universität
innsbruck

SAT and SMT Solving

Sarah Winkler

KRDB
Department of Computer Science
Free University of Bozen-Bolzano

lecture 7
WS 2022

Outline

- Summary of Last Week
- Linear Arithmetic
- Simplex Algorithm

Deciding the Theory of Equality

Definition

- equality logic formula φ_{EQ} is set of equations and inequalities between variables
- write $\mathcal{V}ar(\varphi_{\mathsf{EQ}})$ for set of variables occurring in φ_{EQ}

Definition

equality graph for φ_{EQ} is undirected graph $(V, E_=, E_\neq)$ with two kinds of edges

- nodes $V = \mathcal{V}ar(\varphi_{\mathsf{EQ}})$
- $(x, y) \in E_=$ iff $x = y$ in φ_{EQ} — equality edge
- $(x, y) \in E_\neq$ iff $x \neq y$ in φ_{EQ} — inequality edge

Definition (Contradictory cycle)

contradictory cycle is simple cycle in equality graph with one $E_\neq$ edge
and all others $E_=$ edges

Theorem

φ_{EQ} is satisfiable iff its equality graph has no contradictory cycle

Deciding the Theory of Equality with Uninterpreted Functions

Remark

- can assume that $=$ is the only predicate in φ
- can replace variables by constants (Skolemization)

Congruence Closure

Input: set of equations E and equation $s = t$ (without variables, only constants)

Output: $s = t$ is *implied* ($E \vDash_{EUF} s = t$) or *not implied* ($E \nvDash_{EUF} s = t$)

1. build congruence classes
   (a) put different subterms of terms in $E \cup \{s \approx t\}$ in separate sets
   (b) merge sets $\{\ldots, t_1, \ldots\}$ and $\{\ldots, t_2, \ldots\}$ for all $t_1 \approx t_2$ in E
   (c) merge sets $\{\ldots, f(t_1, \ldots, t_n), \ldots\}$ and $\{\ldots, f(u_1, \ldots, u_n), \ldots\}$
   if t_i and u_i belong to same set for all $1 \leqslant i \leqslant n$, repeatedly
2. if s and t belong to same set then return *implied* else return *not implied*

Satisfiability Check for EUF

$$(\bigwedge P) \wedge (\bigwedge N) \text{ unsatisfiable} \iff \exists\, s \neq t \text{ in } \widehat{N} \text{ such that } \bigwedge \widehat{P} \vDash_{EUF} s = t$$

Correctness of DPLL(T)

Definition (DPLL(T) systems)

- basic system $\mathcal{B}$: unit propagate, decide, fail, T-backjump, T-propagate
- full system $\mathcal{F}$: $\mathcal{B}$ plus T-learn, T-forget, and restart

Theorem (Correctness)

For derivation with final state S_n:

$$\| \; F \quad \Longrightarrow_{\mathcal{F}} \quad S_1 \quad \Longrightarrow_{\mathcal{F}} \quad S_2 \quad \Longrightarrow_{\mathcal{F}} \quad \dots \quad \Longrightarrow_{\mathcal{F}} \quad S_n$$

- *if* $S_n = \text{FailState}$ *then* F *is* T*-unsatisfiable*
- *if* $S_n = M \| F'$ *and* M *is* T*-consistent then* F *is* T*-satisfiable and* $M \vDash_T F$

Theorem (Termination)

Γ: $\quad \| \; F \Longrightarrow^*_{\mathcal{F}} S_0 \Longrightarrow^*_{\mathcal{F}} S_1 \Longrightarrow^*_{\mathcal{F}} \dots$ *is finite if*

- *there is no infinite sub-derivation of only* T*-learn and* T*-forget steps, and*
- *for every sub-derivation* $S_i \overset{restart}{\Longrightarrow}_{\mathcal{F}} S_{i+1} \Longrightarrow^*_{\mathcal{F}} S_j \overset{restart}{\Longrightarrow}_{\mathcal{F}} S_{j+1} \Longrightarrow^*_{\mathcal{F}} S_k$
  - *there are more* $\mathcal{B}$*-steps in* $S_j \Longrightarrow^*_{\mathcal{F}} S_k$ *than in* $S_i \Longrightarrow^*_{\mathcal{F}} S_j$*, or*
  - *a clause is learned in* $S_j \Longrightarrow^*_{\mathcal{F}} S_k$ *that is never forgotten in* Γ

Outline

- Summary of Last Week
- Linear Arithmetic
- Simplex Algorithm

Definition (Theory of Linear Arithmetic over $\mathbb{Z}$ (LIA))

