On Designing and Implementing Satisfiability Modulo Theory (SMT) Solvers

Summer School 2009 , Nancy
Verification Technology, Systems and Applications

Leonardo de Moura
Microsoft Research

Linear Arithmetic

- Many approaches
  - Graph-based for difference logic: $a - b \leq 3$
  - Fourier-Motzkin elimination:

$$t_1 \leq ax,\ \ bx \leq t_2 \ \ \Rightarrow \ \ bt_1 \leq at_2$$

  - Standard Simplex
  - General Form Simplex

Difference Logic: $a - b \leq 5$

Very useful in practice!

Most arithmetical constraints in software verification/analysis are in this fragment.

$$x := x + 1$$

$$\Downarrow$$

$$x_1 = x_0 + 1$$

$$\Downarrow$$

$$x_1 - x_0 \leq 1,\ x_0 - x_1 \leq -1$$

Microsoft Research

Job shop scheduling

$d_{i,j}$	Machine 1	Machine 2
Job 1	2	1
Job 2	3	1
Job 3	2	3

$max = 8$

Solution

$t_{1,1} = 5,\ t_{1,2} = 7,\ t_{2,1} = 2,$
$t_{2,2} = 6,\ t_{3,1} = 0,\ t_{3,2} = 3$

Encoding

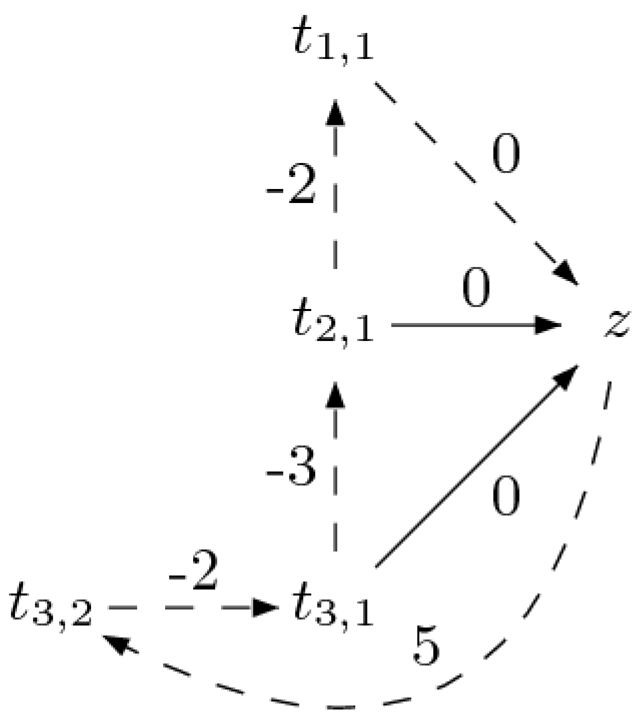
$$
\begin{aligned}
&(t_{1,1} \geq 0) \wedge (t_{1,2} \geq t_{1,1} + 2) \wedge (t_{1,2} + 1 \leq 8) \wedge \\
&(t_{2,1} \geq 0) \wedge (t_{2,2} \geq t_{2,1} + 3) \wedge (t_{2,2} + 1 \leq 8) \wedge \\
&(t_{3,1} \geq 0) \wedge (t_{3,2} \geq t_{3,1} + 2) \wedge (t_{3,2} + 3 \leq 8) \wedge \\
&((t_{1,1} \geq t_{2,1} + 3) \vee (t_{2,1} \geq t_{1,1} + 2)) \wedge \\
&((t_{1,1} \geq t_{3,1} + 2) \vee (t_{3,1} \geq t_{1,1} + 2)) \wedge \\
&((t_{2,1} \geq t_{3,1} + 2) \vee (t_{3,1} \geq t_{2,1} + 3)) \wedge \\
&((t_{1,2} \geq t_{2,2} + 1) \vee (t_{2,2} \geq t_{1,2} + 1)) \wedge \\
&((t_{1,2} \geq t_{3,2} + 3) \vee (t_{3,2} \geq t_{1,2} + 1)) \wedge \\
&((t_{2,2} \geq t_{3,2} + 3) \vee (t_{3,2} \geq t_{2,2} + 1))
\end{aligned}
$$

Difference Logic

Chasing negative cycles!

Algorithms based on Bellman-Ford (O(mn)).

$$
\begin{array}{rcrcr}
z & - & t_{1,1} & \leq & 0 \\
z & - & t_{2,1} & \leq & 0 \\
z & - & t_{3,1} & \leq & 0 \\
t_{3,2} & - & z & \leq & 5 \\
t_{3,1} & - & t_{3,2} & \leq & -2 \\
t_{2,1} & - & t_{3,1} & \leq & -3 \\
t_{1,1} & - & t_{2,1} & \leq & -2
\end{array}
$$



Standard Simplex

Many solvers (e.g., ICS, Simplify) are based on the Standard Simplex.

$$
\begin{aligned}
a - d + 2e &= 3 \\
b - d &= 1 \\
c + d - e &= -1 \\
a, b, c, d, e &\geq 0
\end{aligned}
$$

Microsoft Research

Standard Simplex

Many solvers (e.g., ICS, Simplify) are based on the Standard Simplex.

$$
\begin{aligned}
a - d + 2e &= 3 \\
b - d &= 1 \\
c + d - e &= -1 \\
a, b, c, d, e &\geq 0
\end{aligned}
$$

$$
\begin{pmatrix} 1 & 0 & 0 & -1 & 2 \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 1 & 1 & -1 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \\ e \end{pmatrix} = \begin{pmatrix} 3 \\ 1 \\ -1 \end{pmatrix}
$$

$Ax = b$ and $x \geq 0.$

Microsoft Research

Standard Simplex

Many solvers (e.g., ICS, Simplify) are based on the Standard Simplex.

