g.*, {<, >, <=...}
    - *arity* of each function: $arity\colon \Sigma_F \rightarrow \mathbb{N}$
    - *0 arity* functions are constants
- V*:* set of *variables*

$Terms(\Sigma, V)$*:* combines variables and functions

- 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: combines terms and predicates
  - True, False
  - If $t_1, \ldots, t_k \in Terms(\Sigma, V)$ and $\mathrm{p} \in \Sigma_P$ with arity k, then $\mathrm{p}(t_1, \ldots, t_k) \in AF(\Sigma, V)$
  - 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
    - $\neg\phi_1 \in QFF,\ \phi_1 \wedge \phi_2 \in QFF, \phi_1 \vee \phi_2 \in QFF,\ \phi_1 \rightarrow \phi_2 \in QFF$
  - 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)

AF examples:

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

QFF examples:

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

First order formulas:

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

Sentence:

- $\exists x: +(x, 1) = x$

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 $|\mathrm{M}|$
  - interpretations of all **functions** and **predicate** symbols
  - $M(f): |M|^n \rightarrow |M|$ if $\mathrm{arity}(f) = n$
  - $M(p) \subseteq |M|^n$ 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

A *model* gives meanings or *interpretations* to formulas in theory T

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

$M(+)$	a	b	b
a	a	b	c
b	b	c	a
c	c	a	b

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

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

Define formula $\phi{:} = \forall x\, \exists y + (x, y) = x$
$M \vDash \forall x\, \exists y + (x, y) = x$

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 $M \vDash \phi$?

$\langle\backslash$ model $theory\rangle$

A short overview of theories and models in mathematical logic

Example theories

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

Uninterpreted functions

Useful for abstractly reasoning about programs

- $\Sigma_F := \{f, g, \ldots\}$, $\Sigma_P := \{=\}$, $V := \{x_i\}$

Literals are of the form $x_1 = x_2 \ \wedge \ x_3 \neq x_2 \ \wedge \ f(x_3) \neq f(x_2)$

We know nothing about *f, g*, ... except for its name and arity

How to check its satisfiability?

Decision procedure for 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

Decision procedure for 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 $\{x_1\}\ \{x_2\}\{x_3\}\{x_4\}\{x_5\}\{F(x_1)\}\{F(x_3)\}$

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

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

$Unsat$

Difference Logic (conjunctive fragment)

A useful fragment of linear arithmetic
$\Sigma_F := \{1, 2, .., -\ \}$
$\Sigma_P := \{<\ , \leq\ ,\ \ =\ , \neq\ , >\ , \geq\}$

Literals are of the form $x_1 - x_2 \gtreqless k$, where $\gtreqless \in \{<\ , \leq\ ,\ \ =\ , >\ , \geq\}$

x_1, x_2 are Integers or rational variables

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

Satisfiability problem: checking whether this formula is consistent

An Application: Job shop scheduling problem

Given a finite set of n jobs. Each job i of which consists of a chain of operations (m_1^i, d_1^i), (m_2^i, d_2^i),… There is a finite set of m machines $M = \{m_1, m_2, \ldots, m_m\}$, each of which can handle at most one operation at a time.

The problem of finding a shortest schedule---allocation of machine time to jobs---can be formulated in DL.

Decision procedure for Difference logic

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

Decision procedure:

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

Decision procedure for Difference logic

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

Decision procedure:

