

Lecture 6: Satisfiability modulo theories

Part 2: DPLL(T) and Simplex Algorithm

Sayan Mitra

Today

- SMT (continued)
 - Difference Logic
- Decision procedure for Linear Real Arithmetic
Simplex Algorithm [Dantzig 1947]
- Next 2 - 3 weeks: Verification of Neural Networks and Machine Learning

Review: theories in SMT

- Linear (real) arithmetic
 - $4x - 3y + 6z \leq 10, x + y - z \leq 1;$
- Nonlinear real arithmetic
 - $4x^2 + 6y - 9z^3 \leq 5$
- Bit vectors
- Arrays
 - $x'[i] = x[i] + 1$
- 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 $\Sigma_F := \{0, 1, 2, 3, \dots, -\}, \Sigma_P := \{<, \leq, =, >, \geq\}, V := \{x_i\}$
 - $x_1 - x_2 \geq k$, where $\geq \in \{<, \leq, =, >, \geq\}$

Example decision procedure 1: 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:

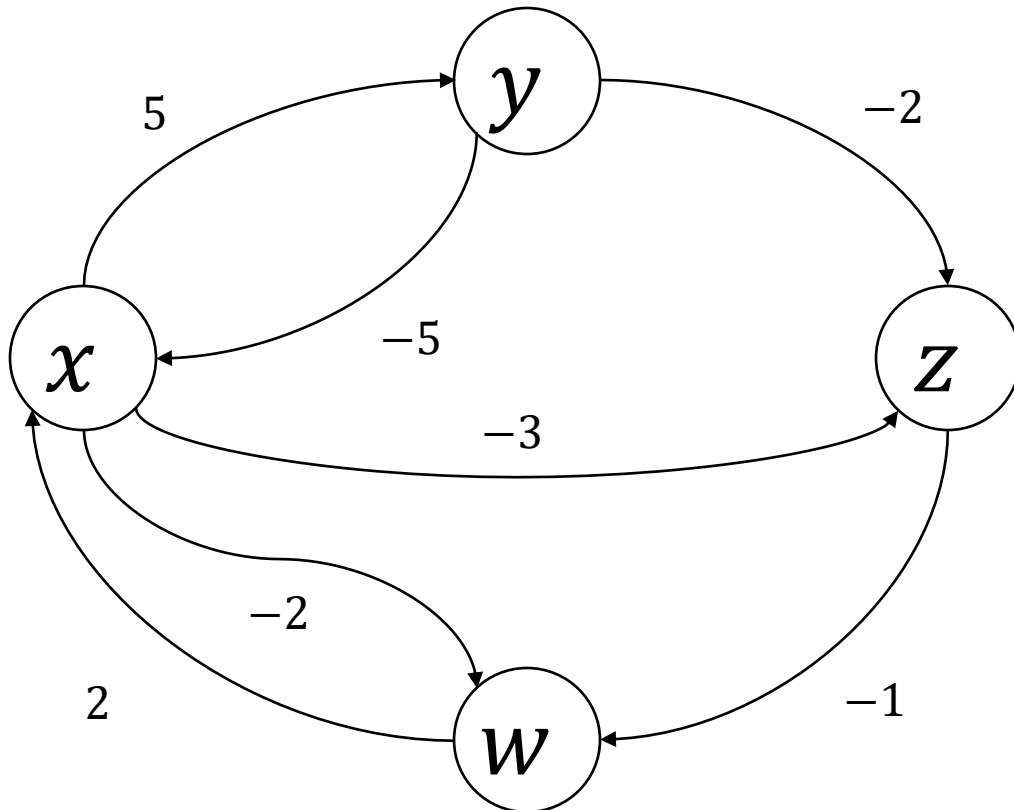
Convert each literal (AF) to $x_1 - x_2 \leq c$ form:

