universität innsbruck

SAT and SMT Solving

Sarah Winkler

KRDB
Department of Computer Science
Free University of Bozen-Bolzano

lecture 8
WS 2022

Outline

- Summary of Last Week
- Cyclic Simplex Example
- Branch and Bound
- Fourier-Motzkin Elimination

Definition (Theory of Linear Arithmetic over C)

- for variables $x_1, \dots, x_n$, formulas built according to grammar

$$\varphi ::= \varphi \wedge \varphi \mid t = t \mid t < t \mid t \leqslant t$$
$$t ::= a_1 x_1 + \cdots + a_n x_n + b \qquad \text{for } a_1, \dots, a_n, b \in \text{in carrier } C$$

- axioms are equality axioms plus calculation rules of arithmetic over C
- solution assigns values in C to $x_1, \dots, x_n$

Definitions

- carrier $\mathbb{Q}$: linear real arithmetic (LRA),
  DPLL(T) simplex algorithm is decision procedure
- carrier $\mathbb{Z}$: linear integer arithmetic (LIA)

DPLL(T) Simplex Algorithm (1)

- linear arithmetic constraint solving over real or rational variables
- $x_1, \dots, x_n$ are split into dependent variables $\overline{x}_D$ and independent variables $\overline{x}_I$

Input
constraints plus upper and lower bounds for $x_1, \dots, x_n$:

$$A\,\overline{x}_I = \overline{x}_D \qquad \text{with tableau } A \in \mathbb{Q}^{|D| \times |I|} \qquad (1)$$
$$l_i \leqslant x_i \leqslant u_i \qquad (2)$$

Output
satisfying assignment or "unsatisfiable"

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

DPLL(T) Simplex Algorithm (2)

$$A\overline{x}_I = \overline{x}_D \tag{1}$$
$$l_i \leqslant x_i \leqslant u_i \tag{2}$$

Method

- if (2) holds for all dependent variables, return current assignment
- otherwise select dependent variable $x_i \in D$ which violates (2)
- select suitable independent variable $x_j \in I$ such that x_i and x_j can be swapped in a pivoting step, resulting in new tableau

  $$A'\, x_{I'} = x_{D'}$$

  with $I' = I \cup \{x_i\} - \{x_j\}$ and $D' = D \cup \{x_j\} - \{x_i\}$
- change value of x_i to l_i or u_i, update values of dependent variables accordingly

DPLL(T) Simplex Algorithm (3)

$$A\overline{x}_I = \overline{x}_D \tag{1}$$
$$l_i \leqslant x_i \leqslant u_i \tag{2}$$

Pivoting

- swap dependent x_i and non-dependent x_j

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

- 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 $\forall x_k \in D - \{x_i\}$

DPLL(T) Simplex Algorithm (4)

$$A\overline{x}_I = \overline{x}_D \tag{1}$$
$$l_i \leqslant x_i \leqslant u_i \tag{2}$$

Suitability

- dependent variable x_i violates lower and/or upper bound
- pick independent variable x_j such that
  - 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 $x_j < u_j$
- problem is unsatisfiable if no suitable pivot exists

Bland's Rule

- pick lexicographically smallest (i, j) that is suitable pivot
- guarantees termination

Outline

Example

$-1 \leqslant x_1 \leqslant 0 \qquad -4 \leqslant x_2 \leqslant 0 \qquad -5 \leqslant x_3 \leqslant -4 \qquad -7 \leqslant x_4 \leqslant 1$

$$\begin{array}{c|cc} & x_1 & x_2 \\ \hline x_3 & 1 & 2 \\ x_4 & 2 & 1 \end{array} \quad \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \\ \hline 0 & 0 & 0 & 0 \end{array} \quad \longleftarrow \quad \begin{array}{c|cc} & x_1 & x_4 \\ \hline x_3 & -3 & 2 \\ x_2 & -2 & 1 \end{array} \quad \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \\ \hline 0 & 1 & 2 & 1 \end{array}$$