- signature
  - binary predicates $<$ and $=$
  - binary function $+$, unary function $-$, constants 0 and 1
- axioms

$$\forall x.\ (x = x) \qquad \forall x\, y.\ (x = y \rightarrow y = x) \qquad \forall x\, y\, z.\ (x = y \wedge y = z \rightarrow x = z)$$

$$\forall x.\ (x + 0 = x) \qquad \forall x\, y.\ (x + y = y + x) \qquad \forall x\, y\, z.\ (x + (y + z) = (x + y) + z)$$

$$\forall x.\ \neg(x < x) \qquad \forall x\, y.\ (x < y \vee y < x \vee x = y) \qquad \forall x\, y\, z.\ (x < y \wedge y < z \rightarrow x < z)$$

$$0 < 1 \qquad \forall x.\ (x + (-x) = 0) \qquad \forall x\, y\, z.\ (x < y \rightarrow x + z < y + z)$$

$$\forall x.\ \neg(0 < x \wedge x < 1) \qquad \forall x\, \exists y.\ \bigvee_{0 \leqslant r < n} x = ny + r$$

Theorem

- *$\mathbb{Z}$ with usual interpretations is model of LIA*
- *and it is unique model up to elementary equivalence*

i.e., same formulas hold

Example

- $x + y + z = 1 + 1 \wedge y < z \wedge -1 < y$ LIA-satisfiable, $v(x) = v(y) = 0$, $v(z) = 2$
- $x < 1 \wedge 1 < x + x$ LIA-unsatisfiable

Remarks

- LIA is also known as Presburger arithmetic: different but equivalent axiomatizations exist
- LIA has no multiplication: $x \cdot y$ and x^2 for variables x, y is not allowed

Syntactic Sugar

$\leqslant$	binary predicate	$s \leqslant t$ abbreviates $\neg(t < s)$
$>$ and $\geqslant$	binary predicates	use $s > t$ for $t < s$ and $s \geqslant t$ for $t \leqslant s$
$n\,\cdot$	unary functions $\forall n \in \mathbb{Z}$	$n \cdot t$ means $\underbrace{t + \ldots + t}_{n}$ if $n \geqslant 0$ $\underbrace{(-t) + \ldots + (-t)}_{n}$ if $n < 0$
n	constants $\forall n \in \mathbb{Z}$	n abbreviates $n \cdot 1$

Example (LIA with syntactic sugar)

- $x + y + z = 2 \ \wedge\ z > y \ \wedge\ y \geqslant 0$
- $x < 1 \ \wedge\ 2x > 1$
- $7x = 41$

Theorem

LIA is decidable and NP-complete

Definition (Theory of Linear Arithmetic over $\mathbb{Q}$ (LRA))

- signature
  - binary predicates $<$ and $=$
  - binary function $+$, unary function $-$, constants 0 and 1
  - unary (division) functions $(\_/n)$ for all $n > 1$
- axioms

$\forall x.\ (x = x)$	$\forall x\, y.\ (x = y \ \rightarrow\ y = x)$	$\forall x\, y\, z.\ (x = y \wedge y = z \ \rightarrow\ x = z)$
$\forall x.\ (x{+}0 = x)$	$\forall x\, y.\ (x{+}y = y{+}x)$	$\forall x\, y\, z.\ (x{+}(y{+}z) = (x{+}y){+}z)$
$\forall x.\ \neg(x < x)$	$\forall x\, y.\ (x < y \vee y < x \vee x = y)$	$\forall x\, y\, z.\ (x < y \wedge y < z \ \rightarrow\ x < z)$
$0 < 1$	$\forall x.\ (x + (-x) = 0)$	$\forall x\, y\, z.\ (x < y \ \rightarrow\ x{+}z < y{+}z)$
	$\forall x.\ (n \cdot (x/n) = x)$	for all $n > 1$

Theorem

- *$\mathbb{Q}$ with usual interpretations is model of LRA*
- *and it is the single unique model up to elementary equivalence*

Example

- $x{+}y{+}z = 1{+}1 \ \wedge\ y < z \ \wedge\ -1 < y$ LRA-satisfiable, $v(x) = v(y) = 0$, $v(z) = 2$
- $x < 1 \ \wedge\ 1 < x + x$ LRA-satisfiable with $v(x) = \frac{2}{3}$

Syntactic Sugar

use same shorthands as for LIA, plus

- $q\,\cdot$ unary functions $\forall q \in \mathbb{Q}$ $q \cdot t$ abbreviates $m \cdot t / n$ if $q = \frac{m}{n}$
- q constants $\forall q \in \mathbb{Q}$ q abbreviates $q \cdot 1$