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

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

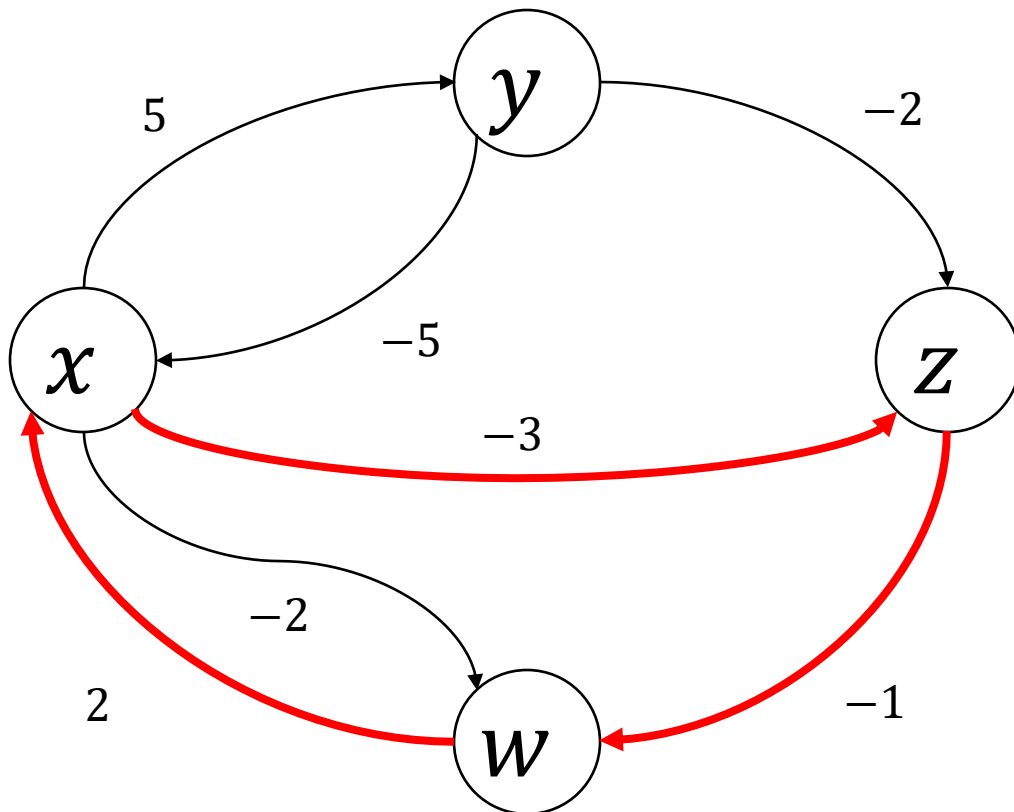
$$\begin{aligned}\phi' = & (x - y \leq 5) \wedge (y - x \leq -5) \\ & \wedge (y - z \leq -2) \wedge \\ & (x - z \leq -3) \wedge \\ & (w - x \leq 2) \wedge (x - w \leq -2) \\ & (z - w \leq -1)\end{aligned}$$

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

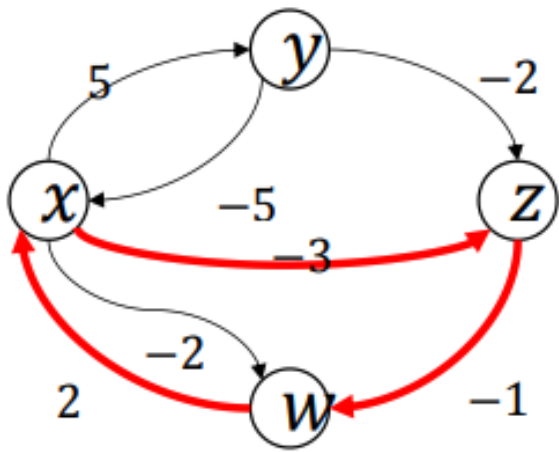


$$\begin{aligned} \phi' = & (x - y \leq 5) \wedge (y - x \leq -5) \\ & \wedge (y - z \leq -2) \wedge \\ & (x - z \leq -3) \wedge \\ & (w - x \leq 2) \wedge (x - w \leq -2) \\ & (z - w \leq -1) \end{aligned}$$

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



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



Proposition. ϕ is satisfiable iff G_ϕ is negative cycle free.

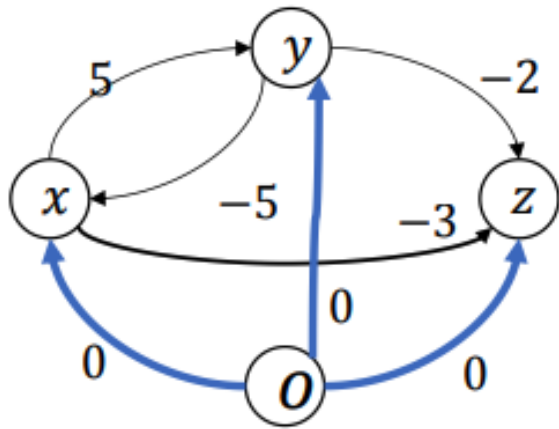
Proof. (\Leftarrow) If there is a negative cycle then there exists

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

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

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

Consider additional vertex o with $o \rightarrow^0 v$ 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

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

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

$$dist(o, y) > dist(x, y) + dist(o, x) \text{ [because } x \rightarrow^k y \text{]}$$

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

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

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

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

Decision Procedure for Difference Logic(DL)

Satisfiability check for difference procedure of DL can be performed using Bellman-Ford algorithm in time $O(|V|.|E|)$

Inconsistency/unsatisfiability explanations are negative cycles

Amenable to incremental check

DPLL^T:DPLL modulo theories

How can we extend DPLL to handle formulas over other theories like

- Difference Logic (DL)
- Uninterpreted functions (UF)
- Linear Real Arithmetic (LRA)

Idea: Start with a Boolean abstraction (or skeleton) and incrementally add more theory information using decision procedure until we can conclusively say SAT or UNSAT

DPLL^T:DPLL modulo theories

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

UF formula $\phi \equiv (a = b) \wedge (b = c) \wedge ((d = e) \vee (e \neq a)) \wedge (F(a) \neq F(c))$

abstract $\phi \equiv x_1 \wedge x_2 \wedge (x_2 \vee \neg x_4) \wedge \neg x_5$

LRA formula $F \equiv (x \leq 0 \vee x \leq 10) \wedge (\neg x \leq 0)$

abstract $F^B \equiv (p \vee q) \wedge (\neg p)$

Abstraction because information is lost: $x > 10 \Rightarrow x > 0$, *i. e.*, $\neg q \Rightarrow \neg p$ not in F^B

Notation: $(F^B)^T$ maps F^B back to theory T, *i.e.*, $(F^B)^T = F$

Proposition. If F^B is UNSAT then F is UNSAT, but the converse does not hold.

