Solving Arithmetic Constraints in SMT

Clark Barrett (joint work with Tim King)

CS 357, Lecture 14, November 4, 2015

Big Ideas of Today's Lecture

- SAT, SMT & DPLL($\mathcal{T}$)
- Simplex for DPLL($\mathcal{T}$)
- Sum of Infeasibilities for SMT
- Leverage LP/MIP Solvers
- Experiments

Big Ideas of Today's Lecture

- SAT, SMT & DPLL($\mathcal{T}$)

  How to combine CDCL + Simplex?

- Simplex for DPLL($\mathcal{T}$)
- Sum of Infeasibilities for SMT
- Leverage LP/MIP Solvers
- Experiments

Big Ideas of Today's Lecture

- SAT, SMT & DPLL($\mathcal{T}$)

  How to combine CDCL + Simplex?

- Simplex for DPLL($\mathcal{T}$)

  SOTA decision procedure for `QF_LRA`

- Sum of Infeasibilities for SMT

- Leverage LP/MIP Solvers

- Experiments

BIG IDEAS OF TODAY'S LECTURE

- SAT, SMT & DPLL($\mathcal{T}$)

  How to combine CDCL + Simplex?

- Simplex for DPLL($\mathcal{T}$)

  SOTA decision procedure for `QF_LRA`

- Sum of Infeasibilities for SMT [FMCAD13]

  Robust decision procedure for `QF_LRA`

- Leverage LP/MIP Solvers

- Experiments

BIG IDEAS OF TODAY'S LECTURE

- SAT, SMT & DPLL($\mathcal{T}$)

  How to combine CDCL + Simplex?

- Simplex for DPLL($\mathcal{T}$)

  SOTA decision procedure for `QF_LRA`

- Sum of Infeasibilities for SMT [FMCAD13]

  Robust decision procedure for `QF_LRA`

- Leverage LP/MIP Solvers [FMCAD14]

  Accelerate exact precision solver

- Experiments

Table of Contents

SATISFIABILITY MODULO THEORIES

- Theories enforce the semantics of the syntax
- $\mathcal{T}_{\mathbb{R}}$: theory of reals

  Domain of values is $\mathbb{R}$
  "+" is mathematical $+$
  "`0`" is mathematical 0
  "<" is mathematical $<$
  ...
- $\mathcal{T}_{\mathbb{Z}}$: theory of integers

SATISFIABILITY MODULO THEORIES

- Theories enforce the semantics of the syntax
- $\mathcal{T}_{\mathbb{R}}$: theory of reals

  Domain of values is $\mathbb{R}$
  "+" is mathematical $+$
  "`0`" is mathematical 0
  "<" is mathematical $<$
  ...

- $\mathcal{T}_{\mathbb{Z}}$: theory of integers

SMT Problem

Does there exist a variable assignment a for the theory $\mathcal{T}$ such that the formula ϕ evaluates to **true**?

QF_LRA Example

Quantifier-Free Linear Real Arithmetic

$$\phi \equiv \begin{array}{cl} & (y \leq 4) \\ \wedge & (y \geq 5 \vee x + y \leq 6) \\ \wedge & (x > 2 \vee x - y \geq 1) \end{array}$$

Is there an assignment that makes ϕ evaluate to **true**?

$$a : \mathcal{X} \to \mathbb{R}$$

DPLL($\mathcal{T}$)

SMT Solver Framework

$$\begin{array}{rcl} y & \leq & 4 \\ y \geq 5 & \vee & x + y \leq 6 \\ x > 2 & \vee & x - y \geq 1 \end{array}$$

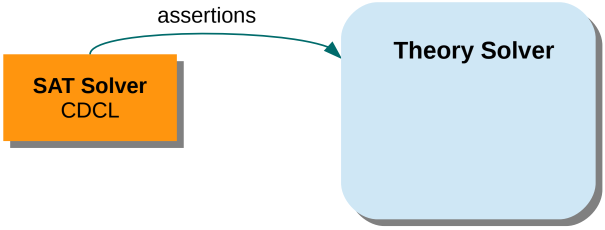
DPLL($\mathcal{T}$)

SMT Solver Framework

$$\begin{array}{lcl} \mathbf{y} & \leq & \mathbf{4} \\ y \geq 5 & \vee & x + y \leq 6 \\ x > 2 & \vee & x - y \geq 1 \end{array}$$

DPLL($\mathcal{T}$)

SMT Solver Framework

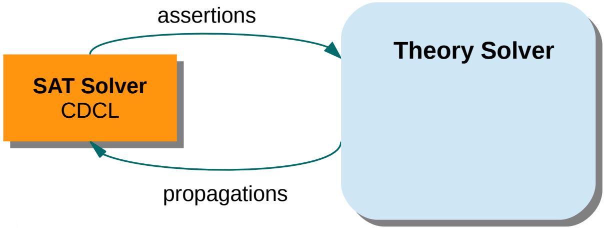


$y \leq 4$

$$\begin{array}{ccc} \mathbf{y} & \leq & \mathbf{4} \\ y \geq 5 & \vee & x + y \leq 6 \\ x > 2 & \vee & x - y \geq 1 \end{array}$$

DPLL($\mathcal{T}$)

SMT Solver Framework

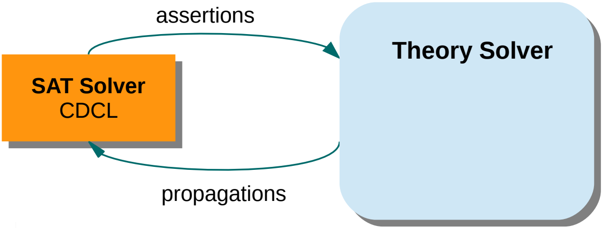


$y \leq 4$

$$\begin{array}{rcl} \mathbf{y} & \leq & \mathbf{4} \\ \mathbf{y \geq 5} & \vee & x + y \leq 6 \\ x > 2 & \vee & x - y \geq 1 \end{array}$$

DPLL($\mathcal{T}$)

SMT Solver Framework

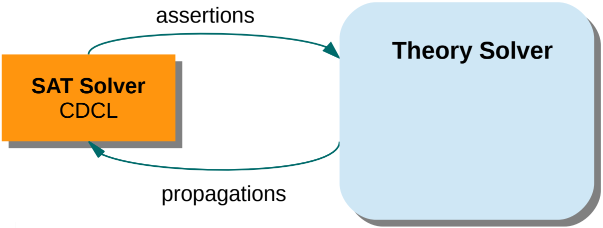


$$y \leq 4$$
$$x + y \leq 6$$

$$\begin{array}{rcl} & & \mathbf{y} \leq \mathbf{4} \\ \mathbf{y} \geq \mathbf{5} & \vee & \mathbf{x} + \mathbf{y} \leq \mathbf{6} \\ x > 2 & \vee & x - y \geq 1 \end{array}$$

DPLL($\mathcal{T}$)

SMT Solver Framework

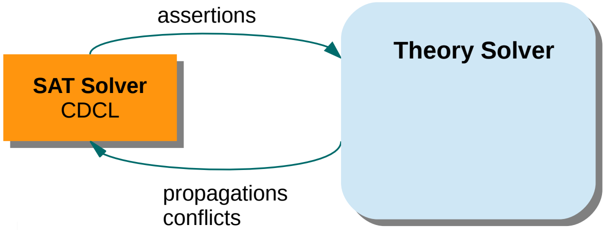


$$y \leq 4$$
$$x + y \leq 6$$
$$x > 2$$

$$\begin{array}{rcl} & & \mathbf{y} \leq \mathbf{4} \\ \mathbf{y} \geq \mathbf{5} & \vee & \mathbf{x} + \mathbf{y} \leq \mathbf{6} \\ \mathbf{x} > \mathbf{2} & \vee & x - y \geq 1 \end{array}$$

DPLL($\mathcal{T}$)

SMT Solver Framework

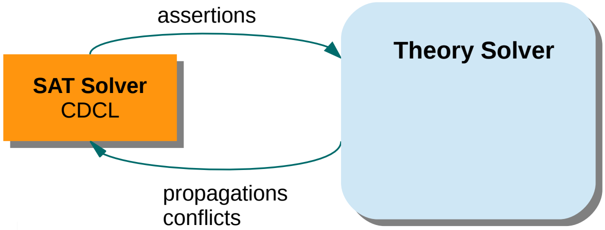


$$\begin{aligned} y &\leq 4 \\ x + y &\leq 6 \\ x &> 2 \end{aligned}$$

$$\begin{array}{rcl} \mathbf{y} \leq \mathbf{4} & & \\ \mathbf{y} \geq \mathbf{5} & \vee & \mathbf{x} + \mathbf{y} \leq \mathbf{6} \\ \mathbf{x} > \mathbf{2} & \vee & x - y \geq 1 \\ \mathbf{x} \leq \mathbf{2} & \vee & \mathbf{x} + \mathbf{y} < \mathbf{6} \vee \mathbf{y} > \mathbf{4} \end{array}$$

DPLL($\mathcal{T}$)

