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 \(\text{QF}_{\text{LRA}}\)

- Sum of Infeasibilities for SMT [FMCAD13]
  Robust decision procedure for \(\text{QF}_{\text{LRA}}\)

- Leverage LP/MIP Solvers

- Experiments
Big Ideas of Today’s Lecture

- SAT, SMT & DPLL(τ)
  How to combine CDCL + Simplex?
- Simplex for DPLL(τ)
  SOTA decision procedure for $\text{QF}_\text{LRA}$
- Sum of Infeasibilities for SMT [FMCAD13]
  Robust decision procedure for $\text{QF}_\text{LRA}$
- Leverage LP/MIP Solvers [FMCAD14]
  Accelerate exact precision solver
- Experiments
Table of Contents

Satisfiability Modulo Theories

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

Sum Of Infeasibilities Simplex [FMCAD13]

Reseed & Replay [FMCAD14]

Empirical Results

Conclusion
Satisfiability Modulo Theories

- Theories enforce the semantics of the syntax

- $\mathcal{T}_R$: theory of reals
  
  Domain of values is $\mathbb{R}$
  "+" is mathematical $+$
  "0" is mathematical $0$
  "<" is mathematical $<$

- $\mathcal{T}_Z$: theory of integers
### Satisfiability Modulo Theories

- **Theories** enforce the semantics of the syntax

- \( \mathcal{T}_R \): theory of reals
  - Domain of values is \( \mathbb{R} \)
  - “+” is mathematical +
  - “0” is mathematical 0
  - “<” is mathematical <
  - ...

- \( \mathcal{T}_Z \): theory of integers

---

**SMT Problem**

Does there exist a variable assignment \( a \) for the theory \( \mathcal{T} \) such that the formula \( \phi \) evaluates to **true**?
**QF_LRA Example**

*Quantifier-Free Linear Real Arithmetic*

\[
\phi \equiv (y \leq 4) \land (y \geq 5 \lor x + y \leq 6) \land (x > 2 \lor x - y \geq 1)
\]

Is there an assignment that makes \( \phi \) evaluate to true?

\[a : \mathcal{X} \rightarrow \mathbb{R}\]
DPLL(\(\mathcal{T}\))

SMT Solver Framework

\begin{align*}
y & \leq 4 \\
y & \geq 5 \lor x + y & \leq 6 \\
x & > 2 \lor x - y & \geq 1
\end{align*}
DPLL(\(\mathcal{T}\))
SMT Solver Framework

\[
\begin{align*}
  y &\leq 4 \\
  y &\geq 5 \lor x + y \leq 6 \\
  x &> 2 \lor x - y \geq 1
\end{align*}
\]
DPLL(\(\mathcal{T}\))

**SMT Solver Framework**

\[
\begin{align*}
y & \leq 4 \\
y & \geq 5 \lor x + y & \leq 6 \\
x & > 2 \lor x - y & \geq 1
\end{align*}
\]
DPLL(\(\mathcal{T}\))

SMT Solver Framework

\[
\begin{align*}
  y &\leq 4 \\
  y &\geq 5 \\
  x &> 2 \\
  x + y &\leq 6 \\
  x - y &\geq 1 
\end{align*}
\]
**DPLL(\(T\))**

**SMT Solver Framework**

SAT Solver

CDCL

**assertions**

**propagations**

Theory Solver

\[ y \leq 4 \]
\[ x + y \leq 6 \]

\[ y \geq 5 \lor x + y \leq 6 \]
\[ x > 2 \lor x - y \geq 1 \]
**DPLL(\(\mathcal{T}\))**

**SMT Solver Framework**

\[
\begin{align*}
    y & \leq 4 \\
    y & \geq 5 \lor x + y \leq 6 \\
    x & > 2 \lor x - y \geq 1
\end{align*}
\]
DPLL(\(\mathcal{T}\))

SMT Solver Framework

\[
\begin{align*}
y & \leq 4 \\
y & \geq 5 \lor x + y & \leq 6 \\
x & > 2 \lor x - y & \geq 1 \\
x & \leq 2 \lor x + y & < 6 \lor y > 4
\end{align*}
\]
DPLL(\(\mathcal{T}\))

SMT Solver Framework

\[
\begin{align*}
y & \leq 4 \\
y & \geq 5 \lor x + y & \leq 6 \\
x & > 2 \lor x - y & \geq 1 \\
x & \leq 2 \lor x + y & < 6 \lor y > 4
\end{align*}
\]
**DPLL(\(\mathcal{T}\))**

