Leveraging Linear and Mixed Integer Programming for SMT

Tim King[1] Clark Barrett[1] Cesare Tinelli[2]

[1]New York University

[2]The University of Iowa

October 23, 2014

Approach

- Floating point LP/MIP solver within SMT to:
  1. Reseed the Simplex solver
  2. Replay an MIP proof

APPROACH

- Floating point LP/MIP solver within SMT to:
  1. Reseed the Simplex solver
  2. Replay an MIP proof
- Philosophy
  - Solve hard/unsolved problems
  - Augment SMT solver
  - Minimize changes in search by external solver

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,

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

evaluate to true?

Decision Procedure for QF_LRA

Quantifier Free Linear Real Arithmetic

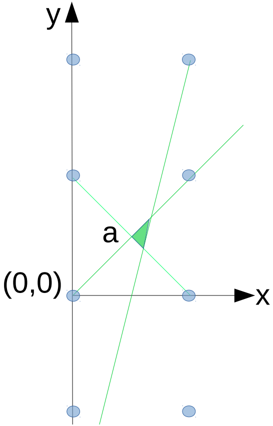
Is there a satisfying assignment, $a : \mathcal{X} \to \mathbb{R}$, that makes,

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

evaluate to true?

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

VISUALLY



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

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

PREPROCESSING

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

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

  and s_i appears in exactly 1 equality.

- Collect into: $T\mathcal{X} = 0$ and $l \leq \mathcal{X} \leq u$

Basic, Nonbasic, & Tableau

- Every row in T is solved for a variable x_i

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

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

Updating Nonbasic Variables

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

procedure UPDATE(j, δ)
  $a_j \leftarrow a_j + \delta$
  for all basic x_i **do**
    $a_i \leftarrow a_i + T_{i,j} \cdot \delta$

Updating Nonbasic Variables

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

procedure UPDATE(j, δ)
  $a_j \leftarrow a_j + \delta$
  for all basic x_i **do**
    $a_i \leftarrow a_i + T_{i,j} \cdot \delta$

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.

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

Tableau Example

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

Tableau Example

$T\mathcal{X} = 0$ is equivalent to

$$\begin{array}{rcrcr} s_1 & = & x & + & y \\ s_2 & = & x & - & y \\ s_3 & = & 4x & + & y \end{array}$$

$$s_1 \geq 1 \wedge s_2 \geq 0 \wedge s_3 \leq 2$$

$$\mathcal{B} = \{s_1, s_2, s_3\}, \mathcal{N} = \{x, y\}$$

SIMPLEX FOR DPLL(T)[DDM06]

while $\neg(l \leq a \leq u)$ **do**

  for all $i \in \mathcal{B}$, row i is $x_i = \sum T_{i,f} x_j$
  if $\exists i \in \mathcal{B}$ s.t. $a_i > u_i$, and $\sum T_{i,j} x_j$ is minimized **then**
    return a row conflict from row i
  else
    select some basic x_i s.t. $a_i > u_i$
    select x_j from $\sum T_{i,j} \cdot x_j$
    Update the assignment of x_j s.t. $a_i \leftarrow u_i$
    PIVOT(i, j) $\triangleright\ O(|T|)$

Ignoring $a_i < l_i$ cases

Row Conflicts

- Suppose $\forall T_{i,j} > 0.\ a_j = l_j$ and $\forall T_{i,j} < 0.\ a_j = u_j$.
- Then $\sum T_{i,j}\, x_j \geq \sum T_{i,j}\, a_j$ (or minimized)

Row Conflicts

- Suppose $\forall T_{i,j} > 0.\ a_j = l_j$ and $\forall T_{i,j} < 0.\ a_j = u_j$.
- Then $x_i = \sum T_{i,j}\, x_j \geq \sum T_{i,j}\, a_j = a_i$ (or minimized)