SMT Solver Framework

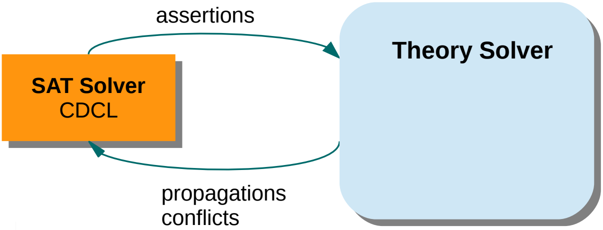


$$\begin{aligned} y &\leq 4 \\ x + y &\leq 6 \\ x &\leq 2 \end{aligned}$$

$$\begin{array}{rcl} & & \mathbf{y} \leq \mathbf{4} \\ \mathbf{y} \geq \mathbf{5} & \vee & \mathbf{x} + \mathbf{y} \leq \mathbf{6} \\ \mathbf{x} > \mathbf{2} & \vee & x - y \geq 1 \\ \mathbf{x} \leq \mathbf{2} & \vee & \mathbf{x} + \mathbf{y} < \mathbf{6} \vee \mathbf{y} > \mathbf{4} \end{array}$$

DPLL($\mathcal{T}$)

SMT Solver Framework



$$\begin{aligned} y &\leq 4 \\ x + y &\leq 6 \\ x &\leq 2 \\ x - y &\geq 1 \end{aligned}$$

$$\begin{array}{rcl} \mathbf{y} & \leq & \mathbf{4} \\ \mathbf{y} \geq \mathbf{5} & \vee & \mathbf{x} + \mathbf{y} \leq \mathbf{6} \\ \mathbf{x} > \mathbf{2} & \vee & \mathbf{x} - \mathbf{y} \geq \mathbf{1} \\ \mathbf{x} \leq \mathbf{2} & \vee & \mathbf{x} + \mathbf{y} < \mathbf{6} \vee \mathbf{y} > \mathbf{4} \end{array}$$

Table of Contents

DECISION PROCEDURE FOR QF_LRA

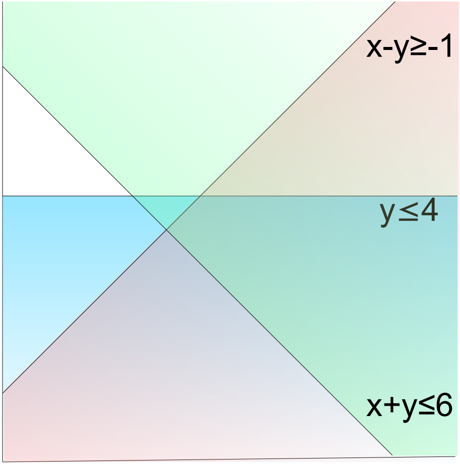
QUANTIFIER FREE LINEAR REAL ARITHMETIC

Is there a satisfying assignment, $a : \mathcal{X} \to \mathbb{R}$, that makes,

$$\begin{array}{ccccc} x & + & y & \leq & 6 \\ x & - & y & \geq & -1 \\ & & y & \leq & 4 \end{array}$$

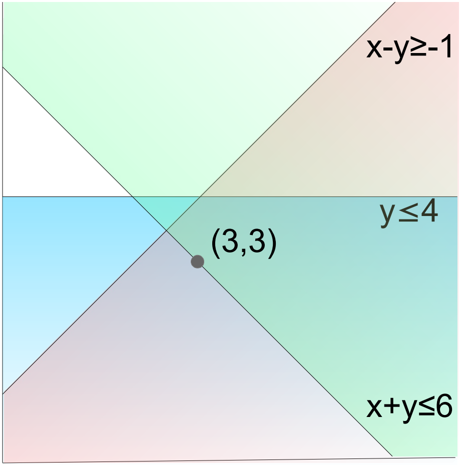
evaluate to true?

VISUALLY



$$\begin{array}{ccccc} x & + & y & \leq & 6 \\ x & - & y & \geq & -1 \\ & & y & \leq & 4 \end{array}$$

VISUALLY

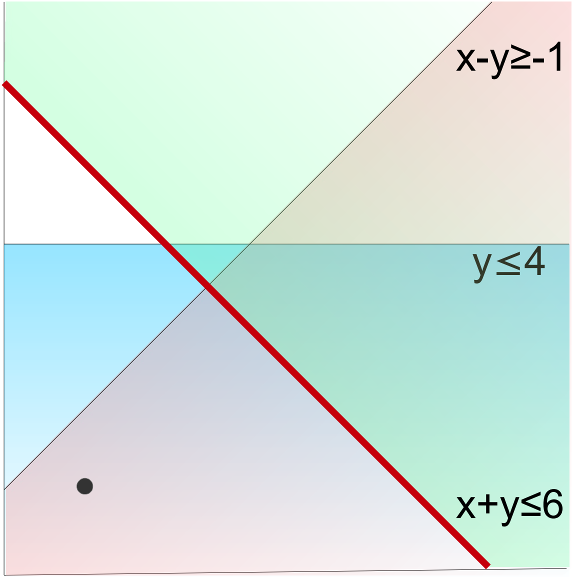


$$\begin{array}{rcrcr} x & + & y & \leq & 6 \\ x & - & y & \geq & -1 \\ & & y & \leq & 4 \end{array}$$

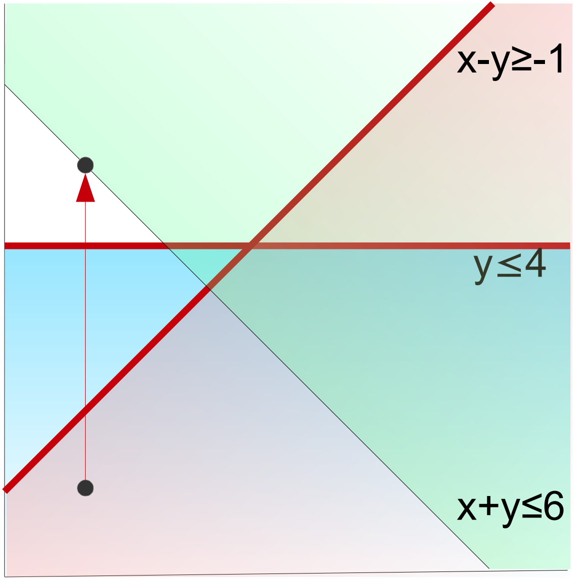
$$\begin{bmatrix} a_x \\ a_y \end{bmatrix} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

How?

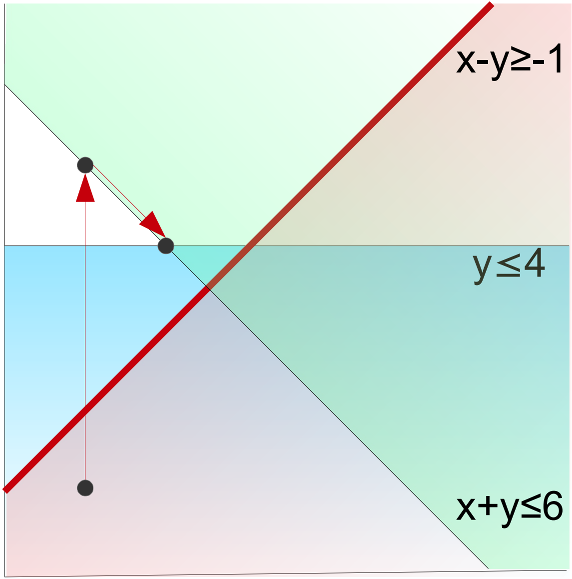
SIMPLEX FOR DPLL($\mathcal{T}$) SEARCH



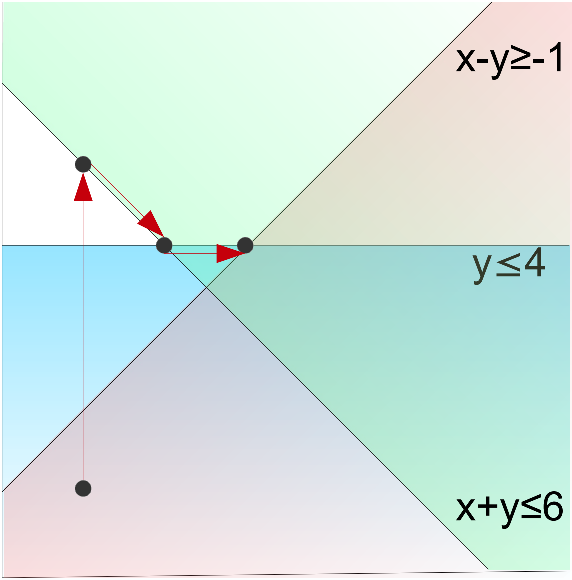
SIMPLEX FOR DPLL($\mathcal{T}$) SEARCH



Simplex for DPLL($\mathcal{T}$) Search



Simplex for DPLL($\mathcal{T}$) Search



PREPROCESSING

- Introduce a fresh s_i for each $\sum T_{i,j} \cdot x_j$
- Literals are of the form:

$$\bigwedge \left(s_i = \sum_{j \in \mathcal{N}} T_{i,j} \cdot x_j \right) \wedge \bigwedge l_i \leq x_i \leq u_i$$

  and s_i appears in exactly 1 equality.

- Collect into:

$$T\vec{\mathcal{X}} = 0 \qquad \vec{l} \leq \vec{\mathcal{X}} \leq \vec{u}$$

PREPROCESSING

- Introduce a fresh s_i for each $\sum T_{i,j} \cdot x_j$
- Literals are of the form:

$$\bigwedge \left(s_i = \sum_{j \in \mathcal{N}} T_{i,j} \cdot x_j \right) \wedge \bigwedge l_i \leq x_i \leq u_i$$

  and s_i appears in exactly 1 equality.