**SMT Solver Framework**

\[
\begin{align*}
y & \leq 4 \\
y & \geq 5 \\
x & \geq 2 \\
x & \leq 2 \\
x + y & < 6 \\
x - y & \geq 1 \\
y & > 4
\end{align*}
\]
Table of Contents

Satisfiability Modulo Theories

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

Sum Of Infeasibilities Simplex [FMCAD13]

Reseed & Replay [FMCAD14]

Empirical Results

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

\[
\begin{align*}
x + y & \leq 6 \\
x - y & \geq -1 \\
y & \leq 4
\end{align*}
\]
evaluate to true?
**VISUALLY**

\[
\begin{align*}
  x - y & \geq -1 \\
  y & \leq 4 \\
  x + y & \leq 6
\end{align*}
\]

\[
\begin{align*}
  x + y & \leq 6 \\
  x - y & \geq -1 \\
  y & \leq 4
\end{align*}
\]
**Visually**

\[
x - y \geq -1
\]

\[
y \leq 4
\]

\[
x + y \leq 6
\]

(3,3)

\[
\begin{bmatrix}
  a_x \\
  a_y
\end{bmatrix} =
\begin{bmatrix}
  3 \\
  3
\end{bmatrix}
\]

How?
Simplex for DPLL(\(\mathcal{T}\)) Search
**Simplex for DPLL(\(\mathcal{T}\)) Search**

\[
\begin{align*}
  x - y & \geq -1 \\
  y & \leq 4 \\
  x + y & \leq 6
\end{align*}
\]
**Simplex for DPLL(\(\mathcal{T}\)) Search**

\[
\begin{align*}
    y &\leq 4 \\
    x - y &\geq -1 \\
    x + y &\leq 6 \\
    x &\leq 6
\end{align*}
\]
SIMPLEX FOR DPLL(\(\mathcal{T}\)) SEARCH

\[
\begin{align*}
x - y & \geq -1 \\
y & \leq 4 \\
x + y & \leq 6
\end{align*}
\]
**Preprocessing**

- Introduce a fresh \( s_i \) for each \( \sum T_{i,j} \cdot x_j \)

- Literals are of the form:

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

and \( s_i \) appears in exactly 1 equality.

- Collect into:

\[
T \vec{x} = 0 \quad \vec{l} \leq \vec{x} \leq \vec{u}
\]
Preprocessing

- Introduce a fresh $s_i$ for each $\sum T_{i,j} \cdot x_j$

- Literals are of the form:

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

and $s_i$ appears in exactly 1 equality.

- Collect into:

$$T\vec{x} = 0 \quad \vec{l} \leq \vec{x} \leq \vec{u}$$
**Preprocessing**

- Introduce a fresh $s_i$ for each $\sum T_{i,j} \cdot x_j$

- Literals are of the form:

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

and $s_i$ appears in exactly 1 equality.

- Collect into:

$$T \vec{x} = 0 \quad \vec{l} \leq \vec{x} \leq \vec{u}$$
**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:

- $a_j += \delta$

- for all $i \in \mathcal{B}$:
  \[ a_i += T_{i,j} \cdot \delta. \]
**UPDATING NONBASIC VARIABLES**

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

- $a_j + = \delta$

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

**Add the Invariant**

The nonbasic variables satisfy their bounds.
**PIVOT**($i,j$)

**Move Variables In/Out of $B$**

### 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.
**PIVOT**\((i, j)\)

**Move Variables In/Out of** \(B\)

** Preconditions**

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

\(\text{PIVOT}(i, j)\) 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 \(B\)**

**Preconditions**

Given \(x_i\) basic, \(x_j\) nonbasic, and \(T_{i,j} \neq 0\), \(\text{PIVOT}(i, j)\) 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**

\(\text{PIVOT}(i, j)\) preserves \(Ta = 0\).
TABLEAU EXAMPLE

\[
\begin{align*}
  x + y &\leq 6 \\
  x - y &\geq -1 \\
  y &\leq 4 
\end{align*}
\]
**Tableau Example**

\[ s_1 = x + y \]
\[ s_2 = x - y \]

\[ s_1 \geq 6 \land s_2 \geq -1 \land y \leq 4 \]
**Tableau Example**

\[ s_1 = x + y \]
\[ s_2 = x - y \]
\[ s_1 \geq 6 \land s_2 \geq -1 \land y \leq 4 \]

\[
T\vec{x} = \begin{bmatrix} -1 & 0 & 1 & 1 \\ 0 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ x \\ y \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}
\]

\[ B = \{ s_1, s_2 \} , N = \{ x, y \} \]
**Tableau Example**

\[ s_1 = x + y \]
\[ s_2 = x - y \]
\[ s_1 \geq 6 \land s_2 \geq -1 \land y \leq 4 \]