$$\begin{aligned} a - d + 2e &= 3 \\ b - d &= 1 \\ c + d - e &= -1 \\ a, b, c, d, e &\geq 0 \end{aligned}$$

We say a,b,c are the basic (or dependent) variables

$$\begin{pmatrix} 1 & 0 & 0 & -1 & 2 \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 1 & 1 & -1 \end{pmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \end{bmatrix} = \begin{bmatrix} 3 \\ 1 \\ -1 \end{bmatrix}$$

$Ax = b$ and $x \geq 0.$

Microsoft Research

Standard Simplex

Many solvers (e.g., ICS, Simplify) are based on the Standard Simplex.

$$
\begin{aligned}
a - d + 2e &= 3 \\
b - d &= 1 \\
c + d - e &= -1 \\
a, b, c, d, e &\geq 0
\end{aligned}
$$

We say a,b,c are the basic (or dependent) variables

We say d,e are the non-basic (or non-dependent) variables.

$$
\begin{pmatrix} 1 & 0 & 0 & -1 & 2 \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 1 & 1 & -1 \end{pmatrix}
\begin{pmatrix} a \\ b \\ c \\ d \\ e \end{pmatrix}
=
\begin{pmatrix} 3 \\ 1 \\ -1 \end{pmatrix}
$$

$Ax = b$ and $x \geq 0.$

Microsoft Research

Standard Simplex

- Incrementality: add/remove equations
- Slow backtracking
- No theory propagation

Microsoft Research

Fast Linear Arithmetic

- Simplex General Form
- Algorithm based on the dual simplex
- Non redundant proofs
- Efficient backtracking
- Efficient theory propagation
- Support for string inequalities: t > 0
- Preprocessing step
- Integer problems:

  Gomory cuts, Branch & Bound, GCD test

Microsoft Research

General Form

General Form: $Ax = 0$ and $l_j \leq x_j \leq u_j$

Example:

$$x \geq 0, (x + y \leq 2 \vee x + 2y \geq 6), (x + y = 2 \vee x + 2y > 4)$$

$$\rightsquigarrow$$

$$s_1 \equiv x + y, s_2 \equiv x + 2y,$$

$$x \geq 0, (s_1 \leq 2 \vee s_2 \geq 6), (s_1 = 2 \vee s_2 > 4)$$

Only bounds (e.g., $s_1 \leq 2$) are asserted during the search.

Unconstrained variables can be eliminated before the beginning of the search.

From Definitions to a Tableau

$$s_1 \equiv x + y, \quad s_2 \equiv x + 2y$$

Microsoft Research

From Definitions to a Tableau

$$s_1 \equiv x + y, \quad s_2 \equiv x + 2y$$

$$s_1 = x + y,$$
$$s_2 = x + 2y$$

Microsoft Research

From Definitions to a Tableau

$$s_1 \equiv x + y, \quad s_2 \equiv x + 2y$$

$$s_1 = x + y,$$

$$s_2 = x + 2y$$

$$s_1 - x - y \quad = 0$$

$$s_2 - x - 2y = 0$$

Microsoft Research

From Definitions to a Tableau

$$s_1 \equiv x + y, \quad s_2 \equiv x + 2y$$

$$s_1 = x + y,$$
$$s_2 = x + 2y$$

$$s_1 - x - y \;\; = 0$$
$$s_2 - x - 2y = 0$$

s_1, s_2 are basic (dependent)

x,y are non-basic

Pivoting

A way to swap a basic with a non-basic variable!

It is just equational reasoning.

Key invariant: a basic variable occurs in only one equation.

Example: swap s_1 and y

$$s_1 - x - y = 0$$
$$s_2 - x - 2y = 0$$

Microsoft Research

Pivoting

A way to swap a basic with a non-basic variable!

It is just equational reasoning.

Key invariant: a basic variable occurs in only one equation.

Example: swap s_1 and y

$$s_1 - x - y = 0$$
$$s_2 - x - 2y = 0$$

$$\Downarrow$$

$$-s_1 + x + y = 0$$
$$s_2 - x - 2y = 0$$

Pivoting

A way to swap a basic with a non-basic variable!

It is just equational reasoning.

Key invariant: a basic variable occurs in only one equation.

Example: swap s_1 and y

$$s_1 - x - y \quad = 0$$
$$s_2 - x - 2y = 0$$

$$\Downarrow$$

$$-s_1 + x + y \quad = 0$$
$$s_2 - x - 2y = 0$$

$$\Downarrow$$

$$-s_1 + x + y \quad = 0$$
$$s_2 - 2s_1 + x = 0$$

Microsoft Research

Pivoting

A way to swap a basic with a non-basic variable!

It is just equational reasoning.

Key invariant: a basic variable occurs in only one equation.

Example: swap s_1 and y

$$s_1 - x - y = 0$$
$$s_2 - x - 2y = 0$$

It is just substituting equals by equals.

$$-s_1 + x + y = 0$$
$$s_2 - x - 2y = 0$$

$$-s_1 + x + y = 0$$
$$s_2 - 2s_1 + x = 0$$

Microsoft Research

Pivoting

Definition:
An assignment (model) is a mapping from variables to values

A way to swap a basic with a non-basic variable!

It is just equational reasoning.

Key invariant: a basic variable occurs in only one equation.

Example: swap s_1 and y

$$s_1 - x - y = 0$$
$$s_2 - x - 2y = 0$$

$$\Downarrow$$

$$-s_1 + x + y = 0$$
$$s_2 - x - 2y = 0$$

$$\Downarrow$$

$$-s_1 + x + y = 0$$
$$s_2 - 2s_1 + x = 0$$

It is just substituting equals by equals.