- Collect into:

$$T\vec{\mathcal{X}} = 0 \qquad \vec{l} \leq \vec{\mathcal{X}} \leq \vec{u}$$

PREPROCESSING

- Introduce a fresh s_i for each $\sum T_{i,j} \cdot x_j$
- Literals are of the form:

$$\bigwedge \left(s_i = \sum_{j \in \mathcal{N}} T_{i,j} \cdot x_j \right) \wedge \bigwedge l_i \leq x_i \leq u_i$$

  and s_i appears in exactly 1 equality.

- Collect into:

$$T\vec{\mathcal{X}} = 0 \qquad \vec{l} \leq \vec{\mathcal{X}} \leq \vec{u}$$

Basic, Nonbasic, & Tableau

- Every row in T is solved for a variable x_i

$$x_i = \sum_{j \in \mathcal{N}} T_{i,j} x_j$$

- Not-solved-for variables are **nonbasic** ($j \in \mathcal{N}$)
- Set of solved-for variables are **basic** ($i \in \mathcal{B}$)

Updating Nonbasic Variables

Changing the assignment to $j \in \mathcal{N}$ is easy:

- $a_j \mathrel{+}= \delta$
- for all $i \in \mathcal{B}$:
  $$a_i \mathrel{+}= T_{i,j} \cdot \delta.$$

Updating Nonbasic Variables

Changing the assignment to $j \in \mathcal{N}$ is easy:

- $a_j \mathrel{+}= \delta$
- for all $i \in \mathcal{B}$:
  $$a_i \mathrel{+}= T_{i,j} \cdot \delta.$$

Add the Invariant

The nonbasic variables satisfy their bounds.

PIVOT(i, j)

MOVE VARIABLES IN/OUT OF $\mathcal{B}$

Preconditions

Given x_i basic, x_j nonbasic, and $T_{i,j} \neq 0$,
PIVOT(i, j) makes x_i nonbasic and x_j basic.

$\textsc{Pivot}(i, j)$

Move Variables In/Out of $\mathcal{B}$

Preconditions

Given x_i basic, x_j nonbasic, and $T_{i,j} \neq 0$,
$\textsc{Pivot}(i, j)$ makes x_i nonbasic and x_j basic.

- Take x_i's row

$$x_i = T_{i,j} x_j + \sum T_{i,k} x_k$$

- Solve for x_j

$$x_j = \frac{1}{T_{i,j}} x_i + \sum -\frac{T_{i,k}}{T_{i,j}} x_k$$

- Replace x_j everywhere else in T

PIVOT(i, j)

MOVE VARIABLES IN/OUT OF $\mathcal{B}$

Preconditions

Given x_i basic, x_j nonbasic, and $T_{i,j} \neq 0$,
PIVOT(i, j) makes x_i nonbasic and x_j basic.

- Take x_i's row

$$x_i = T_{i,j}x_j + \sum T_{i,k}x_k$$

- Solve for x_j

$$x_j = \frac{1}{T_{i,j}}x_i + \sum -\frac{T_{i,k}}{T_{i,j}}x_k$$

- Replace x_j everywhere else in T

Preserves Linear Subspace

PIVOT(i, j) preserves $Ta = 0$.

Tableau Example

$$
\begin{array}{ccccr}
x & + & y & \leq & 6 \\
x & - & y & \geq & -1 \\
 & & y & \leq & 4
\end{array}
$$

Tableau Example

$$
\begin{array}{rcrcr}
s_1 & = & x & + & y \\
s_2 & = & x & - & y
\end{array}
$$

$$s_1 \geq 6 \wedge s_2 \geq -1 \wedge y \leq 4$$

Tableau Example

$$\begin{array}{ccccc} s_1 & = & x & + & y \\ s_2 & = & x & - & y \end{array}$$

$$s_1 \geq 6 \wedge s_2 \geq -1 \wedge y \leq 4$$

$$T\vec{\mathcal{X}} = \begin{bmatrix} -1 & 0 & 1 & 1 \\ 0 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ x \\ y \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\mathcal{B} = \{s_1, s_2\}, \mathcal{N} = \{x, y\}$$

Tableau Example

$$\begin{array}{ccccc} s_1 & = & x & + & y \\ s_2 & = & x & - & y \end{array}$$

$$s_1 \geq 6 \wedge s_2 \geq -1 \wedge y \leq 4$$

$$T\vec{\mathcal{X}} = \begin{bmatrix} -1 & 0 & 1 & 1 \\ 0 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ x \\ y \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\mathcal{B} = \{s_1, s_2\}, \mathcal{N} = \{x, y\}$$

TABLEAU EXAMPLE

$$\begin{array}{rcrcr} s_1 & = & x & + & y \\ s_2 & = & x & - & y \end{array}$$

$$s_1 \geq 6 \wedge s_2 \geq -1 \wedge y \leq 4$$

$$T\vec{\mathcal{X}} = \begin{bmatrix} -1 & 0 & 1 & 1 \\ 0 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ x \\ y \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\mathcal{B} = \{s_1, s_2\}, \mathcal{N} = \{x, y\}$$

Simplex for DPLL($\mathcal{T}$)

Pseudocode

while $i \in \mathcal{B}$ s.t. $a_i > u_i$ or ... **do**

select some $x_i = \sum T_{i,j} \cdot x_j$

if $\sum T_{i,j} \cdot x_j$ is at a minimum **then**

return a row conflict

else

Select j from $\sum T_{i,j} \cdot x_j$

Change the assignment of x_j s.t. $a_i \leftarrow u_i$

Pivot(i, j)

Simplex for DPLL($\mathcal{T}$)

Pseudocode

while $i \in \mathcal{B}$ s.t. $a_i > u_i$ or ... **do**

select some $x_i = \sum T_{i,j} \cdot x_j$

if $\sum T_{i,j} \cdot x_j$ is at a minimum **then**

return a row conflict

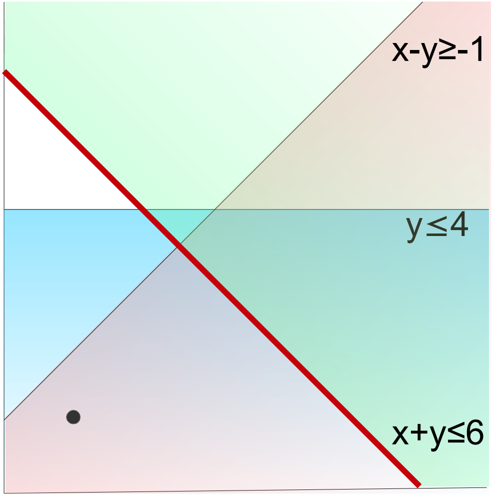
else

Select j from $\sum T_{i,j} \cdot x_j$

Change the assignment of x_j s.t. $a_i \leftarrow u_i$

Pivot(i, j) $\triangleright$ $O(|T|)$

SIMPLEX FOR DPLL($\mathcal{T}$) SEARCH



Greedily fix

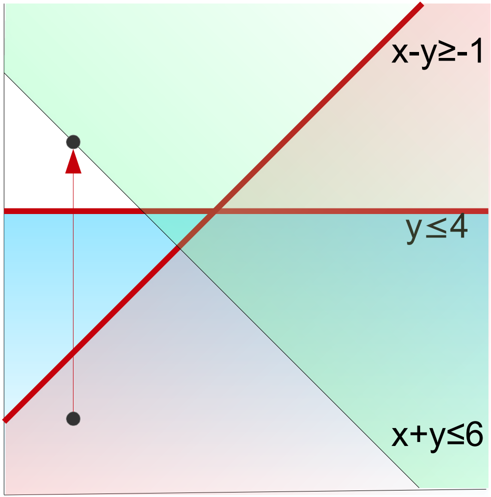
$$x + y \leq 6$$

ignoring

$$x - y \geq -1 \quad \text{and}$$

$$y \leq 4$$

Simplex for DPLL($\mathcal{T}$) Search



Greedily fix

$$x + y \leq 6$$

ignoring

$$x - y \geq -1 \quad \text{and}$$

$$y \leq 4$$

CONFLICT DETECTION

- Select x_i s.t. $a_i > u_i$ and $i \in \mathcal{B}$

$$x_i = \sum_{j \in \mathcal{N}} T_{i,j} x_j$$

Conflict Detection

- Select x_i s.t. $a_i > u_i$ and $i \in \mathcal{B}$

$$x_i = \sum_{j \in \mathcal{N}} T_{i,j} x_j$$

- If
  - $a_j = l_j$ for all $T_{i,j} > 0$ and
  - $a_k = u_k$ for all $T_{i,k} < 0$,
  - then $\sum T_{i,j} x_j$ must be minimized.