$$\downarrow \qquad\qquad\qquad\qquad \uparrow$$

$$\begin{array}{c|cc} & x_3 & x_2 \\ \hline x_1 & 1 & -2 \\ x_4 & 2 & -3 \end{array} \quad \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \\ \hline -4 & 0 & -4 & -8 \end{array} \qquad \begin{array}{c|cc} & x_3 & x_4 \\ \hline x_1 & -\frac{1}{3} & \frac{2}{3} \\ x_2 & \frac{2}{3} & -\frac{1}{3} \end{array} \quad \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \\ \hline \frac{7}{3} & -\frac{11}{3} & -5 & 1 \end{array}$$

$$\downarrow \qquad\qquad\qquad\qquad \uparrow$$

$$\begin{array}{c|cc} & x_3 & x_4 \\ \hline x_1 & -\frac{1}{3} & \frac{2}{3} \\ x_2 & \frac{2}{3} & -\frac{1}{3} \end{array} \quad \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \\ \hline -\frac{10}{3} & -\frac{1}{3} & -4 & -7 \end{array} \qquad \begin{array}{c|cc} & x_3 & x_2 \\ \hline x_1 & 1 & -2 \\ x_4 & 2 & -3 \end{array} \quad \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \\ \hline 3 & -4 & -5 & 2 \end{array}$$

$$\downarrow \qquad\qquad\qquad\qquad \uparrow$$

$$\begin{array}{c|cc} & x_1 & x_4 \\ \hline x_3 & -3 & 2 \\ x_2 & -2 & 1 \end{array} \quad \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \\ \hline -1 & -5 & -11 & -7 \end{array} \quad \longrightarrow \quad \begin{array}{c|cc} & x_1 & x_2 \\ \hline x_3 & 1 & 2 \\ x_4 & 2 & 1 \end{array} \quad \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \\ \hline -1 & -4 & -9 & -6 \end{array}$$

Example

$-1 \leqslant x_1 \leqslant 0 \qquad -4 \leqslant x_2 \leqslant 0 \qquad -5 \leqslant x_3 \leqslant -4 \qquad -7 \leqslant x_4 \leqslant 1$

$$\begin{array}{c|cc} & x_1 & x_2 \\ \hline x_3 & 1 & 2 \\ x_4 & 2 & 1 \end{array} \quad \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \\ \hline 0 & 0 & 0 & 0 \end{array}$$

$$\downarrow$$

$$\begin{array}{c|cc} & x_3 & x_2 \\ \hline x_1 & 1 & -2 \\ x_4 & 2 & -3 \end{array} \quad \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \\ \hline -4 & 0 & -4 & -8 \end{array}$$

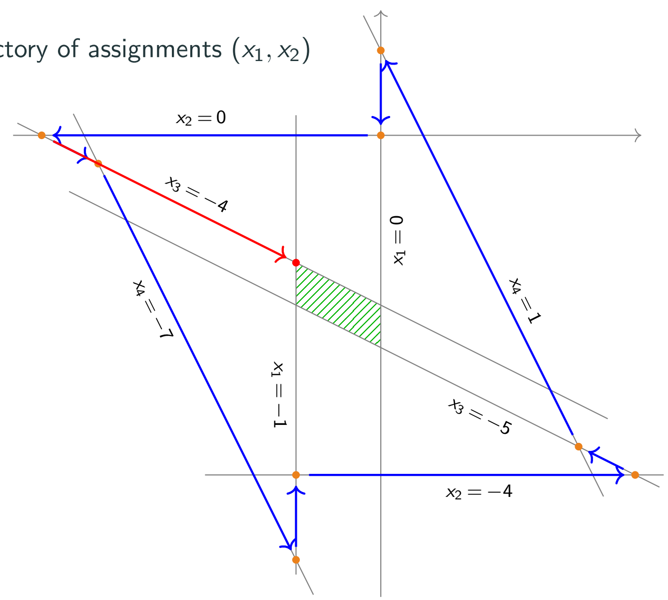
violation of Bland's rule

$$\downarrow$$

$$\begin{array}{c|cc} & x_3 & x_1 \\ \hline x_2 & \frac{1}{2} & -\frac{1}{2} \\ x_4 & \frac{1}{2} & \frac{3}{2} \end{array} \quad \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \\ \hline -1 & -\frac{3}{2} & -4 & -\frac{7}{2} \end{array}$$

satisfying assignment

trajectory of assignments (x_1, x_2)