Row Conflicts

- Suppose $\forall T_{i,j} > 0.\ a_j = l_j$ and $\forall T_{i,j} < 0.\ a_j = u_j$.
- Then $x_i = \sum T_{i,j}\, x_j \geq \sum T_{i,j}\, a_j = a_i$ (or minimized)
- $a_i > u_i \geq x_i \geq a_i \models$ **false**

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

OBSERVATIONS

- Simplex searches for a's that are against bounds
- Pivoting is expensive
- Most checks need few pivots [KBD13]

SUM-OF-INFEASIBILITIES SIMPLEX [KBD13]

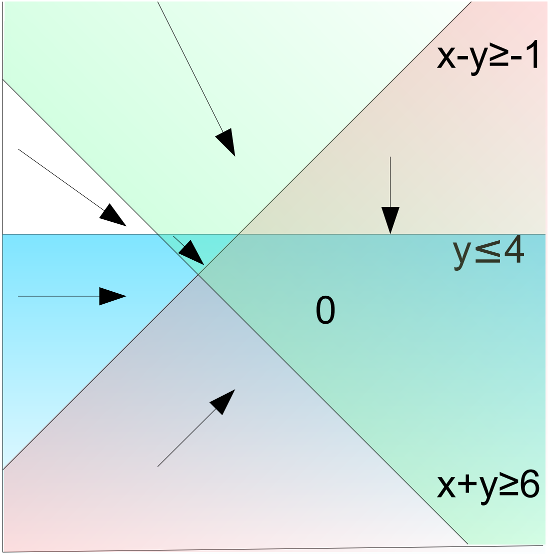


Table of Contents

LEVERAGING LP

- SOISimplex added optimization to Simplex for DPLL(T)
- Linear Programming solvers perform both
  - feasibility checking and
  - optimization

Leveraging LP

- SOISimplex added optimization to Simplex for DPLL(T)
- Linear Programming solvers perform both
  - feasibility checking and
  - optimization
- Decades of research: *fast by SMT standards*

Leveraging LP

- SOISimplex added optimization to Simplex for DPLL(T)
- Linear Programming solvers perform both
  - feasibility checking and
  - optimization
- Decades of research: *fast by SMT standards*
- Tend to use floating point (FP)
- Both Sat/Unsat answers are *unsound*

CAN SMT LEVERAGE LP?

- **Trusting** LP solver [YM06]
- Check each $\mathcal{T}$-conflict used [FNORC08]
- **FORCEDPIVOT procedure** [CBdOM12, Mon09]

CAN SMT LEVERAGE LP?

- **Trusting** LP solver [YM06]
- Check each $\mathcal{T}$-conflict used [FNORC08]
- **FORCEDPIVOT procedure** [CBdOM12, Mon09]
- All use LP solver as main `QF_LRA` solver

Our 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

When $\mathbb{R}$-relaxation is hard

1. Construct a FP problem from exact

$$T\mathcal{X} = 0,\ l \le \mathcal{X} \le u \quad \Longrightarrow \quad \widetilde{T}\mathcal{X} = 0,\ \widetilde{l} \le \mathcal{X} \le \widetilde{u}$$

2. Call *untrusted* LP Simplex solver on $\widetilde{T}, \widetilde{l}, \widetilde{u}$
3. Get back FP $\widetilde{a}$ and $\widetilde{\mathcal{B}}$
4. Convert $(\widetilde{a} : \mathcal{X} \to \mathbb{F})$ into $(a^{massage} : \mathcal{X} \to \mathbb{Q})$
5. $\textsc{Reseed}(a^{massage}, \widetilde{\mathcal{B}})$ to get a new a and T
6. Call SMT's *trusted* $\mathbb{Q}$ Simplex solver

CONCERNS WHEN IMPORTING $\tilde{a}$

$$y = -\frac{2}{3}x + \frac{1}{3}s \qquad s \geq 1 \qquad \begin{bmatrix} a_x \\ a_y \\ a_s \end{bmatrix} = \begin{bmatrix} 0 \\ \frac{1}{3} \\ 1 \end{bmatrix}$$

Suppose $a_y = \frac{1}{3} - \epsilon$. Then $a_s < 1$.