Conflict Detection

- Select x_i s.t. $a_i > u_i$ and $i \in \mathcal{B}$

$$x_i = \sum_{j \in \mathcal{N}} T_{i,j} x_j$$

- If
  - $a_j = l_j$ for all $T_{i,j} > 0$ and
  - $a_k = u_k$ for all $T_{i,k} < 0$,
  - then $\sum T_{i,j} x_j$ must be minimized.
- Thus $x_i \geq a_i$ is entailed by

$$x_i = \sum_{j \in \mathcal{N}} T_{i,j} x_j \wedge \bigwedge_{T_{i,j} > 0} x_j \geq l_j \bigwedge_{T_{i,k} < 0} x_k \geq u_k$$

Conflict Detection

- Select x_i s.t. $a_i > u_i$ and $i \in \mathcal{B}$

$$x_i = \sum_{j \in \mathcal{N}} T_{i,j} x_j$$

- If
  - $a_j = l_j$ for all $T_{i,j} > 0$ and
  - $a_k = u_k$ for all $T_{i,k} < 0$,
  - then $\sum T_{i,j} x_j$ must be minimized.
- Thus $x_i \geq a_i$ is entailed by

$$x_i = \sum_{j \in \mathcal{N}} T_{i,j} x_j \wedge \bigwedge_{T_{i,j} > 0} x_j \geq l_j \bigwedge_{T_{i,k} < 0} x_k \geq u_k$$

- But $u_i \geq x_i \geq a_i > u_i$!

Conflict Detection

continued

Thus, the following is **Unsat** in $\mathcal{T}_{\mathbb{R}}$:

$$\left(x_i = \sum_{j \in \mathcal{N}} T_{i,j} x_j\right) \wedge \left(\bigwedge_{T_{i,j}>0} x_j \geq l_j\right) \wedge \left(\bigwedge_{T_{i,k}<0} x_k \geq u_k\right) \wedge x_i \leq u_i$$

Eager Conflict Detection

Small Contribution

- $\forall i \in \mathcal{B}$ track the cardinalities of the sets:

$$J = \{j | T_{i,j} > 0, a_j = l_j\} \quad K = \{k | T_{i,k} < 0, a_k = u_k\}$$

Eager Conflict Detection

Small Contribution

- $\forall i \in \mathcal{B}$ track the cardinalities of the sets:

$$J = \{j | T_{i,j} > 0, a_j = l_j\} \quad K = \{k | T_{i,k} < 0, a_k = u_k\}$$

- Suppose x_i is basic with n nonbasic vars on its row.
- If $a_i > u_i$ and $|J| + |K| = n$, a conflict can be extracted from the row T_i.

Eager Conflict Detection

Small Contribution

- $\forall i \in \mathcal{B}$ track the cardinalities of the sets:

$$J = \{j | T_{i,j} > 0, a_j = l_j\} \quad K = \{k | T_{i,k} < 0, a_k = u_k\}$$

- Suppose x_i is basic with n nonbasic vars on its row.
- If $a_i > u_i$ and $|J| + |K| = n$, a conflict can be extracted from the row T_i.
- Bookkeeping $\rightarrow O(1)$-amortized conflict detection

Eager Conflict Detection

Small Contribution

- $\forall i \in \mathcal{B}$ track the cardinalities of the sets:

$$J = \{j | T_{i,j} > 0, a_j = l_j\} \quad K = \{k | T_{i,k} < 0, a_k = u_k\}$$

- Suppose x_i is basic with n nonbasic vars on its row.
- If $a_i > u_i$ and $|J| + |K| = n$, a conflict can be extracted from the row T_i.
- Bookkeeping $\rightarrow O(1)$-amortized conflict detection
- Never miss conflicts!

Simplex for DPLL($\mathcal{T}$)

With Eager Conflict Detection

check for row conflicts

while $i \in \mathcal{B}$ s.t. $a_i > u_i$ or … and no row conflicts **do**

select some $x_i = \sum T_{i,j} \cdot x_j$

Select j from $\sum T_{i,j} \cdot x_j$

Change the assignment of x_j s.t. $a_j \leftarrow u_j$

$\textsc{Pivot}(i, j)$

check for row conflicts

Simplex for DPLL($\mathcal{T}$)

With Eager Conflict Detection

check for row conflicts

while $i \in \mathcal{B}$ s.t. $a_i > u_i$ or ... and no row conflicts **do**

select some $x_i = \sum T_{i,j} \cdot x_j$

Select j from $\sum T_{i,j} \cdot x_j$

Change the assignment of x_j s.t. $a_j \leftarrow u_j$

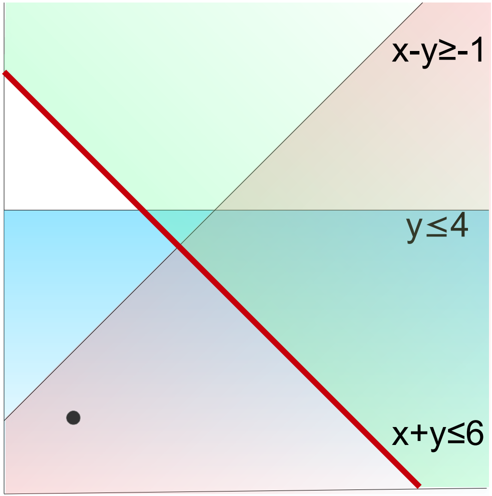
Pivot(i, j)

check for row conflicts

Table of Contents

Simplex for DPLL($\mathcal{T}$) Search

Reminder



Greedily fix

$$x + y \leq 6$$

ignoring

$$x - y \geq -1 \quad \text{and}$$

$$y \leq 4$$

SUM OF INFEASIBILITIES

- Infeasibility of x_i is how much x_i violates its bounds.

$$\mathrm{V}_i = \begin{cases} a_i - u_i & a_i > u_i \\ 0 & l_i \leq a_i \leq u_i \\ l_i - a_i & a_i < l_i \end{cases}$$

- Sum of Infeasibilities:

$$\mathrm{V}(\mathcal{X}) = \sum_{x_i \in \mathcal{X}} \mathrm{V}_i$$

- SOISIMPLEX minimizes $\mathrm{V}(\mathcal{X})$ every round

SUM OF INFEASIBILITIES

- Infeasibility of x_i is how much x_i violates its bounds.

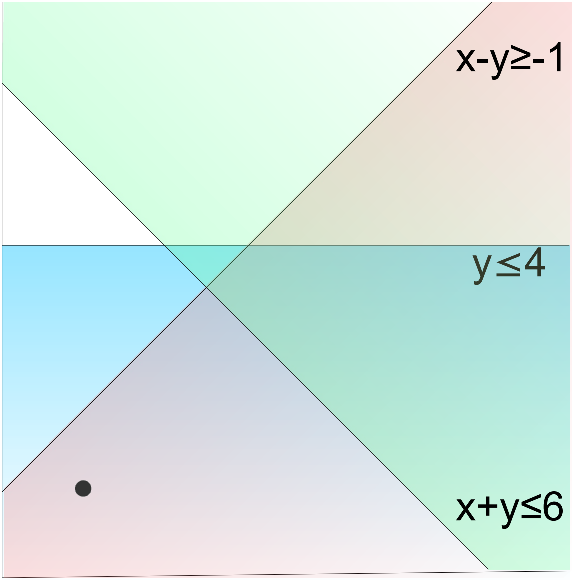
$$\mathrm{V}_i = \begin{cases} a_i - u_i & a_i > u_i \\ 0 & l_i \leq a_i \leq u_i \\ l_i - a_i & a_i < l_i \end{cases}$$

- Sum of Infeasibilities:

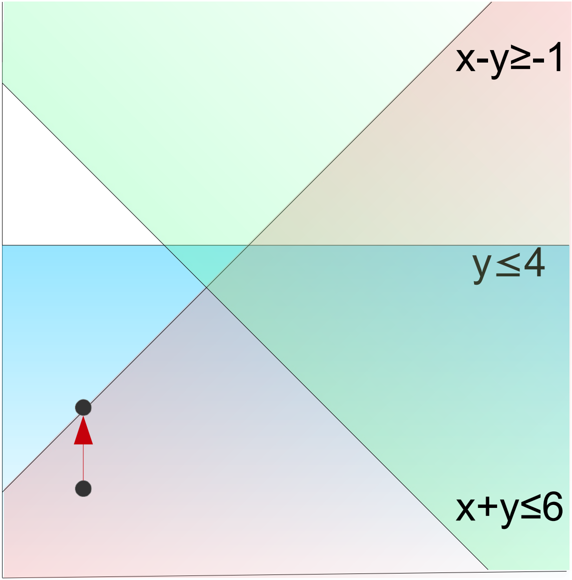
$$\mathrm{V}(\mathcal{X}) = \sum_{x_i \in \mathcal{X}} \mathrm{V}_i$$

- SOISIMPLEX minimizes $\mathrm{V}(\mathcal{X})$ every round
  - Known in optimization
  - New for SMT