\[
T\vec{x} = \begin{bmatrix}
-1 & 0 & 1 & 1 \\
0 & -1 & 1 & -1 \\
\end{bmatrix}
\begin{bmatrix}
s_1 \\
s_2 \\
x \\
y \\
\end{bmatrix}
= \begin{bmatrix}
0 \\
0 \\
\end{bmatrix}
\]

\[ \mathcal{B} = \{ s_1, s_2 \}, \mathcal{N} = \{ x, y \} \]
**Tableau Example**

\[
\begin{align*}
s_1 & = x + y \\
s_2 & = x - y \\
s_1 & \geq 6 \land s_2 \geq -1 \land y \leq 4
\end{align*}
\]

\[
T\tilde{x} = \begin{bmatrix} -1 & 0 & 1 & 1 \\ 0 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ x \\ y \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}
\]

\[
\mathcal{B} = \{s_1, s_2\}, \quad \mathcal{N} = \{x, y\}
\]
**SIMPLEX FOR DPLL(\(\mathcal{T}\))**

**PSEUDOCODE**

```plaintext
while \(i \in B\) s.t. \(a_i > u_i\) or \ldots\) 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 \(i \in B\) 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. \(a_i \leftarrow u_i\)

\text{PIVOT}(i, j) \quad \triangleright O(|T|)
**Simplex for DPLL(\(\mathcal{T}\)) Search**

Greedily fix

\[ x + y \leq 6 \]

ignoring

\[ x - y \geq -1 \quad \text{and} \quad y \leq 4 \]
**Simplex for DPLL($\mathcal{T}$) Search**

Greedily fix

\[
x + y \leq 6
\]

ignoring

\[
x - y \geq -1 \quad \text{and} \quad y \leq 4
\]
CONFLICT DETECTION

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

$$x_i = \sum_{j \in N} T_{i,j} x_j$$
**CONFLICT DETECTION**

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

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

- 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 $i \in B$

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

- 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

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

- Select $x_i$ s.t. $a_i > u_i$ and $i \in B$
  \[ x_i = \sum_{j \in \mathcal{N}} T_{i,j} x_j \]

- 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
  \[ x_i = \sum_{j \in \mathcal{N}} T_{i,j} x_j \land \bigwedge_{T_{i,j} > 0} x_j \geq l_j \land \bigwedge_{T_{i,k} < 0} x_k \geq u_k \]

- But $u_i \geq x_i \geq a_i > u_i$!
Thus, the following is \textbf{Unsat} in $\widetilde{T}_R$:

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

SMALL CONTRIBUTION

- \( \forall i \in B \) track the cardinalities of the sets:

\[
J = \{ j \mid T_{i,j} > 0, a_j = l_j \} \quad K = \{ k \mid T_{i,k} < 0, a_k = u_k \}
\]
Eager Conflict Detection

Small Contribution

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

  $J = \{j | T_{i,j} > 0, a_j = l_j\}$  \hspace{1cm}  $K = \{k | T_{i,k} < 0, a_k = u_k\}$

- 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 $\rightarrow \mathcal{O}(1)$-amortized conflict detection

Never miss conflicts!
Eager Conflict Detection

Small Contribution

- \( \forall i \in B \) 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

- ∀i ∈ B track the cardinalities of the sets:

\[ J = \{j \mid T_{i,j} > 0, a_j = l_j\} \]
\[ K = \{k \mid T_{i,k} < 0, a_k = u_k\} \]

- 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(\(T\))**

*With Eager Conflict Detection*

check for row conflicts

**while** \(i \in B\) s.t. \(a_i > u_i\) or \(\ldots\) 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. \(a_j \leftarrow u_j\)

\(\text{PIVOT}(i, j)\)

check for row conflicts
**Simplex for DPLL(τ)**

**With Eager Conflict Detection**

check for row conflicts

while \( i \in \mathcal{B} \) s.t. \( a_i > u_i \) or \( \ldots \) 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

- Satisfiability Modulo Theories
- Simplex for DPLL\((\mathcal{T})\)
- Sum Of Infeasibilities Simplex [FMCAD13]
- Reseed & Replay [FMCAD14]
- Empirical Results
- Conclusion
Simplex for DPLL(𝒯) Search

Reminder

Greedily fix

\[ x + y \leq 6 \]

ignoring

\[ x - y \geq -1 \quad \text{and} \quad y \leq 4 \]
**Sum of Infeasibilities**

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

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

- Sum of Infeasibilities:

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

- SOISIMPLEX minimizes $V(\mathcal{X})$ every round
SUM OF INFEASIBILITIES

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

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

- Sum of Infeasibilities:

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

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

- New for SMT
SOI Simplex Search

\[
\begin{align*}
x - y & \geq -1 \\
y & \leq 4 \\
x + y & \leq 6
\end{align*}
\]
SOISimplex Search

\[ \begin{align*}
    x - y & \geq -1 \\
    y & \leq 4 \\
    x + y & \leq 6
\end{align*} \]
SOISimplex Search

\[ \begin{align*}
  x - y & \geq -1 \\
  y & \leq 4 \\
  x + y & \leq 6
\end{align*} \]
**Direction of \( V(\mathcal{X}) \)**

\[
\begin{align*}
x - y & \geq -1 \\
y & \leq 4 \\
x + y & \leq 6
\end{align*}
\]

\[
\left\{ \begin{array}{ll}
a_i & \text{if } a_i > u_i \\
0 & \text{otherwise} \\
\ldots - a_i & \text{if } a_i < l_i \end{array} \right.
\]
**Direction of \( V(\mathcal{X}) \)**

\[
\begin{align*}
x - y &\geq -1 \\
y &\leq 4 \\
x + y &\leq 6 \\
0 &
\end{align*}
\]

\[
V(\mathcal{X}) = 0 \text{ iff } a \text{ is sat}
\]

\[
\left\{
\begin{array}{ll}
a_i \ldots & a_i > u_i \\
0 & \text{otherwise} \\
\ldots - a_i & a_i < l_i
\end{array}
\right.
\]
**SOISimplex** Highlevel

**Rough Sketch**