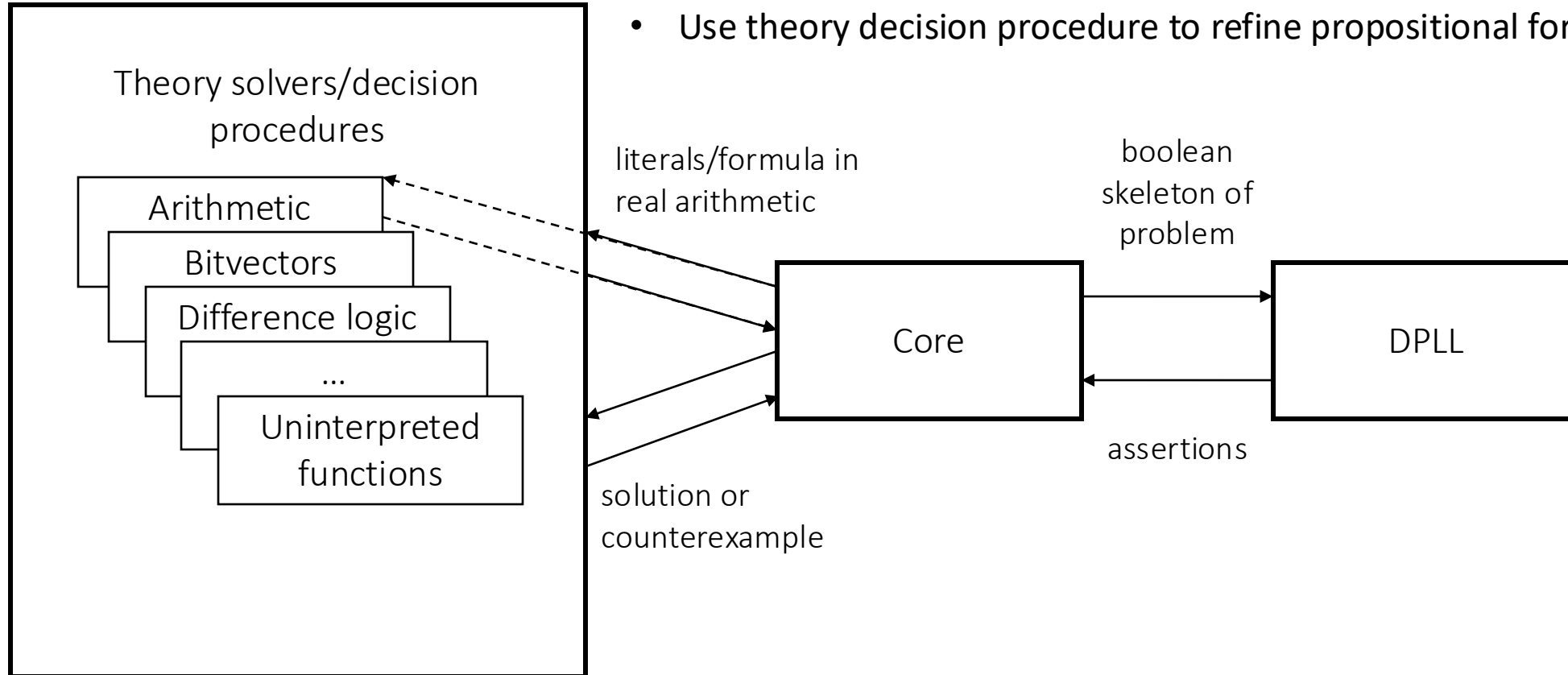
Example. $F_1 \equiv (x \leq 0 \wedge x \geq 10)$ is UNSAT, however $F^B \equiv p \wedge q$ is SAT

Return to SMT

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

Several approaches, lazy approach:

- Abstract ϕ to propositional form $\phi \equiv x_1 \wedge (\neg x_2 \vee x_3) \wedge x_4$
- Feed to DPLL
- Use theory decision procedure to refine propositional formula a guide SAT