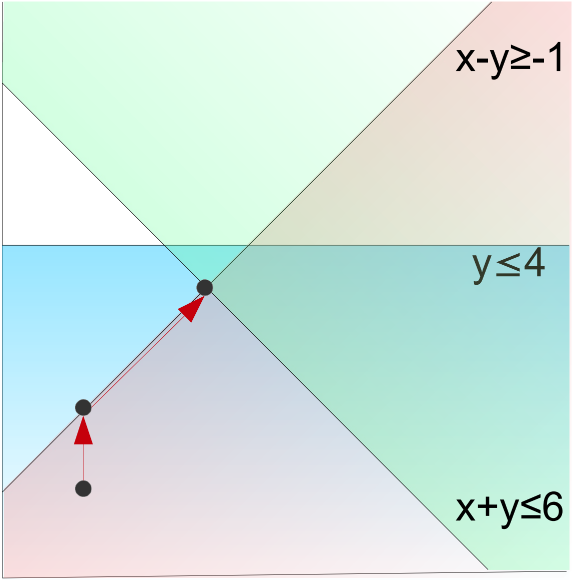
SOISIMPLEX SEARCH



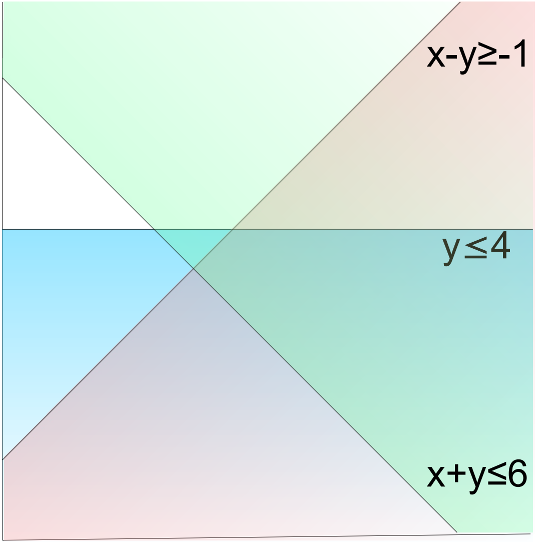
SOISimplex Search



SOISIMPLEX SEARCH

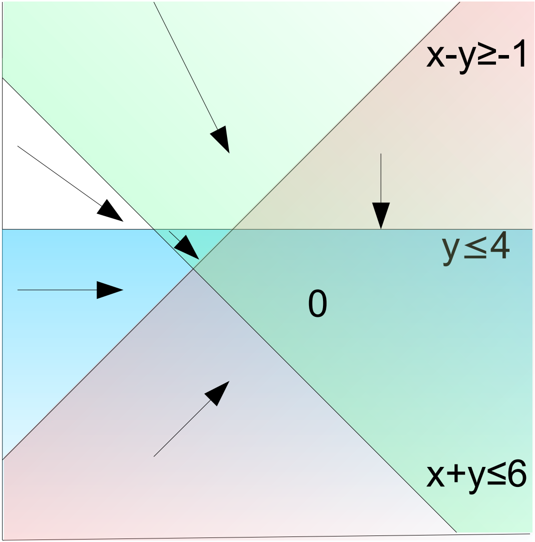


Direction of V($\mathcal{X}$)



$$\begin{cases} a_i \ldots & a_i > u_i \\ 0 & \text{otherwise} \\ \ldots - a_i & a_i < l_i \end{cases}$$

DIRECTION OF V($\mathcal{X}$)



$$\begin{cases} a_i \ldots & a_i > u_i \\ 0 & \text{otherwise} \\ \ldots - a_i & a_i < l_i \end{cases}$$

$$\mathrm{V}(\mathcal{X}) = 0 \text{ iff } a \text{ is sat}$$

SOISimplex Highlevel

Rough Sketch

procedure SOISimplex

- **while** V($\mathcal{X}$) is not at a minimum **do**
  - select a variable x_j
  - update x_j s.t. V($\mathcal{X}$) decreases and
    - $a_i \leftarrow u_i$ (or ...) for some $i \in \mathcal{B}$
  - Pivot(i, j)
    - ▷ can check rows for conflicts
- **return** (**if** (V($\mathcal{X}$) $= 0$) **then Sat else** SoiQE())

SOISimplex Highlevel

Rough Sketch

procedure SOISimplex

while $V(\mathcal{X})$ is not at a minimum **do**

select a variable x_j

update x_j s.t. $V(\mathcal{X})$ decreases and

$a_i \leftarrow u_i$ (or ...) for some $i \in \mathcal{B}$

Pivot(i, j)

$\triangleright$ can check rows for conflicts

return (**if** $(V(\mathcal{X}) = 0)$ **then Sat else** SoiQE())

SOISimplex Highlevel

Rough Sketch

procedure SOISimplex

while V($\mathcal{X}$) is not at a minimum **do**

select a variable x_j

update x_j s.t. V($\mathcal{X}$) decreases and

$a_i \leftarrow u_i$ (or ...) for some $i \in \mathcal{B}$

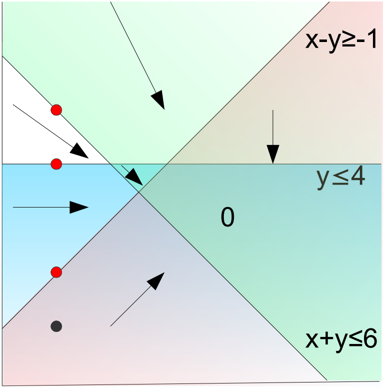
Pivot(i, j)

▷ can check rows for conflicts

return (**if** (V($\mathcal{X}$) = 0) **then Sat else** SoiQE())

Breakpoints

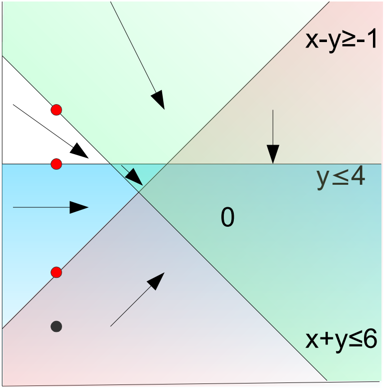
Where V($\mathcal{X}$) changes



$$\delta \in \left\{ \frac{a_i - u_j}{T_{i,j}}, \frac{a_i - l_j}{T_{i,j}}, \ldots, a_j - u_j, a_j - l_j \right\}$$

BREAKPOINTS

WHERE V($\mathcal{X}$) CHANGES



$$\delta \in \{1, 3, 4\}$$

SOISELECT()

Select x_j on $V(\mathcal{X})$'s row

s.t. x_j is not at its bound

Compute breakpoints $\{\delta\}$ for x_j

Compute $V(\mathcal{X})$ post $\textsc{Update}(j, \delta)$ for each δ

return the δ and corresponding i with

the lowest $V(\mathcal{X})$ post $\textsc{Update}(j, \delta)$

SOISELECT()

Select x_j on $\mathrm{V}(\mathcal{X})$'s row

s.t. x_j is not at its bound

Compute breakpoints $\{\delta\}$ for x_j

Compute $\mathrm{V}(\mathcal{X})$ post $\textsc{Update}(j, \delta)$ for each δ

return the δ and corresponding i with

the lowest $\mathrm{V}(\mathcal{X})$ post $\textsc{Update}(j, \delta)$

$\mathrm{V}(\mathcal{X})$ monotonically decreases!

SOISIMPLEX

FILLING IN THE SKETCH

while $\mathrm{V}(\mathcal{X})$ is not at a minimum **do**

  $\langle i, \delta, j \rangle \leftarrow$ SOISELECT()

  UPDATE(j, δ)

  PIVOT(i, j)

    ▷ can check rows for conflicts

return (**if** $(\mathrm{V}(\mathcal{X}) = 0)$ **then Sat else** SoiQE())

SOISimplex

Filling in the Sketch

while $V(\mathcal{X})$ is not at a minimum **do**

$\langle i, \delta, j \rangle \leftarrow$ SOISelect()

Update(j, δ)

Pivot(i, j)

▷ can check rows for conflicts

return (**if** $(V(\mathcal{X}) = 0)$ **then Sat else** SoiQE())

SOISimplex

Filling in the Sketch

while V($\mathcal{X}$) is not at a minimum **do**

$\langle i, \delta, j\rangle \leftarrow$ SOISelect()

Update(j, δ)

Pivot(i, j)

$\triangleright$ can check rows for conflicts

return (**if** (V($\mathcal{X}$) = 0) **then** **Sat** **else** SoiQE())

SOISimplex

Filling in the Sketch

while V($\mathcal{X}$) is not at a minimum **do**

$\langle i, \delta, j \rangle \leftarrow$ SOISelect()

Update(j, δ)

Pivot(i, j)

$\triangleright$ can check rows for conflicts

return (**if** (V($\mathcal{X}$) = 0) **then Sat else** SoiQE())

What happens when $V(\mathcal{X})$ is minimal?

- Suppose $V(\mathcal{X})$ is minimal and $V(\mathcal{X}) > 0$
- Suppose $a_i > u_i$ for all $V_i > 0$

What happens when $V(\mathcal{X})$ is minimal?

- Suppose $V(\mathcal{X})$ is minimal and $V(\mathcal{X}) > 0$
- Suppose $a_i > u_i$ for all $V_i > 0$
- $V_i = (a_i - u_i) > 0$
- But, $0 \geq (x_i - u_i)$

WHAT HAPPENS WHEN $V(\mathcal{X})$ IS MINIMAL?

- Suppose $V(\mathcal{X})$ is minimal and $V(\mathcal{X}) > 0$
- Suppose $a_i > u_i$ for all $V_i > 0$
- $V_i = (a_i - u_i) > 0$
- But, $0 \geq (x_i - u_i)$
- If $V(\mathcal{X})$ is minimal, then

$$\sum x_i \geq \sum a_i$$

What happens when $\mathrm{V}(\mathcal{X})$ is minimal?