Key Property:
If an assignment satisfies the equations before a pivoting step, then it will also satisfy them after!

Pivoting

Definition:
An assignment (model) is a mapping from variables to values

A way to swap a basic with a non-basic variable!

It is just equational reasoning.

Key invariant: a basic variable occurs in only one equation.

Example: swap s_2 and y

$$s_1 - x - y = 0$$
$$s_2 - x - 2y = 0$$

$$-s_1 + x + y = 0$$
$$s_2 - x - 2y = 0$$

$$-s_1 + x + y = 0$$
$$s_2 - 2s_1 + x = 0$$

It is just substituting equals by equals.

Example:
$M(x) = 1$
$M(y) = 1$
$M(s_1) = 2$
$M(s_2) = 3$

Key Property:
If an assignment satisfies the equations before a pivoting step, then it will also satisfy them after!

Equations + Bounds + Assignment

An assignment (model) is a mapping from variables to values.

We maintain an assignment that satisfies all equations and bounds.

The assignment of non dependent variables implies the assignment of dependent variables.

Equations + Bounds can be used to derive new bounds.

Example: $x = y - z,\ y \leq 2,\ z \geq 3 \rightsquigarrow x \leq -1.$

The new bound may be inconsistent with the already known bounds.

Example: $x \leq -1,\ x \geq 0.$

"Repairing Models"

If the assignment of a non-basic variable does not satisfy a bound, then fix it and propagate the change to all dependent variables.

$a = c - d$

$b = c + d$

$M(a) = 0$

$M(b) = 0$

$M(c) = 0$

$M(d) = 0$

$1 \leq c$

$a = c - d$

$b = c + d$

$M(a) = 1$

$M(b) = 1$

$M(c) = 1$

$M(d) = 0$

$1 \leq c$

Microsoft Research

"Repairing Models"

If the assignment of a non-basic variable does not satisfy a bound, then fix it and propagate the change to all dependent variables. Of course, we may introduce new "problems".

$a = c - d$
$b = c + d$
$M(a) = 0$
$M(b) = 0$
$M(c) = 0$
$M(d) = 0$
$1 \leq c$
$a \leq 0$

$a = c - d$
$b = c + d$
$M(a) = 1$
$M(b) = 1$
$M(c) = 1$
$M(d) = 0$
$1 \leq c$
$\mathbf{a \leq 0}$

“Repairing Models”

If the assignment of a basic variable does not satisfy a bound, then pivot it, fix it, and propagate the change to its new dependent variables.

$$
\begin{aligned}
a &= c - d \\
b &= c + d \\
M(a) &= 0 \\
M(b) &= 0 \\
M(c) &= 0 \\
M(d) &= 0 \\
1 &\leq a
\end{aligned}
$$

$$
\begin{aligned}
c &= a + d \\
b &= a + 2d \\
M(a) &= 0 \\
M(b) &= 0 \\
M(c) &= 0 \\
M(d) &= 0 \\
1 &\leq a
\end{aligned}
$$

$$
\begin{aligned}
c &= a + d \\
b &= a + 2d \\
M(a) &= 1 \\
M(b) &= 1 \\
M(c) &= 1 \\
M(d) &= 0 \\
1 &\leq a
\end{aligned}
$$

"Repairing Models"

Sometimes, a model cannot be repaired. It is pointless to pivot.

$a = b - c$

$a \leq 0,\ 1 \leq b,\ c \leq 0$

$M(a) = 1$

$M(b) = 1$

$M(c) = 0$

The value of M(a) is too big. We can reduce it by:

- reducing M(b)

  not possible b is at lower bound

- increasing M(c)

  not possible c is at upper bound

Microsoft Research

"Repairing Models"

Extracting proof from failed repair attempts is easy.

$s_1 \equiv a + d,\ s_2 \equiv c + d$

$a = s_1 - s_2 + c$

$a \leq 0,\ 1 \leq s_1,\ s_2 \leq 0,\ 0 \leq c$

$M(a) = 1$

$M(s_1) = 1$

$M(s_2) = 0$

$M(c) = 0$

Microsoft Research

“Repairing Models”

Extracting proof from failed repair attempts is easy.

$s_1 \equiv a + d,\ s_2 \equiv c + d$

$a = s_1 - s_2 + c$

$a \leq 0,\ 1 \leq s_1,\ s_2 \leq 0,\ 0 \leq c$

$M(a) = 1$

$M(s_1) = 1$

$M(s_2) = 0$

$M(c) = 0$

$\{\ a \leq 0,\ 1 \leq s_1,\ s_2 \leq 0,\ 0 \leq c\ \}$ is inconsistent

Microsoft Research

"Repairing Models"

Extracting proof from failed repair attempts is easy.

$s_1 \equiv a + d,\ s_2 \equiv c + d$

$a = s_1 - s_2 + c$

$a \leq 0,\ 1 \leq s_1,\ s_2 \leq 0,\ 0 \leq c$

$M(a) = 1$

$M(s_1) = 1$

$M(s_2) = 0$

$M(c) = 0$

$\{\ a \leq 0,\ 1 \leq s_1,\ s_2 \leq 0,\ 0 \leq c\ \}$ is inconsistent

$\{\ a \leq 0,\ 1 \leq a + d,\ c + d \leq 0,\ 0 \leq c\ \}$ is inconsistent

Strict Inequalities

The method described only handles non-strict inequalities (e.g., $x \leq 2$).

For integer problems, strict inequalities can be converted into non-strict inequalities. $x < 1 \rightsquigarrow x \leq 0$.

For rational/real problems, strict inequalities can be converted into non-strict inequalities using a small δ. $x < 1 \rightsquigarrow x \leq 1 - \delta$.

We do not compute a δ, we treat it symbolically.