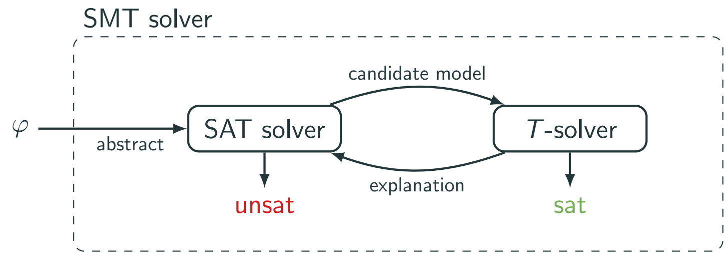
Disclaimer

if advice of Dr. Bland is neglected, no cure is guaranteed!

Outline

- Summary of Last Week
- Cyclic Simplex Example
- Branch and Bound
- Fourier-Motzkin Elimination

How to Be Lazy

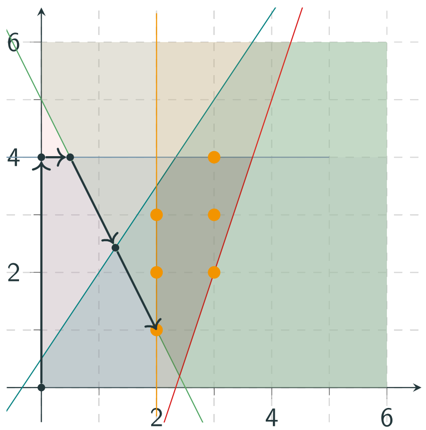


Theory T	**T-solving method**	
▶ equality logic	equality graphs	✓
▶ equality + uninterpreted functions (EUF)	congruence closure	✓
▶ linear real arithmetic (LRA)	DPLL(T) Simplex	✓
▶ linear integer arithmetic (LIA)	DPLL(T) Simplex + cuts	
▶ bitvectors (BV)		
▶ arrays (A)		

Example

$3x - 2y \geqslant -1$
$y \leqslant 4$
$2x + y \geqslant 5$
$3x - y \leqslant 7$

- looking for solution in $\mathbb{Z}^2$
- infinite $\mathbb{Q}^2$ solution space, six solutions in $\mathbb{Z}^2$
- Simplex returns $(\frac{9}{7}, \frac{17}{7})$



Idea (Branch and Bound)

- add constraints that exclude solution in $\mathbb{Q}^2$ but do not change solutions in $\mathbb{Z}^2$
- in current solution $1 < x < 2$, so use Simplex on two augmented problems:
  - $C \wedge x \leqslant 1$ unsatisfiable
  - $C \wedge x \geqslant 2$ satisfiable, Simplex can return $(2, 1)$

Algorithm BranchAndBound(φ)

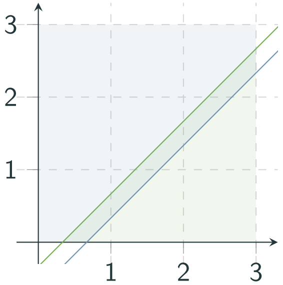
Input: LIA constraint φ
Output: unsatisfiable, or satisfying assignment

```
S ← decide φ over ℚ                                   ▷ e.g. by Simplex
if S′ = unsatisfiable then
    return unsatisfiable
else if S is solution over ℤ then
    return S
else
    x ← variable assigned non-integer value q in S
    S′ = BranchAndBound(φ ∧ x ⩽ ⌊q⌋)
    if S′ ≠ unsatisfiable then
        return S′
    else
        return BranchAndBound(φ ∧ x ⩾ ⌈q⌉)
```