DPLL for SMT

Input: A formula F in CNF form over theory T

Output: $I \models F$ or UNSAT

Let F^B be the abstraction of F

while true do

if DPLL(F^B) is unsat **then return** UNSAT

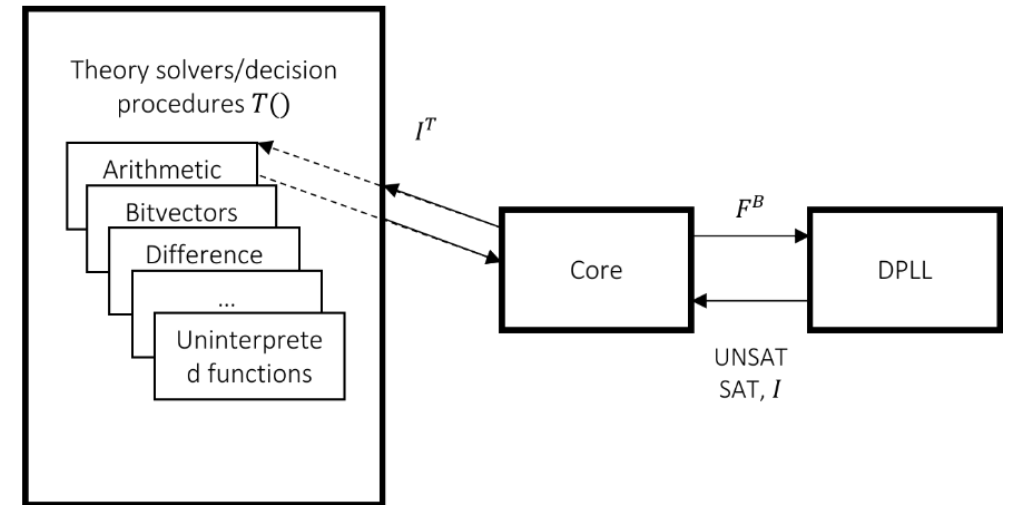
else

 Let I be the model returned by DPLL

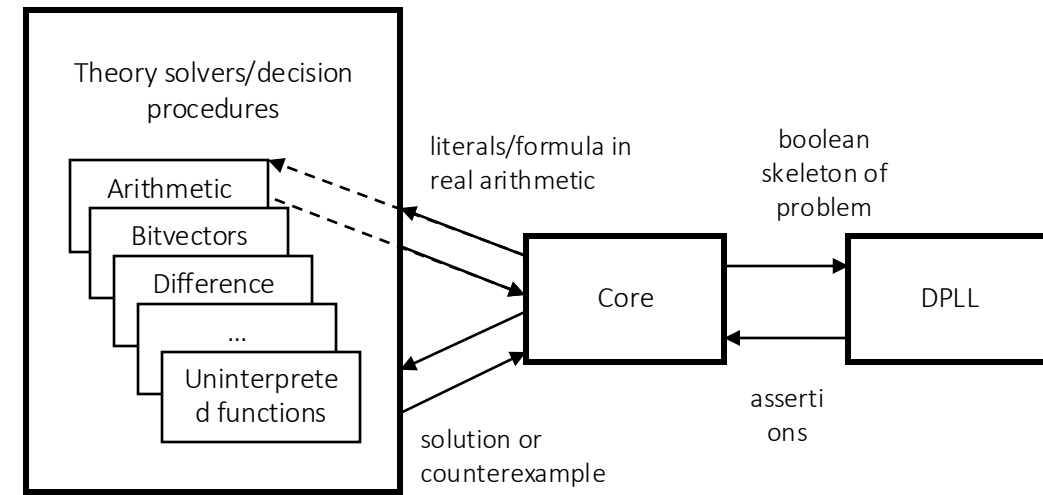
 Assume I is represented as a formula

if $T(I^T)$ is sat **then return** SAT and the model returned by $T()$

else $F^B := F^B \wedge \neg I$



$$\phi \equiv \underbrace{g(a) = c}_1 \wedge \underbrace{f(g(a)) \neq f(c)}_{\bar{2}} \vee \underbrace{g(a) = d}_3 \wedge \underbrace{c \neq d}_{\bar{4}}$$



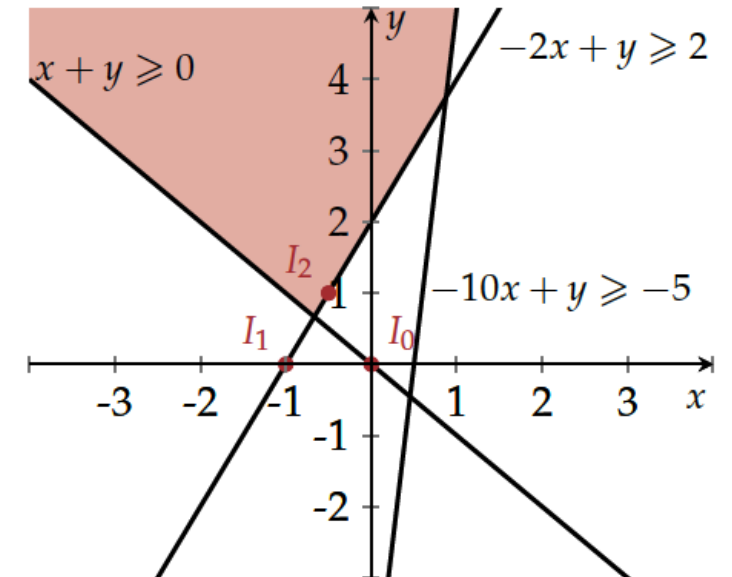
- send $\{1, \bar{2} \vee 3, \bar{4}\}$ to DPLL
- returns model $\{1, \bar{2}, \bar{4}\}$
- UF solver concretizes to $g(a) = c, f(g(a)) \neq f(c), c \neq d$
- checks this as UNSAT
- send $\{1, \bar{2} \vee 3, \bar{4}, \bar{1} \vee 2 \vee 4\}$ to DPLL
- returns model $\{1, 2, 3, \bar{4}\}$
- UF solver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

DPLL^T: DPLL modulo theories

How can we extend DPLL to handle formulas over other theories like

- Difference Logic (DL)
- Uninterpreted functions (UFs)
- **Linear Real Arithmetic (LRA)**

$$(x + y \geq 0) \wedge (-2x + y \geq 2) \wedge (-10x + y \geq -5)$$



Linear Programming Setup

$$\begin{aligned} \max & c_1x_1 + c_2x_2 + \dots c_nx_n \\ \text{such that} & a_{11}x_1 + a_{12}x_2 + \dots a_{1n}x_n \leq b_1 \\ & a_{21}x_1 + a_{22}x_2 + \dots a_{2n}x_n \leq b_2 \\ & \dots \\ & a_{m1}x_1 + a_{m2}x_2 + \dots a_{mn}x_n \leq b_m \end{aligned}$$

General LP problem ($A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m, c \in \mathbb{R}^n$)

$$\max_{x \in \mathbb{R}^n} c^T x \text{ such that } Ax \leq b, x \in \mathbb{R}^n$$

Feasible set: polyhedron $\{x \mid Ax \leq b\}$.