δ is an infinitesimal parameter: $(c, k) = c + k\delta$

Example

- Initial state

$$s \geq 1, x \geq 0$$
$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 0$	$s = x + y$	
$M(y) = 0$	$u = x + 2y$	
$M(s) = 0$	$v = x - y$	
$M(u) = 0$		
$M(v) = 0$		

Example

- Asserting $s \geq 1$

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model

$$\begin{aligned} M(x) &= 0 \\ M(y) &= 0 \\ M(s) &= 0 \\ M(u) &= 0 \\ M(v) &= 0 \end{aligned}$$

Equations

$$\begin{aligned} s &= x + y \\ u &= x + 2y \\ v &= x - y \end{aligned}$$

Bounds

Example

- Asserting $s \geq 1$ assignment does not satisfy new bound.

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 0$	$s = x + y$	$s \geq 1$
$M(y) = 0$	$u = x + 2y$	
$M(s) = 0$	$v = x - y$	
$M(u) = 0$		
$M(v) = 0$		

Example

- Asserting $s \geq 1$ pivot s and x (s is a dependent variable).

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 0$	$s = x + y$	$s \geq 1$
$M(y) = 0$	$u = x + 2y$	
$M(s) = 0$	$v = x - y$	
$M(u) = 0$		
$M(v) = 0$		

Example

- Asserting $s \geq 1$ pivot s and x (s is a dependent variable).

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 0$	$x = s - y$	$s \geq 1$
$M(y) = 0$	$u = x + 2y$	
$M(s) = 0$	$v = x - y$	
$M(u) = 0$		
$M(v) = 0$		

Example

- Asserting $s \geq 1$ pivot s and x (s is a dependent variable).

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 0$	$x = s - y$	$s \geq 1$
$M(y) = 0$	$u = s + y$	
$M(s) = 0$	$v = s - 2y$	
$M(u) = 0$		
$M(v) = 0$		

Example

- Asserting $s \geq 1$ update assignment.

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 0$	$x = s - y$	$s \geq 1$
$M(y) = 0$	$u = s + y$	
$M(s) = 1$	$v = s - 2y$	
$M(u) = 0$		
$M(v) = 0$		

Example

- Asserting $s \geq 1$ update dependent variables assignment.

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 1$	$x = s - y$	$s \geq 1$
$M(y) = 0$	$u = s + y$	
$M(s) = 1$	$v = s - 2y$	
$M(u) = 1$		
$M(v) = 1$		

Example

- Asserting $x \geq 0$

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 1$	$x = s - y$	$s \geq 1$
$M(y) = 0$	$u = s + y$	
$M(s) = 1$	$v = s - 2y$	
$M(u) = 1$		
$M(v) = 1$		

Example

- Asserting $x \geq 0$ assignment satisfies new bound.

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 1$	$x = s - y$	$s \geq 1$
$M(y) = 0$	$u = s + y$	$x \geq 0$
$M(s) = 1$	$v = s - 2y$	
$M(u) = 1$		
$M(v) = 1$		

Example

- Case split $\neg y \leq 1$

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 1$	$x = s - y$	$s \geq 1$
$M(y) = 0$	$u = s + y$	$x \geq 0$
$M(s) = 1$	$v = s - 2y$	
$M(u) = 1$		
$M(v) = 1$		

Example

- Case split $\neg y \leq 1$ assignment does not satisfies new bound.

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 1$	$x = s - y$	$s \geq 1$
$M(y) = 0$	$u = s + y$	$x \geq 0$
$M(s) = 1$	$v = s - 2y$	$y > 1$
$M(u) = 1$		
$M(v) = 1$		

Example

- Case split $\neg y \leq 1$ update assignment.

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 1$	$x = s - y$	$s \geq 1$
$M(y) = 1 + \delta$	$u = s + y$	$x \geq 0$
$M(s) = 1$	$v = s - 2y$	$y > 1$
$M(u) = 1$		
$M(v) = 1$		

Example

- Case split $\neg y \leq 1$ update dependent variables assignment.

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = -\delta$	$x = s - y$	$s \geq 1$
$M(y) = 1 + \delta$	$u = s + y$	$x \geq 0$
$M(s) = 1$	$v = s - 2y$	$y > 1$
$M(u) = 2 + \delta$		
$M(v) = -1 - 2\delta$		

Example

- Bound violation

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = -\delta$	$x = s - y$	$s \geq 1$
$M(y) = 1 + \delta$	$u = s + y$	$x \geq 0$
$M(s) = 1$	$v = s - 2y$	$y > 1$
$M(u) = 2 + \delta$		
$M(v) = -1 - 2\delta$		

Example

- Bound violation pivot x and s (x is a dependent variables).

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = -\delta$	$x = s - y$	$s \geq 1$
$M(y) = 1 + \delta$	$u = s + y$	$x \geq 0$
$M(s) = 1$	$v = s - 2y$	$y > 1$
$M(u) = 2 + \delta$		
$M(v) = -1 - 2\delta$		

Example

- Bound violation pivot x and s (x is a dependent variables).

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = -\delta$	$s = x + y$	$s \geq 1$
$M(y) = 1 + \delta$	$u = s + y$	$x \geq 0$
$M(s) = 1$	$v = s - 2y$	$y > 1$
$M(u) = 2 + \delta$		
$M(v) = -1 - 2\delta$		

Example

- Bound violation pivot x and s (x is a dependent variables).

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model

$$\begin{aligned} M(x) &= -\delta \\ M(y) &= 1+\delta \\ M(s) &= 1 \\ M(u) &= 2+\delta \\ M(v) &= -1-2\delta \end{aligned}$$

Equations

$$\begin{aligned} s &= x+y \\ u &= x+2y \\ v &= x-y \end{aligned}$$

Bounds

$$\begin{array}{ccc} s & \geq & 1 \\ x & \geq & 0 \\ \hline y & > & 1 \end{array}$$

Example

- Bound violation   update assignment.