Example (LRA with syntactic sugar)

- $\frac{4}{5}x = 2 \wedge \frac{x}{7} = \frac{y}{2} + 1$
- $x < \frac{7}{8} \wedge 2x > \frac{5}{4}$
- $7.5x = 41.2$

Theorem

LRA is decidable in polynomial time

Some History

1826 Fourier and Motzkin (1936) developed elimination algorithm for LRA

- takes doubly exponential time

1947 Dantzig proposed Simplex algorithm to solve optimization problem in LRA:

$$\text{maximize } c(\overline{x}) \qquad \text{such that} \qquad A\overline{x} \leqslant b \text{ and } \overline{x} \geqslant 0$$

for linear objective function c, matrix A, vector b, and vector of variables $\overline{x}$

- runs in exponential time, also known as linear programming

1960 Land and Doig: Branch-And-Bound to get LIA solution from LRA solution

1979 Khachiyan proposed polynomial Simplex based on ellipsoid method

1984 Karmakar proposed polynomial version based on interior points method

2000- SMT solvers use DPLL(T) version to solve satisfiability problem

$$A\overline{x} \leqslant b$$

Outline

- Summary of Last Week
- Linear Arithmetic
- Simplex Algorithm

Aim

build theory solver for linear rational arithmetic (LRA):

decide whether set of linear (in)equalities is satisfiable over $\mathbb{Q}$

Disclaimer: Effects and Side Effects

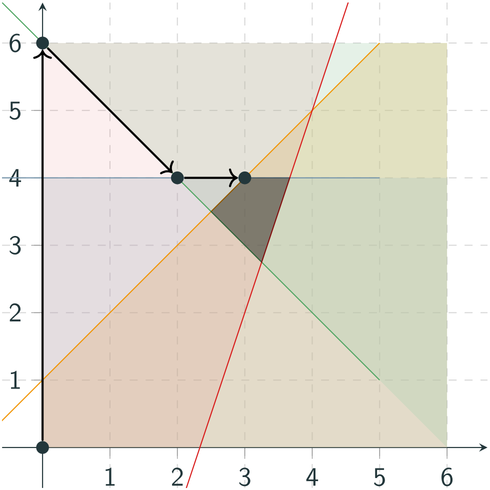
- guaranteed to solve all your real arithmetic problems
- consuming Simplex can cause initial dizzyness
- in some cases solving systems of linear inequalities can become addictive

Simplex, Visually

- constraints

  $$x - y \geqslant -1$$
  $$y \leqslant 4$$
  $$x + y \geqslant 6$$
  $$3x - y \leqslant 7$$

- solution space
- Simplex algorithm: improve assignment in 4 iterations
  - $x = 0,\ y = 0$
  - $x = 0,\ y = 6$
  - $x = 2,\ y = 4$
  - $x = 3,\ y = 4$



Definition (Problem in general form)

- variables $x_1, \dots, x_n$
- m equalities for $a_{ij} \in \mathbb{Q}$

$$\begin{aligned} a_{11}x_1 + \dots a_{1n}x_n &= 0 \\ \dots \\ a_{m1}x_1 + \dots a_{mn}x_n &= 0 \end{aligned}$$

- (optional) lower and upper bounds on variables for $l_i, u_i \in \mathbb{Q}$

$$l_i \leqslant x_i \leqslant u_i$$

no occurrences of $<$, $>$, or $\neq$

Lemma

set of LRA literals where all predicates are $\leqslant$, $\geqslant$, or $=$ can be turned into equisatisfiable general form

Example

$$\begin{aligned} x - y &\geqslant -1 \\ y &\leqslant 4 \\ x + y &\geqslant 6 \\ 3x - y &\leqslant 7 \end{aligned} \quad \Longrightarrow \quad \begin{aligned} -x + y - s_1 &= 0 & s_1 &\leqslant 1 \\ y - s_2 &= 0 & s_2 &\leqslant 4 \\ -x - y - s_3 &= 0 & s_3 &\leqslant -6 \\ 3x - y - s_4 &= 0 & s_4 &\leqslant 7 \end{aligned}$$

slack variables

- s_1, s_2, s_3, s_4 are slack variables, x, y are problem variables

Representation

- represent equalities by $m \times (n+m)$ matrix A such that $A \cdot \begin{pmatrix} \overline{x} \\ \overline{s} \end{pmatrix} = 0$

$$\begin{aligned} -x + y - s_1 &= 0 & s_1 &\leqslant 1 \\ y - s_2 &= 0 & s_2 &\leqslant 4 \\ -x - y - s_3 &= 0 & s_3 &\leqslant -6 \\ 3x - y - s_4 &= 0 & s_4 &\leqslant 7 \end{aligned} \quad \Longrightarrow \quad \begin{pmatrix} -1 & 1 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 & 0 \\ -1 & -1 & 0 & 0 & -1 & 0 \\ 3 & -1 & 0 & 0 & 0 & -1 \end{pmatrix} \quad \begin{aligned} s_1 &\leqslant 1 \\ s_2 &\leqslant 4 \\ s_3 &\leqslant -6 \\ s_4 &\leqslant 7 \end{aligned}$$