CONCERNS WHEN IMPORTING $\widetilde{a}$

$$y = -\frac{2}{3}x + \frac{1}{3}s \qquad s \geq 1 \qquad \begin{bmatrix} a_x \\ a_y \\ a_s \end{bmatrix} = \begin{bmatrix} 0 \\ \frac{1}{3} \\ 1 \end{bmatrix}$$

Suppose $a_y = \frac{1}{3} - \epsilon$. Then $a_s < 1$.

- Fix it with Simplex?
- Flipping coins on tightly satisfied inequalities
- Simplex generates tight solutions

Massaging Assignments

Floats to Rationals

$r \leftarrow \textsc{DioApprox}(\widetilde{a}_i, D)$
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** $r \leftarrow l_i$

$a_i^{massage} \leftarrow r$

Magic $D = 2^{28}$

Massaging Assignments

Floats to Rationals

$r \leftarrow \text{DIOAPPROX}(\tilde{a}_i, D)$
if $|r - a_i|$ See paper for details

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** $r \leftarrow l_i$

$a_i^{massage} \leftarrow r$

Magic $D = 2^{28}$

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

for all $j \in \mathcal{N}$ **do** UPDATE x_j s.t. $a_j \leftarrow a_j^{massage}$

repeat

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

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

  select i, k s.t. $k \in \widetilde{\mathcal{B}}$, $i \notin \widetilde{\mathcal{B}}$, $T_{i,k} \neq 0$, and $a_i > u_i$ $(\ldots)$

  if found x_i and x_k **then**

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

  else

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

until $\mathcal{N} \cap \widetilde{\mathcal{B}} = \varnothing$

return Unknown ▷ Call SMT's simplex solver

Reseeding Simplex ($a^{massage}$, $\widetilde{\mathcal{B}}$): Abstract

Pull in $a^{massage}$ on $\mathcal{N}$

repeat

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

  Select leaving x_i from $\neg\widetilde{\mathcal{B}}$

  Select entering x_j from $\mathcal{N} \cap \widetilde{\mathcal{B}}$

until $\mathcal{N} \cap \widetilde{\mathcal{B}} = \varnothing$ or fail

  Call SMT's simplex solver

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

Pull in $a^{massage}$ on $\mathcal{N}$
repeat
    One Simplex for DPLL($\mathcal{T}$) round
    Select leaving x_i from $\neg\widetilde{\mathcal{B}}$
    Select entering x_j from $\mathcal{N} \cap \widetilde{\mathcal{B}}$
until $\mathcal{N} \cap \widetilde{\mathcal{B}} = \varnothing$ or fail
    Call SMT's simplex solver

Reseeding Simplex ($a^{massage}, \widetilde{\mathcal{B}}$): Abstract

Pull in $a^{massage}$ on $\mathcal{N}$
repeat
    One Simplex for DPLL($\mathcal{T}$) round
    Select leaving x_i from $\neg\widetilde{\mathcal{B}}$
    Select entering x_j from $\mathcal{N} \cap \widetilde{\mathcal{B}}$
until $\mathcal{N} \cap \widetilde{\mathcal{B}} = \varnothing$ or fail
    Call SMT's simplex solver

Table of Contents

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

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

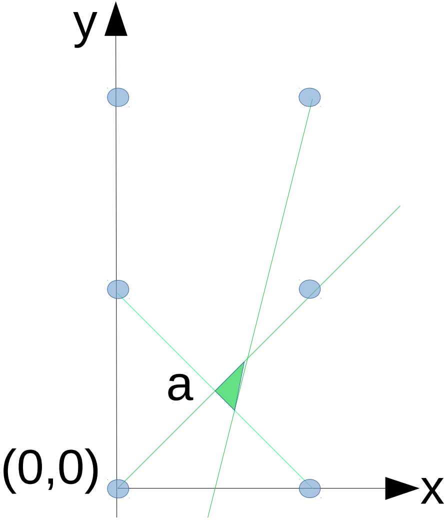
MOVE $\langle \texttt{QF\_LRA} + LP \rangle \rightarrow \langle \texttt{QF\_LIRA} + MIP \rangle$