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 0$	$s = x + y$	$s \geq 1$
$M(y) = 1 + \delta$	$u = x + 2y$	$x \geq 0$
$M(s) = 1$	$v = x - y$	$y > 1$
$M(u) = 2 + \delta$		
$M(v) = -1 - 2\delta$		

Example

- Bound violation update dependent variables assignment.

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 0$	$s = x + y$	$s \geq 1$
$M(y) = 1 + \delta$	$u = x + 2y$	$x \geq 0$
$M(s) = 1 + \delta$	$v = x - y$	$y > 1$
$M(u) = 2 + 2\delta$		
$M(v) = -1 - \delta$		

Example

- Theory propagation $x \geq 0, y > 1 \rightsquigarrow u > 2$

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 0$	$s = x + y$	$s \geq 1$
$M(y) = 1 + \delta$	$u = x + 2y$	$x \geq 0$
$M(s) = 1 + \delta$	$v = x - y$	$y > 1$
$M(u) = 2 + 2\delta$		
$M(v) = -1 - \delta$		

Example

- Theory propagation $u > 2 \leadsto \neg u \leq -1$

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 0$	$s = x + y$	$s \geq 1$
$M(y) = 1 + \delta$	$u = x + 2y$	$x \geq 0$
$M(s) = 1 + \delta$	$v = x - y$	$y > 1$
$M(u) = 2 + 2\delta$		$u > 2$
$M(v) = -1 - \delta$		

Example

- Boolean propagation $\neg y \leq 1 \rightsquigarrow v \geq 2$

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 0$	$s = x + y$	$s \geq 1$
$M(y) = 1 + \delta$	$u = x + 2y$	$x \geq 0$
$M(s) = 1 + \delta$	$v = x - y$	$y > 1$
$M(u) = 2 + 2\delta$		$u > 2$
$M(v) = -1 - \delta$		

Example

- Theory propagation $v \geq 2 \rightsquigarrow \neg v \leq -2$

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 0$	$s = x + y$	$s \geq 1$
$M(y) = 1 + \delta$	$u = x + 2y$	$x \geq 0$
$M(s) = 1 + \delta$	$v = x - y$	$y > 1$
$M(u) = 2 + 2\delta$		$u > 2$
$M(v) = -1 - \delta$		

Example

- Conflict empty clause

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 0$	$s = x + y$	$s \geq 1$
$M(y) = 1 + \delta$	$u = x + 2y$	$x \geq 0$
$M(s) = 1 + \delta$	$v = x - y$	$y > 1$
$M(u) = 2 + 2\delta$		$u > 2$
$M(v) = -1 - \delta$		

Example

- Backtracking

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model

$$\begin{aligned} M(x) &= 0 \\ M(y) &= 1+\delta \\ M(s) &= 1+\delta \\ M(u) &= 2+2\delta \\ M(v) &= -1-\delta \end{aligned}$$

Equations

$$\begin{aligned} s &= x+y \\ u &= x+2y \\ v &= x-y \end{aligned}$$

Bounds

$$\begin{aligned} s &\geq 1 \\ x &\geq 0 \end{aligned}$$

Example

- Asserting $y \leq 1$

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model

$$\begin{aligned} M(x) &= 0 \\ M(y) &= 1+\delta \\ M(s) &= 1+\delta \\ M(u) &= 2+2\delta \\ M(v) &= -1-\delta \end{aligned}$$

Equations

$$\begin{aligned} s &= x+y \\ u &= x+2y \\ v &= x-y \end{aligned}$$

Bounds

$$\begin{aligned} s &\geq 1 \\ x &\geq 0 \end{aligned}$$

Example

- Asserting $y \leq 1$ assignment does not satisfy new bound.

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 0$	$s = x + y$	$s \geq 1$
$M(y) = 1 + \delta$	$u = x + 2y$	$x \geq 0$
$M(s) = 1 + \delta$	$v = x - y$	$y \leq 1$
$M(u) = 2 + 2\delta$		
$M(v) = -1 - \delta$		

Example

- Asserting $y \leq 1$ update assignment.

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 0$	$s = x + y$	$s \geq 1$
$M(y) = 1$	$u = x + 2y$	$x \geq 0$
$M(s) = 1 + \delta$	$v = x - y$	$y \leq 1$
$M(u) = 2 + 2\delta$		
$M(v) = -1 - \delta$		

Example

- Asserting $y \leq 1$ update dependent variables assignment.

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 0$	$s = x + y$	$s \geq 1$
$M(y) = 1$	$u = x + 2y$	$x \geq 0$
$M(s) = 1$	$v = x - y$	$y \leq 1$
$M(u) = 2$		
$M(v) = -1$		

Example

- Theory propagation $s \geq 1, y \leq 1 \rightsquigarrow v \geq -1$

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 0$	$x = s - y$	$s \geq 1$
$M(y) = 1$	$u = s + y$	$x \geq 0$
$M(s) = 1$	$v = s - 2y$	$y \leq 1$
$M(u) = 2$		
$M(v) = -1$		

Example

- Theory propagation $v \geq -1 \rightsquigarrow \neg v \leq -2$

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 0$	$x = s - y$	$s \geq 1$
$M(y) = 1$	$u = s + y$	$x \geq 0$
$M(s) = 1$	$v = s - 2y$	$y \leq 1$
$M(u) = 2$		$v \geq -1$
$M(v) = -1$		

Example

- Boolean propagation $\neg v \leq -2 \rightsquigarrow v \geq 0$

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 0$	$x = s - y$	$s \geq 1$
$M(y) = 1$	$u = s + y$	$x \geq 0$
$M(s) = 1$	$v = s - 2y$	$y \leq 1$
$M(u) = 2$		$v \geq -1$
$M(v) = -1$		

Example

- Bound violation assignment does not satisfy new bound.