Fundamental fact: if an optimal solution exists, it occurs at a vertex of this polyhedron.

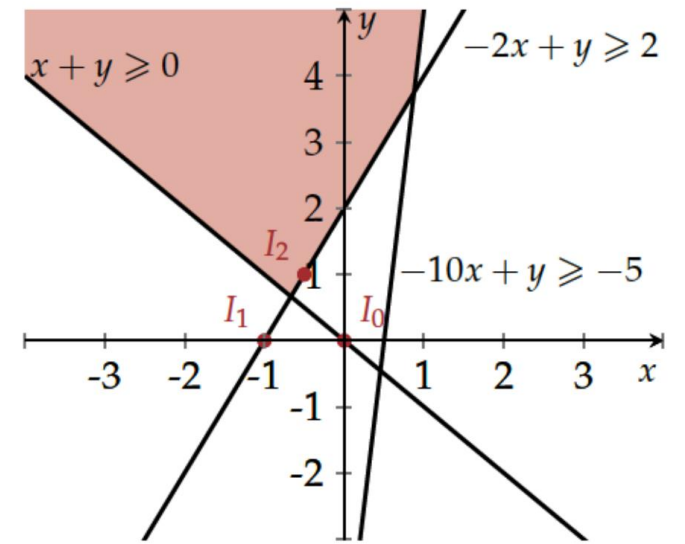
Typically, $m < n$ (fewer constraints than variables).

If $m = n$, the system has a unique solution (if nonsingular) — a single vertex.

If $m > n$, the system is usually overdetermined; feasible solutions may not exist unless rows are redundant.

Standard Form (used by Simplex) obtained by introducing slack variables.

$$\max_{x \in \mathbb{R}^n} c^T x \text{ such that } Ax = b, x \geq 0$$



Decision Procedure for Linear Real Arithmetic

Input: $F \equiv \bigwedge_{i=1}^n \sum_{j=1}^m c_{ij} x_j \leq b_i$ where $c_{ij}, b_i \in \mathbb{R}$

Output: \exists a model $\mathbf{x} \in \mathbb{R}^m$ such that $\mathbf{x} \models F$?

Solution based on Simplex Algorithm [Dantzig 1947]

Simplex solves

$\max \sum_{j=1}^m a_j x_j$ subject to

$$\bigwedge_{i=1}^n \sum_{j=1}^m c_{ij} x_j \leq b_i$$

Our focus will be on finding any solution $\mathbf{x} \in \mathbb{R}^m$ that satisfies F



George B. Dantzig

Decision Procedure for Linear Real Arithmetic

Input: $F \equiv \bigwedge_{i=1}^n \sum_{j=1}^m c_{ij} x_j \leq b_i$ where $c_{ij}, b_i \in \mathbb{R}$

Output: \exists a model $\mathbf{x} \in \mathbb{R}^m$ such that $\mathbf{x} \models F$?

Simplex expects F to be expressed in the Simplex form, which is a **conjunction** of

- Linear equalities: $\sum_{i=1}^m c_i x_i = 0$
- Bounds: $l_i \leq x_i \leq u_i$

Transforming to Simplex Form

Consider the i^{th} inequality in F : $\sum_{j=1}^m c_{ij}x_j \leq b_i$

Rewrite this as:

$$s_i = \sum_{j=1}^m c_{ij}x_j \wedge s_i \leq b_i$$

s_i is called a ***slack variable***

Putting together all the rewritten conjuncts we get F_S

Proposition.

1. Any model of F_S is a model of F , disregarding the assignments to the slack variables.
2. If F_S is UNSAT then F is UNSAT.

The decision procedure: simplex algorithm

Idea: Simultaneously try to find a model or a proof of UNSAT

Start with some *model (or valuation)* that satisfies all **linear equalities** (say, $x_i = 0, \forall i$), but may violate **inequality bounds**

In each iteration, pick a bound that is not satisfied and modify the model to satisfy the bound

OR

discover that the formula is UNSAT

Variable naming and ordering for Simplex

The input formula F_S (after rewriting) has two types of variables

- **Basic variables** appear on the **LHS** of **one** equality; initially these are the *slack variables*
- **Non-basic variables** all others

$$\begin{aligned} s_1 &= x + y \\ s_2 &= -2x + y \\ s_3 &= -10x + y \\ s_1 &\geq 0 \\ s_2 &\geq 2 \\ s_3 &\geq -5 \\ -\infty &\leq x, y \leq \infty \end{aligned}$$

$$\begin{aligned} x &= -0.5s_2 + 0.5y \\ s_1 &= -0.5s_2 + 1.5y \\ s_3 &= 5s_2 - 4y \\ s_1 &\geq 0 \\ s_2 &\geq 2 \\ s_3 &\geq -5 \\ -\infty &\leq x, y \leq \infty \end{aligned}$$