- 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 **$\mathbb{Z}$-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 $\mathbb{Z}$-**compatible** if $\forall x_i \in \mathcal{X}_{\mathbb{Z}}$, then $a_i \in \mathbb{Z}$
- MIP is new for DPLL($\mathcal{T}$)

RETURNING TO THE EXAMPLE



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

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

$\mathbb{R}$-feasible
not
$\mathbb{Z}$-compatible

BRANCHES AND CUTS

REFINING $\mathbb{Z}$-INFEASIBLE ASSIGNMENTS

- Branch:

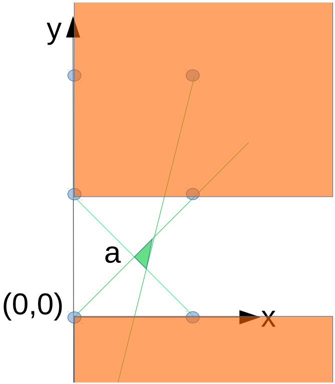
$$\frac{x_i \in \mathcal{X}_{\mathbb{Z}} \qquad \alpha \in \mathbb{R}}{x_i \leq \lfloor \alpha \rfloor \vee x_i \geq \lceil \alpha \rceil}$$

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

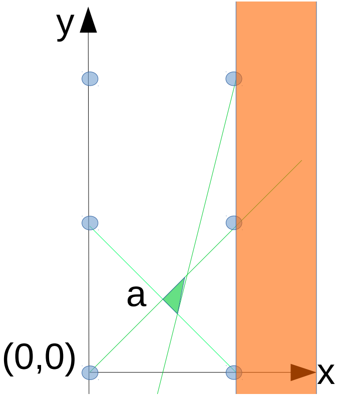
Branches and Cuts

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 this $\mathbb{R}$-*relaxation*
3. If $\mathbb{R}$-infeasible, return $\mathbb{R}$-conflict[s]
4. If $\mathbb{R}$-relaxation is $(\textbf{Sat}\ a)$ and a is $\mathbb{Z}$-compatible, return a
5. Try to derive the cut $\sum c_j x_j \geq d$
6. If successful, add the cut and goto (1)
7. 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 this $\mathbb{R}$-*relaxation*
3. If $\mathbb{R}$-infeasible, return $\mathbb{R}$-conflict[s]
4. If $\mathbb{R}$-relaxation is $(\textbf{Sat}\ a)$ and a is $\mathbb{Z}$-compatible, return a
5. Try to derive the cut $\sum c_j x_j \geq d$
6. If successful, add the cut and goto (1)
7. Branch on some $x_i \in \mathcal{X}_{\mathbb{Z}}$ with $a_i \notin \mathbb{Z}$

Heuristically limit cuts

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 this $\mathbb{R}$-*relaxation*
3. If $\mathbb{R}$-infeasible, return $\mathbb{R}$-conflict[s]
4. If $\mathbb{R}$-relaxation is $(\textbf{Sat}\ a)$ and a is $\mathbb{Z}$-compatible, return a
5. Try to derive the cut $\sum c_j x_j \geq d$
6. If successful, add the cut and goto (1)
7. Branch on some $x_i \in \mathcal{X}_{\mathbb{Z}}$ with $a_i \notin \mathbb{Z}$

Heuristically limit cuts Only at leaves in DPLL($\mathcal{T}$)

Possible answers from MIP?

1. $\mathbb{R}$-infeasible
2. $\mathbb{R}$-feasible and $\mathbb{Z}$-feasible
3. $\mathbb{R}$-feasible and $\mathbb{Z}$-infeasible
4. Failure Cases

Possible answers from MIP?