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 0$	$x = s - y$	$s \geq 1$
$M(y) = 1$	$u = s + y$	$x \geq 0$
$M(s) = 1$	$v = s - 2y$	$y \leq 1$
$M(u) = 2$		$v \geq 0$
$M(v) = -1$		

Example

- Bound violation pivot u and s (u is a dependent variable).

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 0$	$x = s - y$	$s \geq 1$
$M(y) = 1$	$u = s + y$	$x \geq 0$
$M(s) = 1$	$v = s - 2y$	$y \leq 1$
$M(u) = 2$		$v \geq 0$
$M(v) = -1$		

Example

- Bound violation pivot u and s (u is a dependent variable).

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 0$	$x = s - y$	$s \geq 1$
$M(y) = 1$	$u = s + y$	$x \geq 0$
$M(s) = 1$	$s = v + 2y$	$y \leq 1$
$M(u) = 2$		$v \geq 0$
$M(v) = -1$		

Example

- Bound violation pivot u and s (u is a dependent variable).

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 0$	$x = v + y$	$s \geq 1$
$M(y) = 1$	$u = v + 3y$	$x \geq 0$
$M(s) = 1$	$s = v + 2y$	$y \leq 1$
$M(u) = 2$		$v \geq 0$
$M(v) = -1$		

Example

- Bound violation   update assignment.

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 0$	$x = v + y$	$s \geq 1$
$M(y) = 1$	$u = v + 3y$	$x \geq 0$
$M(s) = 1$	$s = v + 2y$	$y \leq 1$
$M(u) = 2$		$v \geq 0$
$M(v) = 0$		

Example

- Bound violation update dependent variables assignment.

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 1$	$x = v + y$	$s \geq 1$
$M(y) = 1$	$u = v + 3y$	$x \geq 0$
$M(s) = 2$	$s = v + 2y$	$y \leq 1$
$M(u) = 3$		$v \geq 0$
$M(v) = 0$		

Example

- Boolean propagation $\neg v \leq -2 \rightsquigarrow u \leq -1$

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 1$	$x = v + y$	$s \geq 1$
$M(y) = 1$	$u = v + 3y$	$x \geq 0$
$M(s) = 2$	$s = v + 2y$	$y \leq 1$
$M(u) = 3$		$v \geq 0$
$M(v) = 0$		

Example

- Bound violation assignment does not satisfy new bound.

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 1$	$x = v + y$	$s \geq 1$
$M(y) = 1$	$u = v + 3y$	$x \geq 0$
$M(s) = 2$	$s = v + 2y$	$y \leq 1$
$M(u) = 3$		$v \geq 0$
$M(v) = 0$		$u \leq -1$

Example

- Bound violation pivot u and y (u is a dependent variable).

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 1$	$x = v + y$	$s \geq 1$
$M(y) = 1$	$u = v + 3y$	$x \geq 0$
$M(s) = 2$	$s = v + 2y$	$y \leq 1$
$M(u) = 3$		$v \geq 0$
$M(v) = 0$		$u \leq -1$

Example

- Bound violation pivot u and y (u is a dependent variable).

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 1$	$x = v + y$	$s \geq 1$
$M(y) = 1$	$y = \frac{1}{3}u - \frac{1}{3}v$	$x \geq 0$
$M(s) = 2$	$s = v + 2y$	$y \leq 1$
$M(u) = 3$		$v \geq 0$
$M(v) = 0$		$u \leq -1$

Example

- Bound violation pivot u and y (u is a dependent variable).

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 1$	$x = \frac{1}{3}u + \frac{2}{3}v$	$s \geq 1$
$M(y) = 1$	$y = \frac{1}{3}u - \frac{1}{3}v$	$x \geq 0$
$M(s) = 2$	$s = \frac{2}{3}u + \frac{1}{3}v$	$y \leq 1$
$M(u) = 3$		$v \geq 0$
$M(v) = 0$		$u \leq -1$

Example

- Bound violation update assignment.

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 1$	$x = \frac{1}{3}u + \frac{2}{3}v$	$s \geq 1$
$M(y) = 1$	$y = \frac{1}{3}u - \frac{1}{3}v$	$x \geq 0$
$M(s) = 2$	$s = \frac{2}{3}u + \frac{1}{3}v$	$y \leq 1$
$M(u) = -1$		$v \geq 0$
$M(v) = 0$		$u \leq -1$

Example

- Bound violation update dependent variables assignment.

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = -\frac{1}{3}$	$x = \frac{1}{3}u + \frac{2}{3}v$	$s \geq 1$
$M(y) = -\frac{1}{3}$	$y = \frac{1}{3}u - \frac{1}{3}v$	$x \geq 0$
$M(s) = -\frac{2}{3}$	$s = \frac{2}{3}u + \frac{1}{3}v$	$y \leq 1$
$M(u) = -1$		$v \geq 0$
$M(v) = 0$		$u \leq -1$

Example

- Bound violations

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = -\frac{1}{3}$	$x = \frac{1}{3}u + \frac{2}{3}v$	$s \geq 1$
$M(y) = -\frac{1}{3}$	$y = \frac{1}{3}u - \frac{1}{3}v$	$x \geq 0$
$M(s) = -\frac{2}{3}$	$s = \frac{2}{3}u + \frac{1}{3}v$	$y \leq 1$
$M(u) = -1$		$v \geq 0$
$M(v) = 0$		$u \leq -1$

Example

- Bound violations pivot s and v (s is a dependent variable).