- Suppose $\mathrm{V}(\mathcal{X})$ is minimal and $\mathrm{V}(\mathcal{X}) > 0$
- Suppose $a_i > u_i$ for all $\mathrm{V}_i > 0$
- $\mathrm{V}_i = (a_i - u_i) > 0$
- But, $0 \geq (x_i - u_i)$
- If $\mathrm{V}(\mathcal{X})$ is minimal, then

$$\sum x_i \geq \sum a_i$$

- Subtract $\sum_{\mathrm{V}_i > 0} u_i$ from both sides

$$\sum (x_i - u_i) \geq \sum \mathrm{V}_i = \mathrm{V}(\mathcal{X}) > 0$$

What happens when $\mathrm{V}(\mathcal{X})$ is minimal?

- Suppose $\mathrm{V}(\mathcal{X})$ is minimal and $\mathrm{V}(\mathcal{X}) > 0$
- Suppose $a_i > u_i$ for all $\mathrm{V}_i > 0$
- $\mathrm{V}_i = (a_i - u_i) > 0$
- But, $0 \geq (x_i - u_i)$
- If $\mathrm{V}(\mathcal{X})$ is minimal, then

$$\sum x_i \geq \sum a_i$$

- Subtract $\sum_{\mathrm{V}_i > 0} u_i$ from both sides

$$\sum (x_i - u_i) \geq \sum \mathrm{V}_i = \mathrm{V}(\mathcal{X}) > 0$$

- But....

$$0 \geq \sum_{i \in \mathcal{B}} (x_i - u_i)$$

WHAT HAPPENS WHEN $V(\mathcal{X})$ IS MINIMAL?

CONTINUED

- Can extract a conflict using $\sum_{V_i>0} T_i$
- Conflict may not be minimal

TABLE OF CONTENTS

LEVERAGING LP & MIP

- SOISimplex added optimization to Simplex for DPLL($\mathcal{T}$)
- Linear Programming solvers perform both
  - feasibility checking and
  - optimization

Leveraging LP & MIP

- SOISimplex added optimization to Simplex for DPLL($\mathcal{T}$)
- Linear Programming solvers perform both
  - feasibility checking and
  - optimization
- Mixed Integer Programming = LP + IsInt(x_i) constraints

LEVERAGING LP & MIP

- SOISimplex added optimization to Simplex for DPLL($\mathcal{T}$)
- Linear Programming solvers perform both
  - feasibility checking and
  - optimization
- Mixed Integer Programming = LP + IsInt(x_i) constraints
- Decades of research: fast by SMT standards

LEVERAGING LP & MIP

- SOISimplex added optimization to Simplex for DPLL($\mathcal{T}$)
- Linear Programming solvers perform both
  - feasibility checking and
  - optimization
- Mixed Integer Programming = LP + IsInt(x_i) constraints
- Decades of research: fast by SMT standards
- Can SMT leverage LP?
  - **Trusting** LP solver **[YM06]**
  - Check each $\mathcal{T}$-conflict used **[FaureNOR08]**
  - **FORCEDPIVOT procedure [Caminha˙Monniaux˙PAAR2012, Monniaux˙CAV09]**
  - All use LP solver as main $\mathcal{T}_{\mathbb{R}}$-solver

Reseeding Simplex States

General Approach

- Call an external off-the-shelf **untrusted** Simplex LP solver
- Reseed the state of the exact precision solver
- Only when it is likely to help
- Implemented with GLPK

RESEEDING THE SIMPLEX STATE

If the $\mathbb{R}$-relaxation is hard, try the following:

1. Construct an approximate problem from exact

$$T\vec{\mathcal{X}} = 0, \vec{l} \leq \vec{\mathcal{X}} \leq \vec{u} \implies \widetilde{T}\vec{\mathcal{X}} = 0, \widetilde{l} \leq \vec{\mathcal{X}} \leq$$

2. Call untrusted floating point Simplex solver on $\widetilde{T}, \widetilde{l}, \widetilde{u}$
3. Get back untrusted $\widetilde{a}$ and $\widetilde{\mathcal{B}}$
4. Convert floating point $\widetilde{a}$ into $a^{massage}$ ($\mathcal{X} \to \mathbb{Q}$)
5. RESEED($a^{massage}, \widetilde{\mathcal{B}}$) to get a new a and T
6. Call exact precision Simplex

MASSAGING ASSIGNMENTS

- Suppose we directly attempted to use $\widetilde{a}$.
- Each row must satisfy:

$$a_i = \sum T_{i,j} a_j$$

- Many variables have assignments near the bounds
- Many slack variables are entailed to be 0 (in practice)
- Get in a Simplex "friendly" state

Massaging Assignments

Floats to Rationals

$r \leftarrow \textsc{DiophantineApprox}(\widetilde{a}_i, D)$

if $|r - a_i| \leq \epsilon$ **then** $r \leftarrow a_i$

if $x \in \mathcal{X}_{\mathbb{Z}}$ and $|r - \lfloor r \rceil| \leq \epsilon$ **then** $r \leftarrow \lfloor r \rceil$

if $r > u_i$ or $|r - u_i| \leq \epsilon$ **then** $r \leftarrow u_i$

else if $r < l_i$ or $|r - l_i| \leq \epsilon$ **then** $r \leftarrow l_i$

$a_i^{massage} \leftarrow r$

RESEEDING SIMPLEX $(a^{massage}, \widetilde{\mathcal{B}})$

for all $j \in \mathcal{N}$ **do** UPDATE$(j, \cdot)$ s.t. $a_j = a_j^{massage}$

$\mathcal{B}_{want} \leftarrow \mathcal{N} \cap \widetilde{\mathcal{B}}$

repeat

  if any row conflict **then** **return Unsat**

  if $l \leq a \leq u$ **then** **return Sat**

  select i, k s.t. $k \in \mathcal{B}_{want}$, $i \notin \widetilde{\mathcal{B}}$, $T_{i,k} \neq 0$, and $\mathrm{V}_i > 0$

  if no such $\langle i, k \rangle$ **then**

    return Unknown ▷ $\widetilde{\mathcal{B}}$ is not valid basis

  else

    PIVOT(i, k) and UPDATE$(i, \cdot)$ s.t. $a_i = a_i^{massage}$

until $\mathcal{B}_{want} = \varnothing$

return Unknown ▷ Call Simplex

ReseedingSimplex $(a^{massage}, \widetilde{\mathcal{B}})$

for all $j \in \mathcal{N}$ **do** $\textsc{Update}(j, \cdot)$ s.t. $a_j = a_j^{massage}$

$\mathcal{B}_{want} \leftarrow \mathcal{N} \cap \widetilde{\mathcal{B}}$

repeat

  if any row conflict **then** **return Unsat**

  if $l \leq a \leq u$ **then** **return Sat**

  select i, k s.t. $k \in \mathcal{B}_{want}$, $i \notin \widetilde{\mathcal{B}}$, $T_{i,k} \neq 0$, and $\mathrm{V}_i > 0$

  if no such $\langle i, k \rangle$ **then**

    return Unknown ▷ $\widetilde{\mathcal{B}}$ is not valid basis

  else

    $\textsc{Pivot}(i, k)$ and $\textsc{Update}(i, \cdot)$ s.t. $a_i = a_i^{massage}$

until $\mathcal{B}_{want} = \varnothing$

return Unknown ▷ Call Simplex

More robust with SOI Simplex **[KBD13]**

Move $\langle \texttt{QF\_LRA}, LP \rangle \to \langle \texttt{QF\_LIRA}, MIP \rangle$

- Partition variables $\mathcal{X}$ into $\mathcal{X}_{\mathbb{R}} \cup \mathcal{X}_{\mathbb{Z}}$

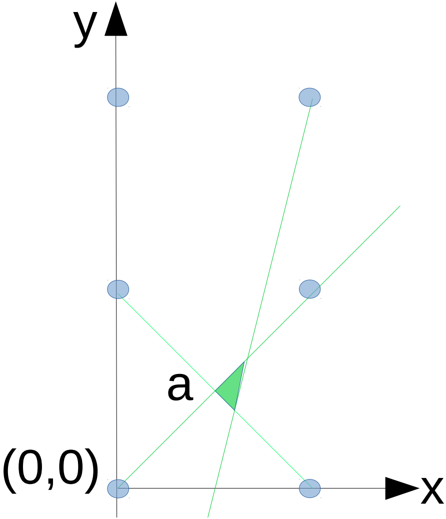
Move $\langle \texttt{QF\_LRA}, LP \rangle \rightarrow \langle \texttt{QF\_LIRA}, MIP \rangle$

- Partition variables $\mathcal{X}$ into $\mathcal{X}_{\mathbb{R}} \cup \mathcal{X}_{\mathbb{Z}}$
- $\mathbb{R}$-relaxation treat all $\mathcal{X}$ as $\mathcal{X}_{\mathbb{R}}$
- a is **integer-compatible** if $\forall x_i \in \mathcal{X}_{\mathbb{Z}}$, then $a_i \in \mathbb{Z}$

MOVE $\langle \texttt{QF\_LRA}, LP \rangle \rightarrow \langle \texttt{QF\_LIRA}, MIP \rangle$