1. $\mathbb{R}$-infeasible
2. $\mathbb{R}$-feasible and $\mathbb{Z}$-feasible
3. $\mathbb{R}$-feasible and $\mathbb{Z}$-infeasible
4. Failure Cases

Just Reseed like $\mathbb{R}$-feasible
If a is $\mathbb{Z}$-compatible $\implies$ done!

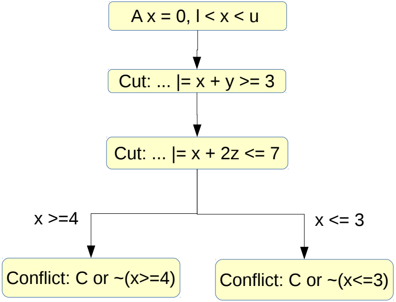
Possible answers from MIP?

1. $\mathbb{R}$-infeasible
2. $\mathbb{R}$-feasible and $\mathbb{Z}$-feasible
3. $\mathbb{R}$-feasible and $\mathbb{Z}$-infeasible
4. Failure Cases

Can we leverage MIP's reasoning?

INFEASIBLE BRANCH-AND-CUT EXECUTIONS

PROOF TREES



- Leaves are $\mathbb{R}$-infeasible
- Internal nodes are branches

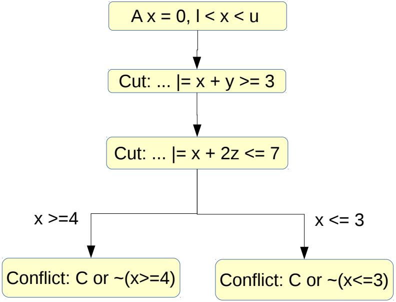
$$x_i \leq \lfloor \alpha \rfloor \vee x_i \geq \lceil \alpha \rceil \quad \text{if } x_i \in \mathcal{X}_{\mathbb{Z}}$$

- Nodes have cuts

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

Infeasible Branch-and-Cut Executions

Proof Trees



- Leaves are $\mathbb{R}$-infeasible
- 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

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

Resolution to remove branches

Replaying the MIP Execution

- 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

- 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

Replaying the MIP Execution

- 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
- Success is a conflict

Replaying the MIP Execution

- 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
- Success is a conflict
- Any failure can be safely dropped

Cutting Planes

- Instantiate a cutting plane procedure from a hint
- Derivation must tightly match to get the "same" cut
- White-box knowledge and detailed hints
- Support for Gomory (easy) and MIR (hard) cuts

Table of Contents

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
total	-	41	34	1534	25	3041	27	2330	21	5684	22	2436

(AR) = Applied either RESEED or REPLAY, **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 [BCC+12]

SMT Solver Comparison

QF_LIA conjunctive

set	# inst.	# sel.	SOI+MIP solved	SOI+MIP time (s)	CVC4 solved	CVC4 time (s)	mathsat5 solved	mathsat5 time (s)	Z3 solved	Z3 time (s)	altergo solved	altergo 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 [JdM11]

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

Table of Contents

FUTURE WORK

- Optimization Modulo Theories
- Logging and replaying FP Farkas's lemma [NS04]
- k-precision FP Simplex solver for SMT [CKSW13]

Replay & Reseed Summary

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

REPLAY & RESEED SUMMARY

- Integrated a floating point LP/MIP solver (GLPK) (Backup. Not the main engine!)
- Reseeding Simplex
  - Helps find models and $\mathbb{R}$-relaxation conflicts
  - 1 week to implement [*]

Replay & Reseed Summary

- Integrated a floating point LP/MIP solver (GLPK)
  (Backup. Not the main engine!)
- Reseeding Simplex
  - Helps find models and $\mathbb{R}$-relaxation conflicts
  - 1 week to implement [*]
- 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 engine!)
- Reseeding Simplex
  - Helps find models and $\mathbb{R}$-relaxation conflicts
  - 1 week to implement [*]
- Replaying MIP conflicts (significantly more effort)
  MIP must be white-box and must log proofs!
- Overall helpful, but there are limitations