```plaintext
procedure SOISimplex

while V(χ) is not at a minimum do

select a variable x_j

update x_j s.t. V(χ) decreases and

\[ a_i \leftarrow u_i \text{ (or \ldots) for some } i \in \mathcal{B} \]

PIVOT(i, j)

▷ can check rows for conflicts

return (if (V(χ) = 0) then Sat else SoiQE())
```
SOI Simplex 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 $i \in B$

PIVOT($i, j$)

▷ can check rows for conflicts

return (if $(V(\mathcal{X}) = 0)$ then Sat else SoiQE())
SOISONPLEX HIGHLEVEL
Rough Sketch

procedure SOISONPLEX

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

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

PIVOT$(i, j)$

▷ can check rows for conflicts

return (if $(V(\mathcal{X}) = 0)$ then Sat else SoiQE())
BREAKPOINTS
WHERE $V(\chi)$ 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 \( V(x) \) changes*

\[
\begin{align*}
\delta & \in \{1, 3, 4\} \\
\end{align*}
\]
**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 UPDATE($j$, $\delta$) for each $\delta$

**return** the $\delta$ and corresponding $i$ with

the lowest $V(\mathcal{X})$ post UPDATE($j$, $\delta$)
**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 UPDATE($j, \delta$) for each $\delta$

**return** the $\delta$ and corresponding $i$ with

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

$V(\mathcal{X})$ monotonically decreases!
SOISIMPLEX
FILLING IN THE SKETCH

while \( V(x') \) is not at a minimum do

\( \langle i, \delta, j \rangle \leftarrow \text{SOISELECT()} \)

\text{UPDATE}(j, \delta)

\text{PIVOT}(i, j)

▷ can check rows for conflicts

return (if \( (V(x') = 0) \) then Sat else SoiQE())
**SOISimplex**

**Filling in the Sketch**

\[
\text{while } V(\mathcal{X}) \text{ is not at a minimum } \text{ do}
\]

\[\langle i, \delta, j \rangle \leftarrow \text{SOISELECT}()\]

UPDATE\((j, \delta)\)

PIVOT\((i, j)\)

\[\triangleright \text{ can check rows for conflicts}\]

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

**Filling in the Sketch**

```latex
\textbf{while} \ V(\mathcal{X}) \text{ is not at a minimum} \hspace{5mm} \text{do}

\langle i, \delta, j \rangle \leftarrow \text{SOISELECT()}

\text{UPDATE}(j, \delta)

\text{PIVOT}(i, j)