- Partition variables $\mathcal{X}$ into $\mathcal{X}_{\mathbb{R}} \cup \mathcal{X}_{\mathbb{Z}}$
- $\mathbb{R}$-relaxation treat all $\mathcal{X}$ as $\mathcal{X}_{\mathbb{R}}$
- a is **integer-compatible** if $\forall x_i \in \mathcal{X}_{\mathbb{Z}}$, then $a_i \in \mathbb{Z}$
- MIP is new for SMT

Another Example: Visually



$$\begin{array}{rcrcl} x & + & y & \geq & 1 \\ x & - & y & \geq & 0 \\ 4x & - & y & \leq & 2 \end{array}$$

$$\begin{bmatrix} a_x \\ a_y \end{bmatrix} = \begin{bmatrix} \frac{1}{2} \\ \frac{1}{2} \end{bmatrix}$$

Branches and Cuts

Refining $\mathbb{Z}$-infeasible assignments

- Branch:

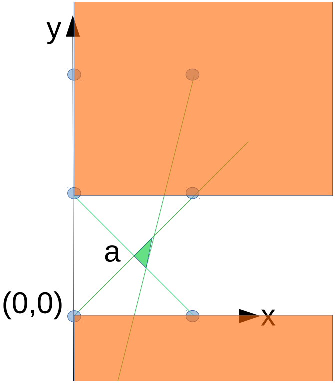
$$\frac{x_i \in \mathcal{X}_{\mathbb{Z}} \qquad \alpha \in \mathbb{R}}{x_i \leq \lfloor \alpha \rfloor \vee x_i \geq \lceil \alpha \rceil}$$

- Cut: $\sum c_i x_j \geq d$ such that
  - $\{l_i\} \models_{\mathbb{RZ}} \sum c_j x_j \geq d$
  - $\{l_i\} \not\models_{\mathbb{R}} \sum c_j x_j \geq d$
  - $\{x_j = a_j\} \not\models \sum c_j x_j \geq d$ (*)

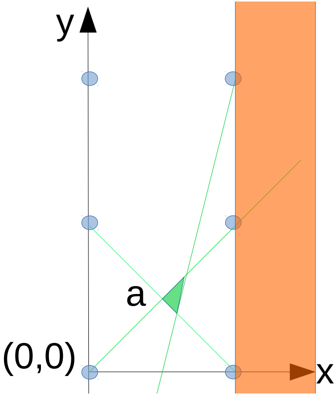
Branches and Cuts

Visually

Branch: $y \geq 1 \vee y \leq 0$



Cut: $\{\cdots\} \models_{\mathbb{RZ}} x \geq 1$



Branch-and-cut Solvers

Most SMT solvers and many MIP solvers

1. Treat all of $\mathcal{X}$ as if they were $\mathcal{X}_{\mathbb{R}}$
2. Solve the $\mathbb{R}$-relaxation
3. If unsat, return $\mathbb{R}$-conflict[s]
4. If $\mathbb{R}$-relaxation is **Sat** and a is $\mathbb{Z}$-compatible, return a
5. [Heuristically] try to derive a cut.
   If successful, add the cut $\sum c_j x_j \geq d$, and goto (1)
6. Branch on some $x_i \in \mathcal{X}_{\mathbb{Z}}$ with $a_i \notin \mathbb{Z}$

Branch-and-cut Solvers

Most SMT solvers and many MIP solvers

1. Treat all of $\mathcal{X}$ as if they were $\mathcal{X}_{\mathbb{R}}$
2. Solve the $\mathbb{R}$-relaxation
3. If unsat, return $\mathbb{R}$-conflict[s]
4. If $\mathbb{R}$-relaxation is **Sat** and a is $\mathbb{Z}$-compatible, return a
5. [Heuristically] try to derive a cut.
   If successful, add the cut $\sum c_j x_j \geq d$, and goto (1)
6. Branch on some $x_i \in \mathcal{X}_{\mathbb{Z}}$ with $a_i \notin \mathbb{Z}$
   Splitting-on-Demand in SMT

MIP Answers

What are the possible answers for `QF_LIA` and `QF_LIRA`?

- $\mathbb{R}$-infeasible
- $\mathbb{R}$-feasible and $\mathbb{Z}$-feasible
- $\mathbb{R}$-feasible and $\mathbb{Z}$-infeasible

MIP Answers

What are the possible answers for `QF_LIA` and `QF_LIRA`?

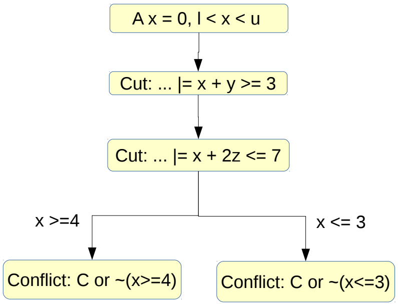
- $\mathbb{R}$-infeasible
- $\mathbb{R}$-feasible and $\mathbb{Z}$-feasible
  Same reseeding trick as $\mathbb{R}$-feasible
- $\mathbb{R}$-feasible and $\mathbb{Z}$-infeasible

MIP Answers

What are the possible answers for `QF_LIA` and `QF_LIRA`?

- $\mathbb{R}$-infeasible
- $\mathbb{R}$-feasible and $\mathbb{Z}$-feasible
  Same reseeding trick as $\mathbb{R}$-feasible
- $\mathbb{R}$-feasible and $\mathbb{Z}$-infeasible

INFEASIBLE BRANCH-AND-CUT EXECUTIONS



- Leaves are conflicts
- Internal nodes are branches

$$x_i \leq \lfloor \alpha \rfloor \vee x_i \geq \lceil \alpha \rceil \quad \text{if } x_i \in \mathcal{X}_{\mathbb{Z}}$$

- Nodes have cuts

$$\{l_i\} \models_{\mathbb{RZ}} \sum c_j x_j \geq d$$

Replaying the MIP Execution

- Minimizes changes to the MIP solver's search

Replaying the MIP Execution

- Minimizes changes to the MIP solver's search
- Instrument GLPK to print hints about:
  branch, unsat leaves, and derivations of cutting planes

Replaying the MIP Execution

- Minimizes changes to the MIP solver's search
- Instrument GLPK to print hints about:
  branch, unsat leaves, and derivations of cutting planes
- Repeat "the big steps" in the SMT solver

Replaying the MIP Execution

- Minimizes changes to the MIP solver's search
- Instrument GLPK to print hints about:
  branch, unsat leaves, and derivations of cutting planes
- Repeat "the big steps" in the SMT solver
- Reconstruct the Resolution+Cutting Planes proof
- Resolution removes branching literals

REPLAYING THE MIP EXECUTION

- Minimizes changes to the MIP solver's search
- Instrument GLPK to print hints about:
  branch, unsat leaves, and derivations of cutting planes
- Repeat "the big steps" in the SMT solver
- Reconstruct the Resolution+Cutting Planes proof
- Resolution removes branching literals
- Any failure can be safely dropped
- Success is a conflict

Cutting Planes

- Hint is used to instantiate a cutting plane procedure
- Proof must tightly match to get the "same" cut
- White-box knowledge and detailed hints
- Support for Gomory (easy) and MK-MIR (hard) cuts

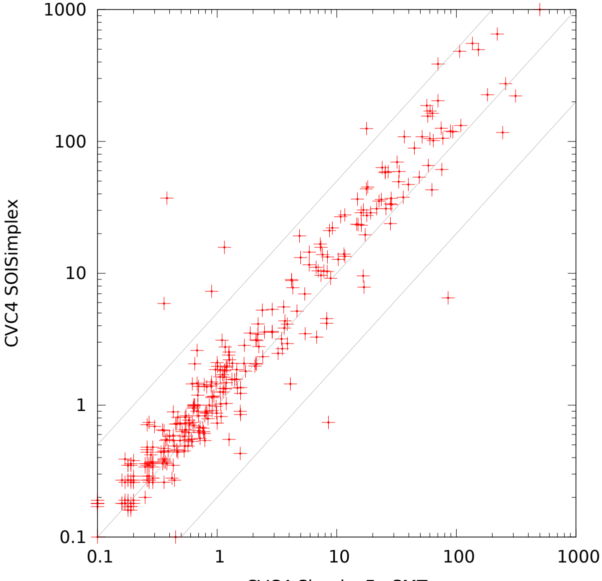
Table of Contents

Two Groups of Experiments

1. Compare: SOISimplex to SimplexForDPLL($\mathcal{T}$)
2. Everything: SOISimplex + Reseed + Replay

SOISimplex versus SimplexForDPLL($\mathcal{T}$)

SMT-LIB QF_LRA



Below $x = y$ means SOISimplex is faster

SOISimplex versus SimplexForDPLL($\mathcal{T}$)

Number Solved

	SOI	Z3	yices2	mathsat
QFLRA (634)	618	620	619	608
`latendresse` (18)	18	8	10	10

SOISimplex versus SimplexForDPLL($\mathcal{T}$)