REPLAY & RESEED SUMMARY

- Integrated a floating point LP/MIP solver (GLPK)
  (Backup. Not the main engine!)
- Reseeding Simplex
  - Helps find models and $\mathbb{R}$-relaxation conflicts
  - 1 week to implement [*]
- Replaying MIP conflicts (significantly more effort)
  MIP must be white-box and must log proofs!
- Overall helpful, but there are limitations

Thank you for your attention!

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 `QF_LIA` bit-extracts for $\sim m + n > 40$

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

- Decreasing the maximum gap leads $\implies$ cycling
- Need bigger floating point numbers or more pre-processing

References I

François Bobot, Sylvain Conchon, Évelyne Contejean, Mohamed Iguernelala, Assia Mahboubi, Alain Mebsout, and Guillaume Melquiond, *A Simplex-based extension of Fourier-Motzkin for solving linear integer arithmetic*, IJCAR 2012: Proceedings of the 6th International Joint Conference on Automated Reasoning (Manchester, UK) (Bernhard Gramlich, Dale Miller, and Ulrike Sattler, eds.), Lecture Notes in Computer Science, vol. 7364, Springer, June 2012, pp. 67–81.

Diego Caminha Barbosa de Oliveira and David Monniaux, *Experiments on the feasibility of using a floating-point simplex in an SMT solver*, Workshop on Practical Aspects of Automated Reasoning (PAAR), CEUR Workshop Proceedings, 2012.

References II

William Cook, Thorsten Koch, Daniel E. Steffy, and Kati Wolter, *A hybrid branch-and-bound approach for exact rational mixed-integer programming*, Math. Program. Comput. **5** (2013), no. 3, 305–344.

Bruno Dutertre and Leonardo de Moura, *Integrating Simplex with DPLL(T)*, Tech. Report SRI-CSL-06-01, Computer Science Laboratory, SRI International, May 2006.

Germain Faure, Robert Nieuwenhuis, Albert Oliveras, and Enric Rodríguez-Carbonell, *Sat modulo the theory of linear arithmetic: Exact, inexact and commercial solvers*, SAT, 2008, pp. 77–90.

References III

- Dejan Jovanović and Leonardo Mendonça de Moura, *Cutting to the chase solving linear integer arithmetic*, CADE, 2011, pp. 338–353.

- Timothy King, Clark Barrett, and Bruno Dutertre, *Simplex with sum of infeasibilities for SMT*, Proceedings of the 13^{th} International Conference on Formal Methods In Computer-Aided Design (FMCAD '13), Lecture Notes in Computer Science, November 2013, pp. 189–196.

- David Monniaux, *On using floating-point computations to help an exact linear arithmetic decision procedure*, Computer-aided verification (CAV), Lecture Notes in Computer Science, no. 5643, Springer-Verlag, 2009, pp. 570–583.

References IV

Arnold Neumaier and Oleg Shcherbina, *Safe bounds in linear and mixed-integer linear programming*, Mathematical Programming **99** (2004), no. 2, 283–296.

Yinlei Yu and Sharad Malik, *Lemma learning in smt on linear constraints*, Theory and Applications of Satisfiability Testing - SAT 2006 (Armin Biere and CarlaP. Gomes, eds.), Lecture Notes in Computer Science, vol. 4121, Springer Berlin Heidelberg, 2006, pp. 142–155.

Appendix

Resolution Phase

The proof reconstruction phase uses the following heuristics:

- All up-branch conflicts are resolved with all down-branch conflicts
  (DP-style)
- Performed eager subsumption checking
  Pays for itself by keeping the set of conflicts small
- Non-chronological backtracks when possible
  (One branch has a conflict not involving its branch literal)