- simplified matrix presentation

$$\begin{array}{c} \\ \text{dependent variables} \rightarrow \end{array} \quad \begin{array}{c|cc} & x & y \\ \hline s_1 & -1 & 1 \\ s_2 & 0 & 1 \\ s_3 & -1 & -1 \\ s_4 & 3 & -1 \end{array} \quad \leftarrow \text{independent variables}$$

Notation

- simplified matrix is called tableau
- D is set of dependent (or basic) variables, in tableau listed on the left
- I is set of independent (or non-basic) variables, in tableau on top

DPLL(T) Simplex Algorithm

Input: conjunction of LRA literals φ without $<$, $>$, $\neq$

Output: satisfiable or unsatisfiable

1. transform φ into general form and construct tableau
2. fix order on variables and assign 0 to each variable
3. if all dependent variables satisfy their bounds then return satisfiable
4. otherwise, let $x \in D$ be variable that violates one of its bounds b
5. search for suitable variable $y \in I$ for pivoting with x
   (i.e., look for y whose value can be changed such that x is within b)
6. return unsatisfiable if no such variable exists
7. perform pivot operation on x and y
   (i.e., make x independent and y dependent)

9. improve assignment: set x to b, and update accordingly
10. go to step 3

Example

tableau	bounds	assignment

$$\begin{array}{c} \\ s_3 \\ x \\ y \\ s_4 \end{array} \begin{array}{c} \begin{array}{cc} s_2 & s_1 \end{array} \\ \begin{pmatrix} -2 & 1 \\ 1 & -1 \\ 1 & 0 \\ 2 & -3 \end{pmatrix} \end{array}$$

$s_1 \leqslant 1$
$s_2 \leqslant 4$
$s_3 \leqslant -6$
$s_4 \leqslant 7$

x	y	s_1	s_2	s_3	s_4
3	4	1	4	−7	5

1. Iteration 1
   - s_3 violates its bounds
   - decreasing s_3 requires to increase x or y because $s_3 = -x - y$: both suitable since they have no upper bound
   - pivot s_3 with y:

$$y = -x - s_3 \qquad s_1 = -2x - s_3$$
$$s_2 = -x - s_3 \qquad s_4 = 4x + s_3$$

   - update assignment: set s_3 to violated bound -6 and propagate

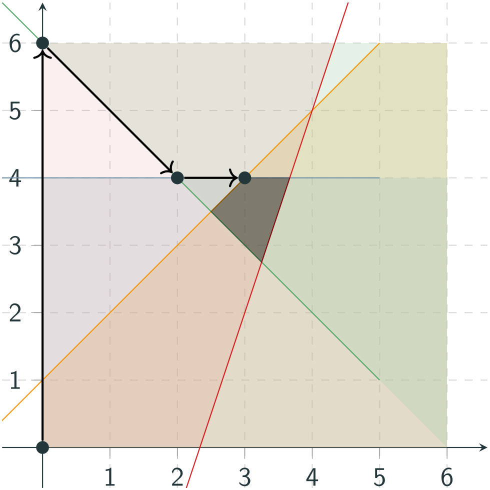
$$s_3 = -6 \qquad y = 6$$
$$s_1 = 6 \qquad s_2 = 6 \qquad s_4 = -6$$

Simplex, Visually

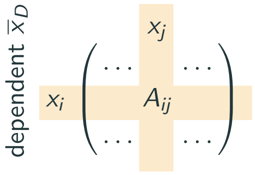
- constraints

  $$x - y \geqslant -1$$
  $$y \leqslant 4$$
  $$x + y \geqslant 6$$
  $$3x - y \leqslant 7$$

- solution space
- Simplex algorithm: improve assignment in 4 iterations
  - $x = 0,\ y = 0$
  - $x = 0,\ y = 6$
  - $x = 2,\ y = 4$
  - $x = 3,\ y = 4$