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = -\frac{1}{3}$	$x = \frac{1}{3}u + \frac{2}{3}v$	$s \geq 1$
$M(y) = -\frac{1}{3}$	$y = \frac{1}{3}u - \frac{1}{3}v$	$x \geq 0$
$M(s) = -\frac{2}{3}$	$s = \frac{2}{3}u + \frac{1}{3}v$	$y \leq 1$
$M(u) = -1$		$v \geq 0$
$M(v) = 0$		$u \leq -1$

Example

- Bound violations pivot s and v (s is a dependent variable).

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = -\frac{1}{3}$	$x = \frac{1}{3}u + \frac{2}{3}v$	$s \geq 1$
$M(y) = -\frac{1}{3}$	$y = \frac{1}{3}u - \frac{1}{3}v$	$x \geq 0$
$M(s) = -\frac{2}{3}$	$v = 3s - 2u$	$y \leq 1$
$M(u) = -1$		$v \geq 0$
$M(v) = 0$		$u \leq -1$

Example

- Bound violations pivot s and v (s is a dependent variable).

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = -\frac{1}{3}$	$x = 2s - u$	$s \geq 1$
$M(y) = -\frac{1}{3}$	$y = -s + u$	$x \geq 0$
$M(s) = -\frac{2}{3}$	$v = 3s - 2u$	$y \leq 1$
$M(u) = -1$		$v \geq 0$
$M(v) = 0$		$u \leq -1$

Example

- Bound violations update assignment.

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = -\frac{1}{3}$	$x = 2s - u$	$s \geq 1$
$M(y) = -\frac{1}{3}$	$y = -s + u$	$x \geq 0$
$M(s) = 1$	$v = 3s - 2u$	$y \leq 1$
$M(u) = -1$		$v \geq 0$
$M(v) = 0$		$u \leq -1$

Example

- Bound violations update dependent variables assignment.

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 3$	$x = 2s - u$	$s \geq 1$
$M(y) = -2$	$y = -s + u$	$x \geq 0$
$M(s) = 1$	$v = 3s - 2u$	$y \leq 1$
$M(u) = -1$		$v \geq 0$
$M(v) = 5$		$u \leq -1$

Example

- Found satisfying assignment

$$s \geq 1, x \geq 0$$

$$(y \leq 1 \vee v \geq 2), (v \leq -2 \vee v \geq 0), (v \leq -2 \vee u \leq -1)$$

Model	Equations	Bounds
$M(x) = 3$	$x = 2s - u$	$s \geq 1$
$M(y) = -2$	$y = -s + u$	$x \geq 0$
$M(s) = 1$	$v = 3s - 2u$	$y \leq 1$
$M(u) = -1$		$v \geq 0$
$M(v) = 5$		$u \leq -1$

Correctness

Completeness: trivial

Soundness: also trivial

Termination: non trivial.

We cannot choose arbitrary variable to pivot.

Assume the variables are ordered.

Bland's rule: select the smallest basic variable c that does not satisfy its bounds, then select the smallest non-basic in the row of c that can be used for pivoting.

Too technical.

Uses the fact that a tableau has a finite number of configurations. Then, any infinite trace will have cycles.

Microsoft Research

Data-structures

- Array of rows (equations).
  - Each row is a dynamic array of tuples:
  - (coefficient, variable, pos_in_occs, is_dead)
- Each variable x has a "set" (dynamic array) of occurrences:
  - (row_idx, pos_in_row, is_dead)
- Each variable x has a "field" row[x]
  - row[x] is -1 if x is non basic
  - otherwise, row[x] contains the idx of the row containing x
- Each variable x has "fields": lower[x], upper[x], and value[x]

Data-structures

rows: array of rows (equations).

Each row is a dynamic array of tuples:

(coefficient, variable, pos_in_occs, is_dead)

occs[x]: Each variable x has a "set" (dynamic array) of occurrences:

(row_idx, pos_in_row, is_dead)

row[x]:

row[x] is -1 if x is non basic

otherwise, row[x] contains the idx of the row containing x

Other "fields": lower[x], upper[x], and value[x]

atoms[x]: atoms (assigned/unassigned) that contains x

Data-structures

$s_1 \equiv a + b$, $s_2 \equiv c - b$

$p_1 \equiv a \leq 0$, $p_2 \equiv 1 \leq s_1$, $p_3 \equiv 1 \leq s_2$

p_1, p_2 were already assigned

$a - s_1 + s_2 + c = 0$

$b - c + s_2 = 0$

$a \leq 0$, $1 \leq s_1$

$M(a) = 0$ value[a] = 0

$M(b) = -1$ value[a] = -1

$M(c) = 0$ value[c] = 0

$M(s_1) = 1$ value[s_1] = 1

$M(s_2) = 1$ value[s_2] = 1

rows = [

[(1, a, 0, t), (-1, s_1, 0, t), (1, s_2, 1, t), (1, c, 0, t)],

[(1,b, 0, t), (-1, c, 1, t), (1, s_2, 2, t)]]

occs[a] = [(0, 0, f)]

occs[b] = [(1,0,f)]

occs[c] = [(0,3,f), (1,1,f)]

occs[s_1] = [(0,1,f)]

occs[s_2] = [(0,0,t), (0,2,f), (1,2,f)]

row[a] = 0, row[b] = 1, row[c] = -1, …

upper[a] = 0, lower[s_1] = 1

atoms[a] = {p_1}, atoms[s_1] = {p_2}, …

Microsoft Research

Combining Theories

In practice, we need a combination of theories.

$b + 2 = c$ and $f(read(write(a,b,3), c-2)) \neq f(c-b+1)$

A theory is a set (potentially infinite) of first-order sentences.

Main questions:

Is the union of two theories $T1 \cup T2$ consistent?