\triangleright \text{can check rows for conflicts}

\textbf{return} \ (\textbf{if} \ (V(\mathcal{X}) = 0) \ \textbf{then} \textbf{Sat} \ \textbf{else} \text{SoiQE()})
```
SOISIMPLEX
FILLING IN THE SKETCH

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

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

$\text{UPDATE}(j, \delta)$

$\text{PIVOT}(i, j)$

▷ 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 $V(\mathcal{X})$ 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
  \[ \sum x_i \geq \sum a_i \]
- Subtract $\sum_{V_i > 0} u_i$ from both sides
  \[ \sum (x_i - u_i) \geq \sum V_i = V(\mathcal{X}) > 0 \]
WHAT HAPPENS WHEN V(≰) IS MINIMAL?

▶ Suppose V(≰) is minimal and V(≰) > 0

▶ Suppose a_i > u_i for all V_i > 0

▶ V_i = (a_i - u_i) > 0

▶ But, 0 ≥ (x_i - u_i)

▶ If V(≰) is minimal, then

\[ \sum x_i \geq \sum a_i \]

▶ Subtract \( \sum_{V_i>0} u_i \) from both sides

\[ \sum (x_i - u_i) \geq \sum V_i = V(≰) > 0 \]

▶ But....

\[ 0 \geq \sum_{i \in B} (x_i - u_i) \]
**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

- Satisfiability Modulo Theories
- Simplex for DPLL($\mathcal{T}$)
- Sum Of Infeasibilities Simplex [FMCAD13]
- Reseed & Replay [FMCAD14]
- Empirical Results
- Conclusion
LEVERAGING LP & MIP

- SOISimplex added optimization to Simplex for DPLL(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]

  - **FORCED PIVOT** procedure
    [Caminha, Monniaux, PAAR2012, Monniaux, CAV09]

  - All use LP solver as main $\mathcal{T}_R$-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

\[
T \tilde{X} = 0, \tilde{l} \leq \tilde{X} \leq \tilde{u} \implies \tilde{T} \tilde{X} = 0, \tilde{l} \leq \tilde{X} \leq \tilde{u}
\]

2. Call untrusted floating point Simplex solver on \( \tilde{T}, \tilde{l}, \tilde{u} \)

3. Get back untrusted \( \tilde{a} \) and \( \tilde{B} \)

4. Convert floating point \( \tilde{a} \) into \( a^{\text{massage}} (\mathcal{X} \rightarrow \mathbb{Q}) \)

5. \text{RESEED}(a^{\text{massage}}, \tilde{B}) to get a new \( a \) and \( T \)

6. Call exact precision Simplex
Massaging Assignments

▶ Suppose we directly attempted to use $\tilde{a}$.

▶ 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**

\[ r \leftarrow \text{DiophantineApprox}(\tilde{a}_i, D) \]

\[ \text{if } |r - a_i| \leq \epsilon \text{ then } r \leftarrow a_i \]

\[ \text{if } x \in \mathcal{X}_\mathbb{Z} \text{ and } |r - \lfloor r \rfloor| \leq \epsilon \text{ then } r \leftarrow \lfloor r \rfloor \]

\[ \text{if } r > u_i \text{ or } |r - u_i| \leq \epsilon \text{ then } r \leftarrow u_i \]

\[ \text{else if } r < l_i \text{ or } |r - l_i| \leq \epsilon \text{ then } r \leftarrow l_i \]

\[ a_i^{\text{massage}} \leftarrow r \]
**Reseeding Simplex** \((a^{\text{massage}}, \tilde{B})\)

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

\(\mathcal{B}_{\text{want}} \leftarrow \mathcal{N} \cap \tilde{B}\)

Repeat

- If any row conflict then return Unsat
- If \(l \leq a \leq u\) then return Sat
- Select \(i, k\) s.t. \(k \in \mathcal{B}_{\text{want}}, i \notin \tilde{B}, T_{i,k} \neq 0,\) and \(V_i > 0\)
- If no such \(\langle i, k \rangle\) then
  - return Unknown \(\triangleright \tilde{B}\) is not valid basis
- Else
  - \(\text{PIVOT}(i, k)\) and \(\text{UPDATE}(i, \cdot)\) s.t. \(a_i = a_i^{\text{massage}}\)

Until \(\mathcal{B}_{\text{want}} = \emptyset\)

Return Unknown \(\triangleright\) Call Simplex
**RESEEDING SIMPLEX** \((a_{\text{massage}}, \tilde{B})\)

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

\(\mathcal{B}_{\text{want}} \leftarrow \mathcal{N} \cap \tilde{\mathcal{B}}\)

repeat
  if any row conflict then  return Unsat
  if \(l \leq a \leq u\) then  return Sat

select \(i, k\) s.t. \(k \in \mathcal{B}_{\text{want}}, i \notin \tilde{\mathcal{B}}, T_{i,k} \neq 0,\) and \(V_i > 0\)

if no such \(\langle i, k \rangle\) then
  return Unknown  \(\triangleright \tilde{\mathcal{B}}\) is not valid basis
else
  PIVOT\((i, k)\) and UPDATE\((i, \cdot)\) s.t. \(a_i = a_i^{\text{massage}}\)

until \(\mathcal{B}_{\text{want}} = \emptyset\)

return Unknown  \(\triangleright\) Call Simplex
```