Definition

$\mathbb{Q}^2$-solution space of linear arithmetic problem $Ax \leqslant b$ is bounded
if for all x_i there exist $l_i, u_i \in \mathbb{Q}$ such that all $\mathbb{Q}^2$-solutions v satisfy $l_i \leqslant v(x_i) \leqslant u_i$

Example



- $3x - 3y \geqslant 1 \wedge 3x - 3y \leqslant 2$
- unbounded problem
- no solution in $\mathbb{Z}^2$
- BranchAndBound keeps adding $x \geqslant n,\ y \geqslant m$
  $x \leqslant n,\ y \leqslant m$

Remarks

- BranchAndBound might not terminate if solution space is unbounded
- methods exist to derive solution bounds from tableau, but bounds are often too high for efficient practical procedures
- use cutting planes to restrict solution space more efficiently

LIA Application: Finding Work Schedules

Example (Scheduling Problem)

Is there a six-week cyclic work schedule for 22 employees who work 8-hour shifts, 24/7, 5 working days per week such that the following holds: In morning and afternoon shifts 6 employees are present, in night shifts 3. Joe does only morning shifts, Sally does not work on Sundays. Nobody works more than 6 days in a row.

Shift Schedule Requirements

- number of employees n
- set of shifts A (activities to be distributed)
- length of schedule (e.g. one week) and cyclicity
- requirement matrix R: R_{ij} is # employees required in shift i of day j
- prohibited shift sequences, maximal length of work blocks, ...

LIA Encoding

- integer variable corresponding to employee for each activity
- cardinality constraints for requirement matrix
- ...

Outline

- Summary of Last Week
- Cyclic Simplex Example
- Branch and Bound
- Fourier-Motzkin Elimination

Fourier-Motzkin Elimination

Aim

build theory solver for linear rational arithmetic (LRA):
decide whether conjunction of linear (in)equalities φ is satisfiable over $\mathbb{Q}$

Preprocessing: eliminate $\neq$

$(t_1 \neq t_2) \wedge \varphi$ is satisfiable iff $(t_1 < t_2) \wedge \varphi$ or $(t_1 > t_2) \wedge \varphi$ are satisfiable

Definition (Elimination step)

- for variable x in φ, can write φ as
  $$\bigwedge_i (x < U_i) \ \wedge\ \bigwedge_j (x \leqslant u_j) \ \wedge\ \bigwedge_k (L_k < x) \ \wedge\ \bigwedge_m (\ell_m \leqslant x) \ \wedge\ \psi$$
  where U_i, u_j, L_k, ℓ_m, ψ are without x

  formula without x
- let *elim*(φ, x) be conjunction of
  $$\bigwedge_i \bigwedge_k (L_k < U_i) \quad \bigwedge_i \bigwedge_m (\ell_m < U_i) \quad \bigwedge_j \bigwedge_k (L_k < u_j) \quad \bigwedge_j \bigwedge_m (\ell_m \leqslant u_j) \quad \psi$$

Lemma

φ is LRA-satisfiable iff elim(φ, x) is LRA-satisfiable

Observation

- can subsequently eliminate all variables
- checking satisfiability of formula without variables is easy
- so obtain decision procedure for LRA!

Example

... on blackboard

Bibliography

Bruno Dutertre and Leonardo de Moura.
A Fast Linear-Arithmetic Solver for DPLL(T).
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

Daniel Kroening and Ofer Strichman
The Simplex Algorithm
Section 5.2 of Decision Procedures — An Algorithmic Point of View
Springer, 2008

Bertram Felgenhauer and Aart Middeldorp
Constructing Cycles in the Simplex Method for DPLL(T)
Proc. 14th International Colloquium on Theoretical Aspects of Computing,
LNCS 10580, pp. 213 – 228, 2017

Christoph Erkinger and Nysret Musliu
Personnel Scheduling as Satisfiability Modulo Theories
Proc. 26th International Joint Conference on Artificial Intelligence,
pp. 614 – 621, 2017