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

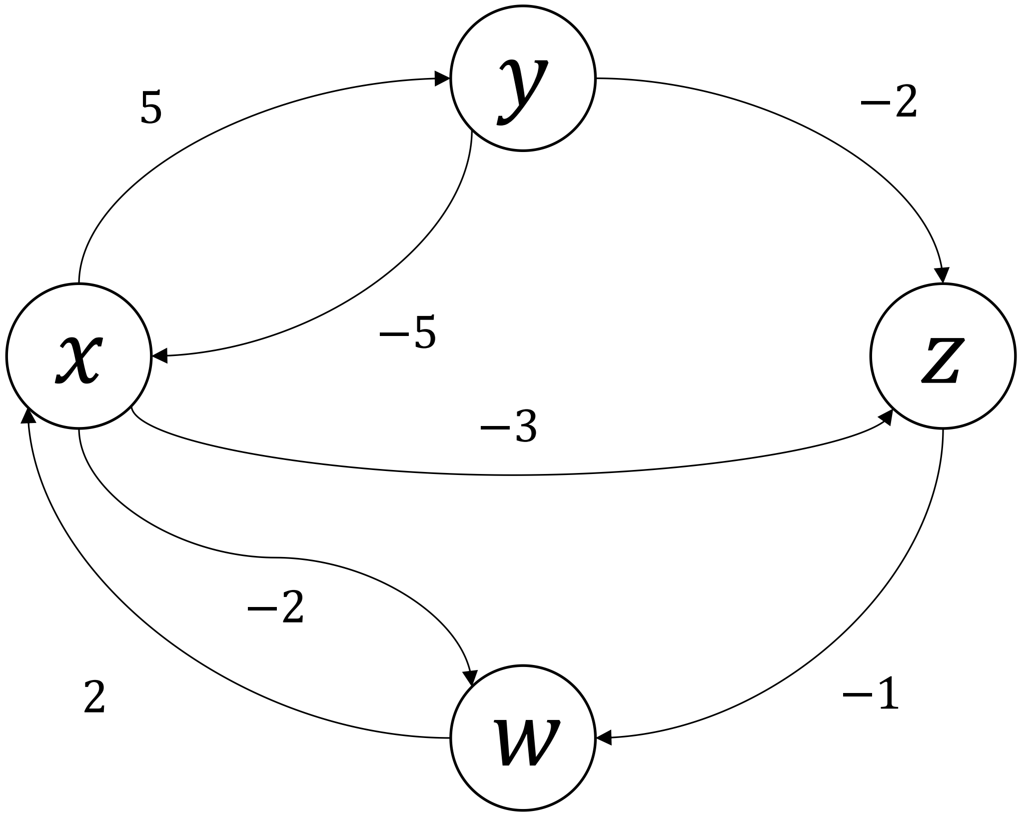
$$\begin{aligned}\phi' = \ & (x - y \leq 5) \wedge (y - x \leq -5) \\ & \wedge (y - z \leq -2) \wedge \\ & (x - z \leq -3) \wedge \\ & (w - x \leq 2) \wedge (x - w \leq -2) \\ & (z - w \leq -1)\end{aligned}$$

For integer domain $(x_1 - x_2 < k)$ is replaced by $(x_1 - x_2 \leq k - 1)$

How to check satisfiability or consistency of formula ϕ'?

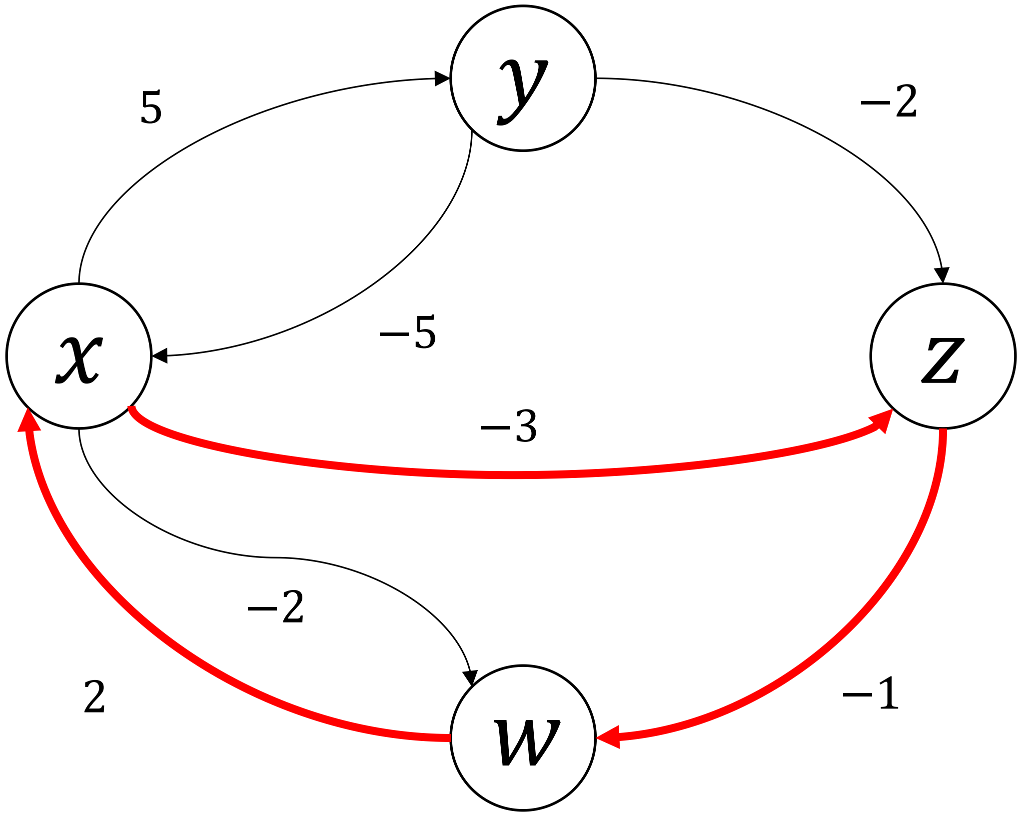
$$\phi' = (x - y \leq 5) \wedge (y - x \leq -5) \wedge (y - z \leq -2) \wedge (x - z \leq -3) \wedge (w - x \leq 2) \wedge (x - w \leq -2) \; (z - w \leq -1)$$