Variable naming and ordering for Simplex

The input formula F_S (after rewriting) has two types of variables

- **Basic variables** appear on the **LHS of one equality**; initially these are the *slack variables*
- **Non-basic variables** all others

We fix an *arbitrary total ordering* on variables x_1, \dots, x_n

For a basic variable x_i and non-basic variable x_j we denote by c_{ij} the coefficient of x_j in the definition of x_i , i.e.,

$$x_i = \dots + c_{ij} x_j + \dots$$

Basic variables are given in terms of a linear equality on the non-basic variables

The upper and lower bounds of x_i are called u_i and l_i (possibly $\infty, -\infty$)

Pivoting: switch basic and non-basic variables

The **pivoting operation** changes one non-basic variable to a basic variable (we say this variable is “entering”), while one other basic variable is changed to non-basic (we say this variable is “leaving”)

$$\begin{aligned} s_1 &= x + y \\ s_2 &= -2x + y \\ s_3 &= -10x + y \\ s_1 &\geq 0 \\ s_2 &\geq 2 \\ s_3 &\geq -5 \\ -\infty &\leq x, y \leq \infty \end{aligned}$$

$$\begin{aligned} x &= -0.5s_2 + 0.5y \\ s_1 &= -0.5s_2 + 1.5y \\ s_3 &= 5s_2 - 4y \\ s_1 &\geq 0 \\ s_2 &\geq 2 \\ s_3 &\geq -5 \\ -\infty &\leq x, y \leq \infty \end{aligned}$$

Pivoting: switch basic and non-basic variables

The pivoting operation change one non-basic variable to a basic variable (we say this variable is “entering”), while one other basic variable is changed to non-basic (we say this variable is “leaving”)

pluggin new definition of x

$$\begin{array}{l} s_1 = x + y \\ s_2 = -2x + y \\ s_3 = -10x + y \\ s_1 \geq 0 \\ s_2 \geq 2 \\ s_3 \geq -5 \\ -\infty \leq x \leq \infty \end{array} \quad \begin{array}{l} \longrightarrow \\ x = -0.5s_2 + 0.5y \\ \longrightarrow \end{array} \quad \begin{array}{l} x = -0.5s_2 + 0.5y \\ s_1 = -0.5s_2 + 1.5y \\ s_3 = 5s_2 - 4y \\ s_1 \geq 0 \\ s_2 \geq 2 \\ s_3 \geq -5 \\ -\infty \leq x \leq \infty \end{array}$$

pluggin new definition of x

Pivoting: switch basic and non-basic variables

The pivoting operation change one non-basic variable to a basic variable (we say this variable is “entering”), while one other basic variable is changed to non-basic (we say this variable is “leaving”)

$$\begin{aligned} s_1 &= x + y \\ s_2 &= -2x + y \\ s_3 &= -10x + y \\ s_1 &\geq 0 \\ s_2 &\geq 2 \\ s_3 &\geq -5 \\ -\infty &\leq x \leq \infty \end{aligned}$$



$$x = -0.5s_2 + 0.5y$$

$$\begin{aligned} x &= -0.5s_2 + 0.5y \\ s_1 &= -0.5s_2 + 1.5y \\ s_3 &= 5s_2 - 4y \\ s_1 &\geq 0 \\ s_2 &\geq 2 \\ s_3 &\geq -5 \\ -\infty &\leq x \leq \infty \end{aligned}$$

$$x_i = \sum_{k \in N}^m c_{ik} x_k, j \in N$$

Pivoting x_i and x_j rewrites x_j as basic variable

$$x_i = c_{ij} x_j + \sum_{k \in N \setminus \{j\}}^m c_{ik} x_k$$

$$x_j = \frac{x_i}{c_{ij}} - \sum_{k \in N \setminus \{j\}}^m \frac{c_{ik}}{c_{ij}} x_k$$

Simplex (Formal)

The algorithm maintains two invariants

(1) The (basic variables in the) model \mathbf{x} always satisfy the **equalities**; bounds may be violated.

Why is this invariant satisfied by our initialization of all 0s?

(2) The **bounds** of all **non-basic** variables are all satisfied.

Why is this invariant satisfied by our initialization?

$$F \equiv \bigwedge_{i=1}^n \sum_{j=1}^m c_{ij} x_j \leq b_i \text{ where } c_{ij}, b_i \in \mathbb{R}$$



Linear equalities: $\sum_{i=1}^m c_i x_i = 0$

Bounds: $l_i \leq x_i \leq u_i$ (only for slack variables)

Transforming to Simplex Form

The algorithm maintains two invariants (1) and (2). Given the initialization:

$$\sum_{j=1}^m c_{ij}x_j \leq b_i$$

Rewrite this as:

$$s_i = \sum_{j=1}^m c_{ij}x_j \wedge s_i \leq b_i$$

s_i is called a *slack variable*

(1) Setting all x_j and s_i to 0 satisfies all equality constraints (but may violating bounds on s_i)

(2) Initially, all slack variables are basic variables. Non-basic variables have no bounds.