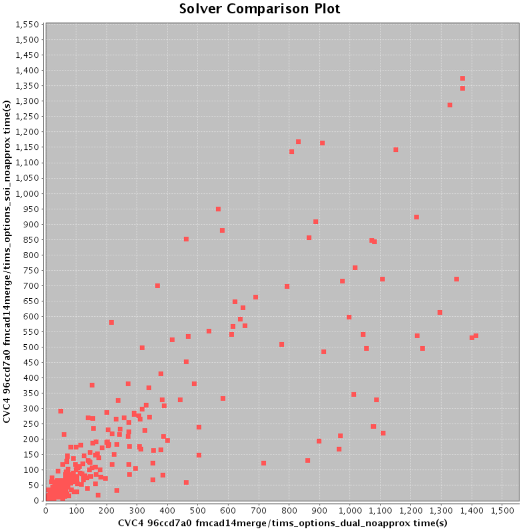
Pivots needed

- $\sim 95\%$ of calls to theory solver need 0 simplex round
- $\sim 1.8\%$ of calls to the theory solver need 1 simplex round
- $\sim 2.5\%$ of calls to the theory solver need $[2 - 10]$ rounds
- This is about 50% of the simplex rounds in total

Most problems in March 2014 SMT-LIB don't need SOISimplex

SOISimplex versus SimplexForDPLL($\mathcal{T}$)

New family LassoRanker



SOISimplex + Reseed + Replay Results

SMT Solver Comparison

QF_LRA

			SOI+MIP		CVC4		yices2		mathsat5		Z3	
set	# inst.	# sel.	solved	time (s)	solved	time (s)	solved	time (s)	solved	time (s)	solved	time (s)
QF_LRA	634	634	627	6199	618	7721	620	5265	612	10814	615	5696
latendresse	18	18	18	129	10	44	12	85	10	99	0	0
miplib	42	37	30	1530	21	3037	23	2730	17	5682	18	2435
DTP-*	91	4	4	4	4	4	4	0	4	2	4	1
total	-	41	34	1534	25	3041	27	2330	21	5684	22	2436

(AR) = Applied either Reseed or Replay, $\mathbf{K} = 1000$, & SOI+MIP is CVC4 1.4 with options

SMT Solver Comparison

QF_LIA ¬-CONJUNCTIVE

			SOI+MIP		CVC4		mathsat5		Z3		altergo	
set	# inst.	# sel.	solved	time (s)	solved	time (s)	solved	time (s)	solved	time (s)	solved	time (s)
everything												
QF_LIA	5882	5882	5738	97**K**	5540	117**K**	5697	88**K**	5513	94**K**	5188	264**K**
conjuncts	1303	1303	1249	11**K**	1068	31**K**	1154	33**K**	1039	19**K**	1232	2055
(AR) ¬ conjuntive												
convert	319	282	208	9646	193	9343	274	1876	282	118	166	272
bofill-*	652	460	460	5401	458	4490	460	1519	460	2060	67	55
CIRC	51	11	11	0	11	0	11	0	11	0	11	0
calypto	37	37	37	3	37	3	37	6	36	5	35	24
nec-smt	2780	207	207	17**K**	207	18**K**	207	17**K**	201	7209	184	23**K**
wisa	5	1	1	0	1	0	1	1	1	0	1	0
total	-	998	924	32**K**	907	31**K**	990	21**K**	991	9392	464	24**K**

(AR) = Applied either RESEED or REPLAY, **K** = 1000, & SOI+MIP is CVC4 1.4 with options

AltErgo is using **[bobot12ijcar]**

SMT Solver Comparison

QF_LIA conjunctive

			SOI+MIP		CVC4		mathsat5		Z3		altergo	
set	# inst.	# sel.	solved	time (s)	solved	time (s)	solved	time (s)	solved	time (s)	solved	time (s)
everything												
QF_LIA	5882	5882	5738	97**K**	5540	117**K**	5697	88**K**	5513	94**K**	5188	264**K**
conjuncts	1303	1303	1249	11**K**	1068	31**K**	1154	33**K**	1039	19**K**	1232	2055
(AR) conjuntive												
dillig	233	189	189	49	157	9823	188	7185	166	1269	189	5
miplib2003	16	8	4	307	4	1283	5	354	5	1089	0	0
prime-cone	37	37	37	2	37	2	37	1	37	2	37	1
slacks	233	188	166	61	93	2003	119	4741	90	1994	188	84
CAV_2009	591	424	424	69	346	10**K**	421	10**K**	354	2759	423	323
cut_lem.	93	74	62	9581	64	6865	45	9472	38	5858	74	267
total	-	920	882	10**K**	701	30**K**	815	31**K**	690	12**K**	911	680

(AR) = Applied either RESEED or REPLAY, **K** = 1000, & SOI+MIP is CVC4 1.4 with options

Comparison with conjunctive solvers

			SOI+MIP		cutsat		scip		glpk	
set	# inst.	# sel.	solved	time (s)	solved	time (s)	solved	time (s)	solved	time (s)
conjuncts	1303	1303	1249	11130	1018	35330	1255	7164	1173	8895
(AR) conjuntive										
dillig	233	189	189	49	166	5840	189	42	189	3
miplib2003	16	8	4	307	6	146	7	17	6	295
prime-cone	37	37	37	2	37	4	37	1	37	0
slacks	233	188	166	61	96	6324	161	2361	101	11
CAV_2009	591	424	424	69	377	17015	424	105	424	6
cut_lemmas	93	74	62	9581	15	1887	72	1757	71	760
total	-	920	882	10069	697	31216	890	4283	828	1075

(AR) = Applied either RESEED or REPLAY, $\mathbf{K} = 1000$, & SOI+MIP is CVC4 1.4 with options

cutsat is using **[JovanovicM11]**

QF_LIA Reseed and Replay success rates

			Reseed		Replay	
set	# inst.	solve int calls	attempts	successes	attempts	successes
QF_LIA	1806	3873	2559	1058	652	425
convert	208	2130	1356	1	178	3
bofill-scheduling	460	254	245	245	0	0
CIRC	11	85	6	5	79	77
calypto	37	375	77	23	293	278
wisa	1	1	1	1	0	0
dillig	189	228	225	185	3	2
miplib2003	4	10	3	3	5	4
prime-cone	37	37	19	19	18	18
slacks	166	195	168	162	3	3
CAV_2009	424	469	459	414	8	7
cut_lemmas	62	89	0	0	65	33

Only includes solved instances

SMT-COMP'14

- CVC4 won `QF_LRA`
- [CVC4-with-bugfix] solved the most `QF_LIA` benchmarks
- Won a number of combination & quantified divisions

Also won `Typed First-order Theorems +*-/` at CASCJ7

Table of Contents

SOISimplex versus SimplexForDPLL($\mathcal{T}$)

Final Word

SOISimplex

- Minimize $V(\mathcal{X})$
- More expensive analysis
- Fewer rounds on average*

SimplexForDPLL($\mathcal{T}$)

- Greedily fixes some $V_i > 0$
- Cheaper analysis
- Faster on easy instances

Replay Results

What happened on the `convert` family?

- MIP solver is wrong about feasibility too often
- Variables are in bounds up to a "dual gap"
  - Intuitively: Let a_i violate u_i by a litle where little is scaled by the size of the numbers
  - Numerically stability of floating points
- Gap is too large for `QF_LIA` bit-extracts for $\sim m + n > 40$

$$x = 2^m y + z \land z \in [0, 2^m), y \in [0, 2^n), x \in [0, 2^{m+n})$$

- Decreasing the gap leads to cycling [in practice]
- Need bigger floats if MIP solver is to work

Replay & Reseed Summary

- Integrated a floating point LP/MIP solver (GLPK) (Backup. Not the main theory solver!)

Replay & Reseed Summary

- Integrated a floating point LP/MIP solver (GLPK) (Backup. Not the main theory solver!)
- Reseeding Simplex (1 week to implement[*])
  - Gives candidate assignments and gives $\mathbb{R}$-relaxation conflicts
  - Massaging floating points is really important

REPLAY & RESEED SUMMARY

- Integrated a floating point LP/MIP solver (GLPK)
  (Backup. Not the main theory solver!)
- Reseeding Simplex (1 week to implement[*])
  - Gives candidate assignments and gives $\mathbb{R}$-relaxation conflicts
  - Massaging floating points is really important
- Replaying MIP conflicts (significantly more effort)
  MIP must be white-box and must log proofs!

Replay & Reseed Summary

- Integrated a floating point LP/MIP solver (GLPK)
  (Backup. Not the main theory solver!)
- Reseeding Simplex (1 week to implement[*])
  - Gives candidate assignments and gives $\mathbb{R}$-relaxation conflicts
  - Massaging floating points is really important
- Replaying MIP conflicts (significantly more effort)
  MIP must be white-box and must log proofs!
- Overall performance is good
- But there are known problems

Future Work

- SOISimplex
  - Help Replay & Reseed
  - Mix in primal optimization
- Replay & Reseed
  - Optimization Modulo Theories
  - Different heuristics for cuts
  - Logging and replaying approximate Farkas's lemma instances[**Neumaier2004**]
  - *k*-precision floating Simplex solver for SMT[**CookKSW13**]

Conference Papers

- "Leveraging Linear and Mixed Integer Programming for SMT". Tim King, Clark Barrett and Cesare Tinelli. [to appear] FMCAD '14
- "Finding Minimum Type Error Sources". Zvonimir Pavlinovic, Tim King, Thomas Wies. OOPSLA '14
- "Simplex with Sum of Infeasibilities for SMT". Tim King, Clark Barrett and Bruno Dutertre FMCAD '13
- "CVC4." Clark Barrett, Chris Conway, Morgan Deters, Liana Hadarean, Dejan Jovanović, Tim King, Andrew Reynolds, and Cesare Tinelli. CAV '11

REFERENCES I