More robust with SOI Simplex [KBD13]
MOVE $\langle \text{QF\_LRA}, LP \rangle \rightarrow \langle \text{QF\_LIRA}, MIP \rangle$

- Partition variables $\mathcal{X}$ into $\mathcal{X}_R \cup \mathcal{X}_Z$
MOVE $\langle \text{QF}_L\text{RA}, LP \rangle \rightarrow \langle \text{QF}_L\text{IRA}, MIP \rangle$

- Partition variables $\mathcal{X}$ into $\mathcal{X}_R \cup \mathcal{X}_Z$
- $\mathbb{R}$-relaxation treat all $\mathcal{X}$ as $\mathcal{X}_R$
- $a$ is integer-compatible if $\forall x_i \in \mathcal{X}_Z$, then $a_i \in \mathbb{Z}$
**MOVE** \( \langle QF\_LRA, LP \rangle \rightarrow \langle QF\_LIRA, MIP \rangle \)

- Partition variables \( \mathcal{X} \) into \( \mathcal{X}_R \cup \mathcal{X}_Z \)
- \( \mathbb{R} \)-relaxation treat all \( \mathcal{X} \) as \( \mathcal{X}_R \)
- \( a \) is **integer-compatible** if \( \forall x_i \in \mathcal{X}_Z, \text{then } a_i \in \mathbb{Z} \)
- MIP is new for SMT
**Another Example: Visually**

\[
x + y \geq 1 \\
x - y \geq 0 \\
4x - y \leq 2
\]

\[
\begin{bmatrix}
a_x \\
a_y
\end{bmatrix} = \begin{bmatrix}
\frac{1}{2} \\
\frac{1}{2}
\end{bmatrix}
\]
Branches and Cuts
Refining \( \mathbb{Z} \)-infeasible assignments

- **Branch:**
  \[
  x_i \in \mathcal{X}_\mathbb{Z} \quad \alpha \in \mathbb{R} \\
  x_i \leq \lfloor \alpha \rfloor \lor x_i \geq \lceil \alpha \rceil
  \]

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

*Visually*

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

Cut: $\{ \cdots \} \models_{RZ} x \geq 1$
Branch-and-cut Solvers
Most SMT solvers and many MIP solvers

1. Treat all of \( \mathcal{X} \) as if they were \( \mathcal{X}_R \)

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 \( \sum c_jx_j \geq d \), and goto (1)

6. Branch on some \( x_i \in \mathcal{X}_\mathbb{Z} \) with \( a_i \not\in \mathbb{Z} \)
**Branch-and-cut Solvers**

Most SMT solvers and many MIP solvers

1. Treat all of $\mathcal{X}$ as if they were $\mathcal{X}_R$

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 \(\text{QF}_{\text{LIA}}\) and \(\text{QF}_{\text{LIRA}}\)?

- \(\mathbb{R}\)-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 $\text{QF}._{\text{LIA}}$ and $\text{QF}._{\text{LIRA}}$?

- $\mathbb{R}$-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
  \[ x_i \leq \lfloor \alpha \rfloor \lor x_i \geq \lceil \alpha \rceil \quad \text{if } x_i \in \mathcal{X}_\mathbb{Z} \]
- 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

Satisfiability Modulo Theories

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

Sum Of Infeasibilities Simplex [FMCAD13]

Reseed & Replay [FMCAD14]

Empirical Results

Conclusion
**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

<table>
<thead>
<tr>
<th></th>
<th>SOI</th>
<th>Z3</th>
<th>yices2</th>
<th>mathsat</th>
</tr>
</thead>
<tbody>
<tr>
<td>QFLRA (634)</td>
<td>618</td>
<td>620</td>
<td>619</td>
<td>608</td>
</tr>
<tr>
<td>latendresse (18)</td>
<td>18</td>
<td>8</td>
<td>10</td>
<td>10</td>
</tr>
</tbody>
</table>
SOI Simplex 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 SOI Simplex
SOI Simplex versus Simplex for DPLL(τ)

New family Lasso Ranker

Solver Comparison Plot
SOISimplex + Reseed + Replay Results
## SMT Solver Comparison

### QF_LRA

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

(AR) = Applied either RESEED or REPLAY, $K = 1000$, & SOI+MIP is CVC4 1.4 with options
# SMT Solver Comparison