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

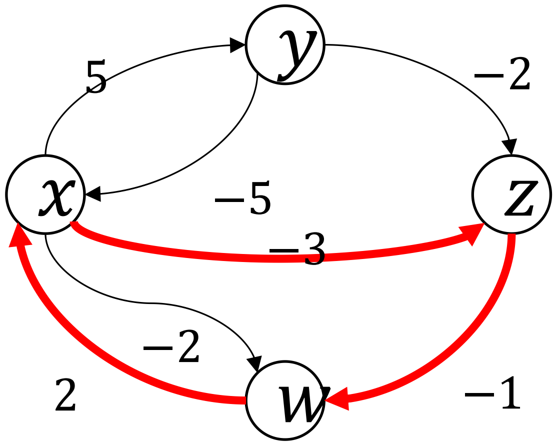


Proposition. ϕ is satisfiable iff $G_{\phi'}$ is negative cycle free.

Proof. (<=) If there is a negative cycle then

$(x - z \leq -3); (z - w \leq -1); (w - x \leq 2)$

adding all up: $(0 \leq -2)$ which is inconsistent.



Proposition. ϕ is satisfiable iff G_{ϕ}, is negative cycle free.

Proof. (<=) If there is a negative cycle then

$(x - z \leq -3); (z - w \leq -1); (w - x \leq 2)$ adding all up: $(0 \leq -2)$ which is inconsistent.

(=>) Let us assume that there is no negative cycle. We will construct a satisfying solution $\sigma: V \to \mathbb{Z}$

Consider additional vertex o with $o \to^{0} v$ edges for all v

For each variable v define solution as the shortest distance from o to v (be aware of negative distances): $\sigma(v) = -dist(o, v)$

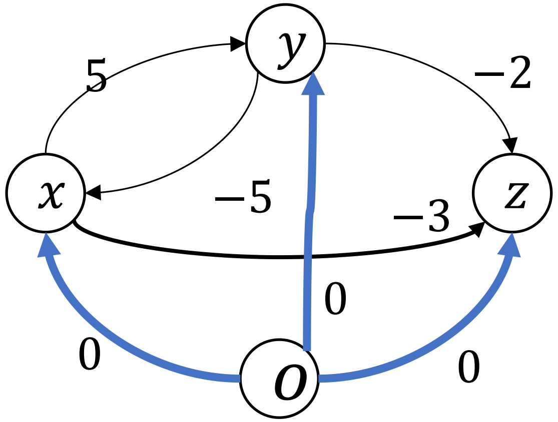
Suppose FSOC, σ does not satisfy a literal $x - y \leq k$ then

$-dist(o, x) + dist(o, y) > k$

$dist(o, y) > k + dist(o, x)$

$dist(o, y) > dist(x, y) + dist(o, x)$

violates definition of $dist(o, y)$!



$\sigma(x) = -dist(o, x) = 5$

$\sigma(y) = -dist(o, y) = 0$

$\sigma(z) = -dist(o, z) = 8$

Summary of DP for Difference Logic

- Satisfiability check for conjunctive fragment of DL can be performed using Bellman-Ford algorithm in time $O(|V|.|E|)$
- Inconsistency/unsatisfiability explanations are negative cycles
- Amenable to incremental checks