Simplex Algorithm: DP for LRA

Input: A formula F_S in Simplex form

Output: $x \models F_S$ or UNSAT

$x := \langle x_i \mapsto 0 \rangle$

while true do

if $x \models F_S$ **then return** x

Let x_i be the first basic variable s.t. $x_i < l_i$ or $x_i > u_i$

if $x_i < l_i$ **then**

Let x_j be the first non-basic variable s.t.

$(x_j < u_j \wedge c_{ij} > 0) \vee (x_j > l_j \wedge c_{ij} < 0)$

If no such x_j exists **then return** UNSAT “blue” conditions

$x_j := x_j + \frac{l_i - x_i}{c_{ij}}; x_i := l_i$

else Let x_j be the first non-basic variable s.t.

$(x_j > l_j \wedge c_{ij} > 0) \vee (x_j < u_j \wedge c_{ij} < 0)$

If no such x_j exists **then return** UNSAT

$x_j := x_j + \frac{u_i - x_i}{c_{ij}}; x_i := u_i$

Pivot x_i and x_j ; update x_i, x_j , and all basic variables

x_i currently violates its bounds

Finding a suitable variable we can “adjust” in order to satisfy the bound of x_i

Pivoting procedure: given

$$x_i = \sum_{k \in N} c_{ik} x_k, j \in N$$

Pivoting x_i and x_j rewrites x_j as basic variable

$$x_i = c_{ij} x_j + \sum_{k \in N \setminus \{j\}} c_{ik} x_k$$

$$x_j = \frac{x_i}{c_{ij}} - \sum_{k \in N \setminus \{j\}} \frac{c_{ik}}{c_{ij}} x_k$$

Example

$$x + y \geq 0$$

$$-2x + y \geq 2$$

$$-10x + y \geq -5$$

Rewritten in Simplex form

$$s_1 = x + y$$

$$s_2 = -2x + y$$

$$s_3 = -10x + y$$

$$s_1 \geq 0$$

$$s_2 \geq 2$$

$$s_3 \geq -5$$

Example continued

Variable ordering

x, y, s_1, s_2, s_3

Initialization $\mathbf{x}_0 = \langle x \mapsto 0, y \mapsto 0, s_1 \mapsto 0, s_2 \mapsto 0, s_3 \mapsto 0 \rangle$

\mathbf{x}_0 satisfies equalities, bounds of s_1, s_3 are satisfied

Pick the first variable x to fix the bound of s_2

Since upper and lower bounds of x are ∞ and $-\infty$, x can easily satisfy the “blue” conditions

To increase s_2 to 2 and satisfy its lowerbound we decrease x to -1

$$\mathbf{x}_1 = \langle x \mapsto 0 + \frac{2 - 0}{-2} = -1, y \mapsto 0, s_1 \mapsto -1, s_2 \mapsto 2, s_3 \mapsto 10 \rangle$$

Pivot s_2 with x

Now x becomes a basic variable, s_2 non-basic

$$\begin{aligned} s_1 &= x + y \\ s_2 &= -2x + y \\ s_3 &= -10x + y \\ s_1 &\geq 0 \\ s_2 &\geq 2 \\ s_3 &\geq -5 \\ -\infty &\leq x, y \leq \infty \end{aligned}$$

$$\begin{aligned} x &= -0.5s_2 + 0.5y \\ s_1 &= -0.5s_2 + 1.5y \\ s_3 &= 5s_2 - 4y \\ s_1 &\geq 0 \\ s_2 &\geq 2 \\ s_3 &\geq -5 \\ -\infty &\leq x, y \leq \infty \end{aligned}$$

Example continued

$$\mathbf{x}_1 = \langle x \mapsto -1, y \mapsto 0, s_1 \mapsto -1, s_2 \mapsto 2, s_3 \mapsto 10 \rangle$$

All equalities are still satisfied (invariant)

The only basic variable not satisfying its bounds is now s_1

The first non-basic variable we can tweak is y

Setting $s_1 \mapsto 0$ to satisfy the lowerbound of s_1 we get

$$\mathbf{x}_2 = \langle x \mapsto -2/3, y \mapsto 0 + \frac{0 - (-1)}{1.5} = 2/3, \\ s_1 \mapsto 0, s_2 \mapsto 2, s_3 \mapsto 22/3 \rangle$$

Pivot s_1 with y

$$s_1 = -0.5s_2 + 1.5y \quad \longrightarrow \quad y = \frac{2}{3}s_1 + \frac{1}{3}s_2$$

$$\mathbf{x}_2 \models F_S$$

$$\begin{aligned} x &= -0.5s_2 + 0.5y \\ s_1 &= -0.5s_2 + 1.5y \\ s_3 &= 5s_2 - 4y \\ s_1 &\geq 0 \\ s_2 &\geq 2 \\ s_3 &\geq -5 \\ -\infty &\leq x, y \leq \infty \end{aligned}$$

$$\begin{aligned} y &= \frac{2}{3}s_1 + \frac{1}{3}s_2 \\ x &= \frac{1}{3}s_1 - \frac{1}{3}s_2 \\ s_3 &= -\frac{8}{3}s_1 + \frac{11}{3}s_2 \\ s_2 &\geq 2 \\ s_1 &\geq 0 \\ s_3 &\geq -5 \\ -\infty &\leq x, y \leq \infty \end{aligned}$$

Why is simplex correct?