## QF-LIA ¬-conjunctive

<table>
<thead>
<tr>
<th></th>
<th>SOI+MIP</th>
<th>CVC4</th>
<th>mathsat5</th>
<th>Z3</th>
<th>altergo</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>set</strong></td>
<td># inst.</td>
<td># sel.</td>
<td>solved time (s)</td>
<td>solved time (s)</td>
<td>solved time (s)</td>
</tr>
<tr>
<td>everything</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>QF-LIA</td>
<td>5882</td>
<td>5882</td>
<td>5738</td>
<td>97K</td>
<td>5540</td>
</tr>
<tr>
<td>conjuncts</td>
<td>1303</td>
<td>1303</td>
<td>1249</td>
<td>11K</td>
<td>1068</td>
</tr>
<tr>
<td>(AR) ¬ conjunctive</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>convert</td>
<td>319</td>
<td>282</td>
<td>208</td>
<td>9646</td>
<td>193</td>
</tr>
<tr>
<td>bofill-*</td>
<td>652</td>
<td>460</td>
<td>460</td>
<td>5401</td>
<td>458</td>
</tr>
<tr>
<td>CIRC</td>
<td>51</td>
<td>11</td>
<td>11</td>
<td>0</td>
<td>11</td>
</tr>
<tr>
<td>calypto</td>
<td>37</td>
<td>37</td>
<td>37</td>
<td>3</td>
<td>37</td>
</tr>
<tr>
<td>nec-smt</td>
<td>2780</td>
<td>207</td>
<td>207</td>
<td>17K</td>
<td>207</td>
</tr>
<tr>
<td>wisa</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td><strong>total</strong></td>
<td>-</td>
<td>998</td>
<td>924</td>
<td>32K</td>
<td>907</td>
</tr>
</tbody>
</table>

(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

<table>
<thead>
<tr>
<th></th>
<th>SOI+MIP</th>
<th>CVC4</th>
<th>mathsat5</th>
<th>Z3</th>
<th>altergo</th>
</tr>
</thead>
<tbody>
<tr>
<td>set</td>
<td># inst.</td>
<td># sel.</td>
<td>solved time (s)</td>
<td>solved time (s)</td>
<td>solved time (s)</td>
</tr>
<tr>
<td>everything</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>QF_LIA</td>
<td>5882</td>
<td>5882</td>
<td>5738</td>
<td>97K</td>
<td>5540</td>
</tr>
<tr>
<td>conjuncts</td>
<td>1303</td>
<td>1303</td>
<td>1249</td>
<td>11K</td>
<td>1068</td>
</tr>
</tbody>
</table>

### (AR) Conjunctive

<table>
<thead>
<tr>
<th></th>
<th>SOI+MIP</th>
<th>CVC4</th>
<th>mathsat5</th>
<th>Z3</th>
<th>altergo</th>
</tr>
</thead>
<tbody>
<tr>
<td>dillig</td>
<td>233</td>
<td>189</td>
<td>189</td>
<td>49</td>
<td>157</td>
</tr>
<tr>
<td>miplib2003</td>
<td>16</td>
<td>8</td>
<td>4</td>
<td>307</td>
<td>4</td>
</tr>
<tr>
<td>prime-cone</td>
<td>37</td>
<td>37</td>
<td>37</td>
<td>2</td>
<td>37</td>
</tr>
<tr>
<td>slacks</td>
<td>233</td>
<td>188</td>
<td>166</td>
<td>61</td>
<td>93</td>
</tr>
<tr>
<td>CAV 2009</td>
<td>591</td>
<td>424</td>
<td>424</td>
<td>69</td>
<td>346</td>
</tr>
<tr>
<td>cut lem.</td>
<td>93</td>
<td>74</td>
<td>62</td>
<td>9581</td>
<td>64</td>
</tr>
<tr>
<td>total</td>
<td>-</td>
<td>920</td>
<td>882</td>
<td>10K</td>
<td>701</td>
</tr>
</tbody>
</table>

(AR) = Applied either RESEED or REPLAY, K = 1000, & SOI+MIP is CVC4 1.4 with options
### COMPARISON WITH CONJUNCTIVE SOLVERS