DPLL(T) Simplex Algorithm



$$A\overline{x}_I = \overline{x}_D \qquad (1)$$

$$-\infty \leqslant l_i \leqslant x_i \leqslant u_i \leqslant +\infty \qquad (2)$$

Invariant

- (1) is satisfied and (2) holds for all independent variables

Pivoting

- swap dependent x_i and independent x_j, so $x_i \in D$ and $x_j \in I$

$$x_i = \sum_{x_k \in I} A_{ik} x_k \quad \Longrightarrow \quad x_j = \underbrace{\frac{1}{A_{ij}}(x_i - \sum_{x_k \in I - \{x_j\}} A_{ik} x_k)}_{t} \qquad (\star)$$

new row

updated other rows

- new tableau A' consists of $(\star)$ and $x_m = A_{mj} t + \sum_{x_k \in I - \{x_j\}} A_{mk} x_k \quad \forall x_m \in D - \{x_i\}$

Update

- assignment of x_i is updated to previously violated bound l_i or u_i,
- assignment of x_k is updated using A' for all $\forall x_m \in D - \{x_i\}$

DPLL(T) Simplex Algorithm

$$A\overline{x}_I = \overline{x}_D \qquad (1)$$

$$-\infty \leqslant l_i \leqslant x_i \leqslant u_i \leqslant +\infty \qquad (2)$$

Suitable pivot variable

- suppose dependent variable x_i violates lower and/or upper bound
- then x_j is suitable for pivoting with x_i if
  - if $x_i < l_i$: $(A_{ij} > 0$ and $x_j < u_j)$ or $(A_{ij} < 0$ and $x_j > l_j)$
  - if $x_i > u_i$: $(A_{ij} > 0$ and $x_j > l_j)$ or $(A_{ij} < 0$ and $x_j < u_j)$

want to increase x_i | need to increase x_j | need to decrease x_j

want to decrease x_i | need to decrease x_j | need to increase x_j

Observation

selecting variables and pivots in unfortunate order may lead to non-termination

Bland's rule

select variable x_i in step 4 and x_j in step 5 such that (x_i, x_j) is minimal with respect to lexicographic extension of order on variables

Lemma

- *Simplex terminates if pivot variables are selected according to Bland's rule*
- *problem is satisfiable iff Simplex returns satisfiable*

How to Deal With Strict Inequalities?

replace in LRA formula φ every strict inequality

$$a_1x_1 + \cdots + a_nx_n < b$$

by non-strict inequality

$$a_1x_1 + \cdots + a_nx_n \leqslant b - \delta$$

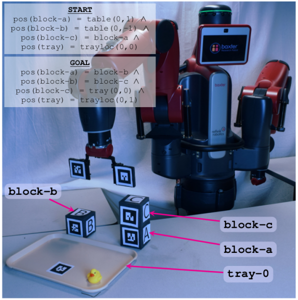
to obtain formula φ_δ in LRA without $<$, and treat δ as variable during Simplex

Lemma

φ *is satisfiable* $\iff$ $\exists$ *rational number* $\delta > 0$ *such that* φ_δ *is satisfiable*

Application: Motion Planning for Robots

- robots needs to plan motions
  to place objects correctly
- instance of *constraint based planning*
- encoding
  - fix number of time slots $t_1, \ldots, t_n$
  - action variable a_i for time t_i encodes which action performed at time t_i (one action per time)
  - actions require precondition and imply postcondition
  - use arithmetic to minimize path



Neil T. Dantam, Zachary K. Kingston, Swarat Chaudhuri, and Lydia E. Kavraki.
Incremental Task and Motion Planning: A Constraint-Based Approach.
In: The International Journal of Robotics Research, 2018.

(Almost) Everything is Better With Arithmetic

LRA and LIA admit more efficient encodings of

- n-queens
- Sudoku
- graph coloring
- Minesweeper
- travelling salesperson
- rabbit problem
- planning problems
- scheduling problems
- component configuration problems
- everything with cardinality constraints
- ...

Bibliography

Bruno Dutertre and Leonardo de Moura.
A Fast Linear-Arithmetic Solver for DPLL(T).
In Proc. of International Conference on Computer Aided Verification, pp. 81–94, 2006.

Bruno Dutertre and Leonardo de Moura
Integrating Simplex with DPLL(T)
Technical Report SRI–CSL–06–01, SRI International, 2006

Test on December 2

- 50 minutes
- open (paper) book: bring arbitrary amount of printed paper, but use no electronic devices
- questions are like homework exercises:
  e.g., DPLL, implication graphs, give minimal unsatisfiable core of formula, equality graphs, congruence closure, DPLL(T), ... (no Simplex)