- Why does it terminate?

Because we always look for the first variable violating the bounds. There is a property (Bland's rule) that ensures that we never revisit the same set of basic and non-basic variables.

- Why does it give the right answer (sound)?

- If it returns x does it satisfy $x \models F$?

This follows from the condition before **return** x

- If it returns UNSAT is F really unsatisfiable?

For proofs, check [Dutertre, B., de Moura, L.: Integrating Simplex with DPLL\(T\). Technical report, CSL-06-01, SRI International \(2006\)](#)

Unsatisfiable example

$$s_1 = x + y$$

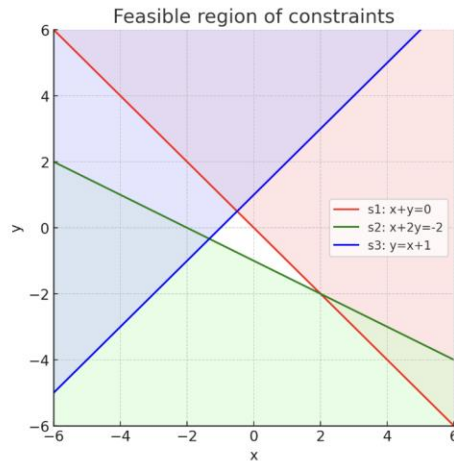
$$s_2 = -x - 2y$$

$$s_3 = -x + y$$

$$s_1 \geq 0$$

$$s_2 \geq 2$$

$$s_3 \geq 1$$



The red line is $s_1 : x + y = 0$, feasible side is above/right ($x + y \geq 0$).

The green line is $s_2 : x + 2y = -2$, feasible side is below/left ($x + 2y \leq -2$).

The blue line is $s_3 : y = x + 1$, feasible side is above ($y \geq x + 1$).

0) Initialize non-basics at bounds; compute basics

$$x_0 = \langle x \mapsto 0, y \mapsto 0, s_1 \mapsto 0, s_2 \mapsto 0, s_3 \mapsto 0 \rangle$$

$$\text{Violations: } s_2 < 2, s_3 < 1$$

1) Fix s_2 (enter x , leave s_2)

Need $s_2 = -x - 2y$ to reach 2 $\Rightarrow x \leftarrow -2$.

Pivot $s_2 \leftrightarrow x$: $x = -s_2 - 2y, s_1 = -s_2 - y, s_3 = s_2 + 3y$.

$$x_1 = \langle x \mapsto -2, y \mapsto 0, s_1 \mapsto -2, s_2 \mapsto 2, s_3 \mapsto 2 \rangle$$

Violation: $s_1 < 0$. First eligible non-basic to raise $s_1 = -s_2 - y$

2) Fix s_1 (enter y , leave s_1) Make s_1 hit 0: $y \leftarrow -2$

Pivot $s_1 \leftrightarrow y$: $x = s_2 + 2s_1, y = -s_2 - s_1, s_3 = -3s_1 - 2s_2$

$$x_2 = \langle x \mapsto 2, y \mapsto -2, s_1 \mapsto 0, s_2 \mapsto 2, s_3 \mapsto -4 \rangle$$

Violation: $s_3 < 1$.

3) UNSAT test for s_3

With non-basics fixed at their bounds $s_1 \geq 0, s_2 \geq 2$, we have

$$s_3 = -3s_1 - 2s_2 \leq -2 \cdot 2 = -4 < 1.$$

To increase s_3 , we'd need to **decrease** s_1 or s_2 , but they are at their lower bounds. No eligible non-basic x_j satisfies the selection rule \Rightarrow **UNSAT**.

Simplex Algorithm: DP for LRA

Input: A formula F_S in Simplex form

Output: $x \models F_S$ or UNSAT

$x := \langle x_i \mapsto 0 \rangle$

while true do

if $x \models F_S$ **then return** x

Let x_i be the first basic variable s.t. $x_i < l_i$ or $x_i > u_i$

if $x_i < l_i$ **then**

Let x_j be the first non-basic variable s.t.

$$(x_j < u_j \wedge c_{ij} > 0) \vee (x_j > l_j \wedge c_{ij} < 0)$$

If no such x_j exists **then return** UNSAT

$$x_j := x_j + \frac{l_i - x_i}{c_{ij}}; x_i := l_i$$

else Let x_j be the first non-basic variable s.t.

$$(x_j > l_j \wedge c_{ij} > 0) \vee (x_j < u_j \wedge c_{ij} < 0)$$

If no such x_j exists **then return** UNSAT

$$x_j := x_j + \frac{u_i - x_i}{c_{ij}}; x_i := u_i$$

Pivot x_i and x_j ; update x_i , x_j , and all basic variables

i.e., find a non-basic variable that is **not** set to their extreme value u_j or l_j according to the sign of c_{ij}

If the non-basic variable x_j has been set to its extreme, we cannot further change it to fix x_i

Summary and Takeaways

- Satisfiability modulo theory solvers use theory solvers and DPLL to check satisfiability of formulas in other theories
 - DPLL takes care of disjunctions
 - Theory solvers take care of conjunctions
- Simplex or more generally Linear programming (LP) solvers is a theory solver for linear real arithmetic
 - Simplex algorithm solves LP by incrementally fixing the bounds of basic variables