<table>
<thead>
<tr>
<th>set</th>
<th># inst.</th>
<th># sel.</th>
<th>solved (conjuncts)</th>
<th>time (s)</th>
<th>solved (conjuncts)</th>
<th>time (s)</th>
<th>solved (conjuncts)</th>
<th>time (s)</th>
<th>solved (conjuncts)</th>
<th>time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>conjuncts</td>
<td>1303</td>
<td>1303</td>
<td>1249</td>
<td>11130</td>
<td>1018</td>
<td>35330</td>
<td>1255</td>
<td>7164</td>
<td>1173</td>
<td>8895</td>
</tr>
<tr>
<td>(AR) conjunctive</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>dillig</td>
<td>233</td>
<td>189</td>
<td>189</td>
<td>49</td>
<td>166</td>
<td>5840</td>
<td>189</td>
<td>42</td>
<td>189</td>
<td>3</td>
</tr>
<tr>
<td>miplib2003</td>
<td>16</td>
<td>8</td>
<td>4</td>
<td>307</td>
<td>6</td>
<td>146</td>
<td>7</td>
<td>17</td>
<td>6</td>
<td>295</td>
</tr>
<tr>
<td>prime-cone</td>
<td>37</td>
<td>37</td>
<td>37</td>
<td>2</td>
<td>37</td>
<td>4</td>
<td>37</td>
<td>1</td>
<td>37</td>
<td>0</td>
</tr>
<tr>
<td>slacks</td>
<td>233</td>
<td>188</td>
<td>166</td>
<td>61</td>
<td>96</td>
<td>6324</td>
<td>161</td>
<td>2361</td>
<td>101</td>
<td>11</td>
</tr>
<tr>
<td>CAV 2009</td>
<td>591</td>
<td>424</td>
<td>424</td>
<td>69</td>
<td>377</td>
<td>17015</td>
<td>424</td>
<td>105</td>
<td>424</td>
<td>6</td>
</tr>
<tr>
<td>cut_lemmas</td>
<td>93</td>
<td>74</td>
<td>62</td>
<td>9581</td>
<td>15</td>
<td>1887</td>
<td>72</td>
<td>1757</td>
<td>71</td>
<td>760</td>
</tr>
<tr>
<td>total</td>
<td>-</td>
<td>920</td>
<td>882</td>
<td>10069</td>
<td>697</td>
<td>31216</td>
<td>890</td>
<td>4283</td>
<td>828</td>
<td>1075</td>
</tr>
</tbody>
</table>

(AR) = Applied either RESEED or REPLAY, $K = 1000$, & SOI+MIP is CVC4 1.4 with options
cutsat is using [JovanovicM11]
# QF_LIA Reseed and Replay Success Rates

<table>
<thead>
<tr>
<th>Set</th>
<th># Inst.</th>
<th>Solve Int Calls</th>
<th>Reseed Attempts</th>
<th>Reseed Successes</th>
<th>Replay Attempts</th>
<th>Replay Successes</th>
</tr>
</thead>
<tbody>
<tr>
<td>QF_LIA</td>
<td>1806</td>
<td>3873</td>
<td>2559</td>
<td>1058</td>
<td>652</td>
<td>425</td>
</tr>
<tr>
<td>Convert</td>
<td>208</td>
<td>2130</td>
<td>1356</td>
<td>1</td>
<td>178</td>
<td>3</td>
</tr>
<tr>
<td>Bofill-Scheduling</td>
<td>460</td>
<td>254</td>
<td>245</td>
<td>245</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>CIRC</td>
<td>11</td>
<td>85</td>
<td>6</td>
<td>5</td>
<td>79</td>
<td>77</td>
</tr>
<tr>
<td>Calypto</td>
<td>37</td>
<td>375</td>
<td>77</td>
<td>23</td>
<td>293</td>
<td>278</td>
</tr>
<tr>
<td>Wisa</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Dillig</td>
<td>189</td>
<td>228</td>
<td>225</td>
<td>185</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>Miplib2003</td>
<td>4</td>
<td>10</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>Prime-Cone</td>
<td>37</td>
<td>37</td>
<td>19</td>
<td>19</td>
<td>18</td>
<td>18</td>
</tr>
<tr>
<td>Slacks</td>
<td>166</td>
<td>195</td>
<td>168</td>
<td>162</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>CAV_2009</td>
<td>424</td>
<td>469</td>
<td>459</td>
<td>414</td>
<td>8</td>
<td>7</td>
</tr>
<tr>
<td>Cut_Lemmas</td>
<td>62</td>
<td>89</td>
<td>0</td>
<td>0</td>
<td>65</td>
<td>33</td>
</tr>
</tbody>
</table>

Only includes solved instances.
SMT-COMP’14

- CVC4 won $\text{QF}_{\text{LRA}}$
- [CVC4-with-bugfix] solved the most $\text{QF}_{\text{LIA}}$ benchmarks
- Won a number of combination & quantified divisions

Also won Typed First-order Theorems +*/- at CASCJ7
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 little where little is scaled by the size of the numbers
  - Numerically stability of floating points
- Gap is too large for $\text{QF} \_{\text{LIA}}$ 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**

- **SOI Simplex**
  - 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


- “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