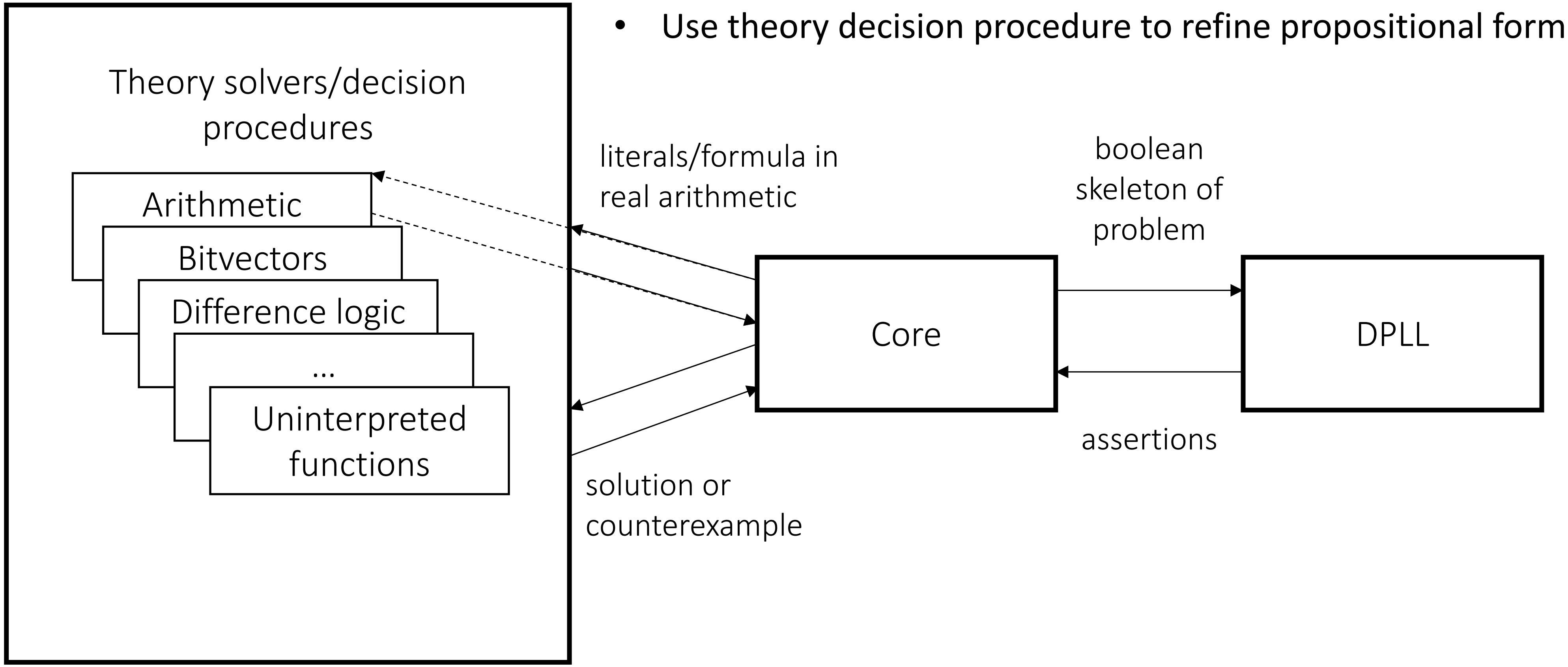
Return to SMT

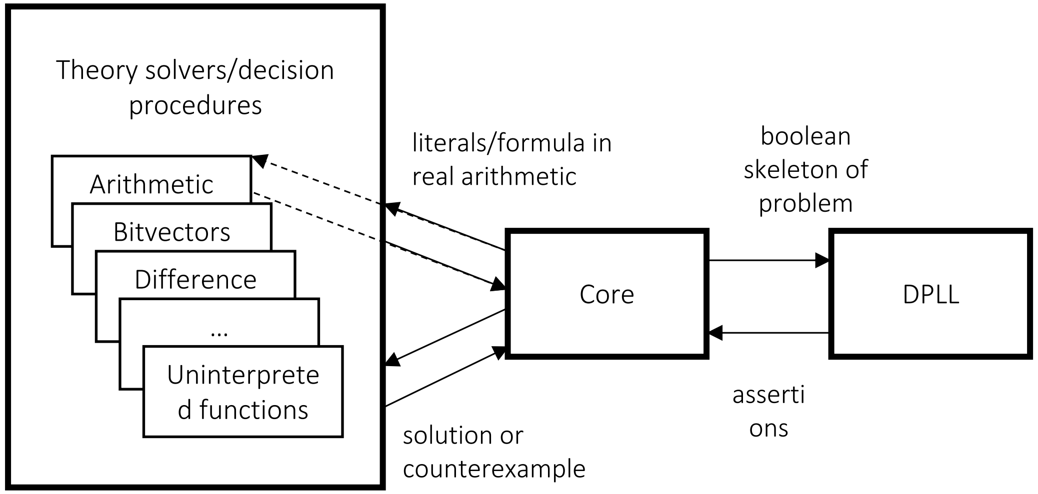
$$\phi \equiv (g(a) = c) \land (f(g(a)) \neq f(c) \lor g(a) = d) \land c \neq d$$

Several approaches, lazy approach:

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

$$\phi \equiv x_1 \land (\neg x_2 \lor x_3) \land \neg x_4$$





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

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

Readings

- Reading more about decision procedures
- Learn neural networks if you never used them