Given a solvers for $T1$ and $T2$, how can we build a solver for $T1 \cup T2$?

Microsoft Research

Disjoint Theories

Two theories are disjoint if they do not share function/constant and predicate symbols.

= is the only exception.

Example:

The theories of arithmetic and arrays are disjoint.

Arithmetic symbols: {0, -1, 1, -2, 2, ..., +, -, *, >, <, ≥, ≤}

Array symbols: { read, write }

Purification

It is a different name for our "naming" subterms procedure.

$b + 2 = c$, $f(read(write(a,b,3), c\text{-}2)) \neq f(c\text{-}b\text{+}1)$

$b + 2 = c$, $v_6 \neq v_7$

$v_1 \equiv 3$, $v_2 \equiv write(a, b, v_1)$, $v_3 \equiv c\text{-}2$, $v_4 \equiv read(v_2, v_3)$,

$v_5 \equiv c\text{-}b\text{+}1$, $v_6 \equiv f(v_4)$, $v_7 \equiv f(v_5)$

Purification

It is a different name for our "naming" subterms procedure.

$b + 2 = c,\ f(read(write(a,b,3),\ c\text{-}2)) \neq f(c\text{-}b\text{+}1)$

$b + 2 = c,\ v_6 \neq v_7$

$v_1 \equiv 3,\ v_2 \equiv write(a,\ b,\ v_1),\ v_3 \equiv c\text{-}2,\ v_4 \equiv read(v_2,\ v_3),$

$v_5 \equiv c\text{-}b\text{+}1,\ v_6 \equiv f(v_4),\ v_7 \equiv f(v_5)$

$b + 2 = c,\ v_1 \equiv 3,\ v_3 \equiv c\text{-}2,\ v_5 \equiv c\text{-}b\text{+}1,$

$v_2 \equiv write(a,\ b,\ v_1),\ v_4 \equiv read(v_2,\ v_3),$

$v_6 \equiv f(v_4),\ v_7 \equiv f(v_5),\ v_6 \neq v_7$

Microsoft Research

Stably Infinite Theories

A theory is stably infinite if every satisfiable QFF is satisfiable in an infinite model.

EUF and arithmetic are stably infinite.

Bit-vectors are not.

Important Result

The union of two consistent, disjoint, stably infinite theories is consistent.

Convexity

A theory T is convex iff

for all finite sets S of literals and

for all $a_1 = b_1 \vee \ldots \vee a_n = b_n$

S implies $a_1 = b_1 \vee \ldots \vee a_n = b_n$

iff

S implies $a_i = b_i$ for some $1 \leq i \leq n$

Convexity: Results

Every convex theory with non trivial models is stably infinite.

All Horn equational theories are convex.

formulas of the form $s_1 \neq r_1 \vee \ldots \vee s_n \neq r_n \vee t = t'$

Linear rational arithmetic is convex.

Microsoft Research

Convexity: Negative Results

Linear integer arithmetic is not convex

$1 \leq a \leq 2,\ b = 1,\ c = 2$ implies $a = b \vee a = c$

Nonlinear arithmetic

$a^2 = 1,\ b = 1,\ c = -1$ implies $a = b \vee a = c$

Theory of bit-vectors

Theory of arrays

$c_1 = read(write(a, i, c_2), j),\ c_3 = read(a, j)$

implies $c_1 = c_2 \vee c_1 = c_3$

Microsoft Research

Combination of non-convex theories

EUF is convex (O(n log n))

IDL is non-convex (O(nm))

EUF $\cup$ IDL is NP-Complete

Reduce 3CNF to EUF $\cup$ IDL

For each boolean variable p_i add $0 \leq a_i \leq 1$

For each clause $p_1 \lor \neg p_2 \lor p_3$ add

$$f(a_1, a_2, a_3) \neq f(0, 1, 0)$$

Combination of non-convex theories

EUF is convex (O(n log n))

IDL is non-convex (O(nm))

EUF $\cup$ IDL is NP-Complete

Reduce 3CNF to EUF $\cup$ IDL

For each boolean variable p_i add $0 \leq a_i \leq 1$

For each clause $p_1 \vee \neg p_2 \vee p_3$ add

$$f(a_1, a_2, a_3) \neq f(0, 1, 0)$$

$\Downarrow$ implies

$$a_1 \neq 0 \vee a_2 \neq 1 \vee a_3 \neq 0$$

Microsoft Research

Nelson-Oppen Combination

Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be consistent, stably infinite theories over disjoint (countable) signatures. Assume satisfiability of conjunction of literals can decided in $O(T_1(n))$ and $O(T_2(n))$ time respectively. Then,

1. The combined theory $\mathcal{T}$ is consistent and stably infinite.
2. Satisfiability of quantifier free conjunction of literals in $\mathcal{T}$ can be decided in $O(2^{n^2} \times (T_1(n) + T_2(n))$.
3. If $\mathcal{T}_1$ and $\mathcal{T}_2$ are convex, then so is $\mathcal{T}$ and satisfiability in $\mathcal{T}$ is in $O(n^3 \times (T_1(n) + T_2(n)))$.

Microsoft Research

Nelson-Oppen Combination

The combination procedure:

Initial State: ϕ is a conjunction of literals over $\Sigma_1 \cup \Sigma_2$.

Purification: Preserving satisfiability transform ϕ into $\phi_1 \wedge \phi_2$,

such that, $\phi_i \in \Sigma_i$.

Interaction: Guess a partition of $\mathcal{V}(\phi_1) \cap \mathcal{V}(\phi_2)$ into disjoint

subsets. Express it as conjunction of literals ψ.

Example. The partition $\{x_1\}, \{x_2, x_3\}, \{x_4\}$ is represented

as $x_1 \neq x_2, x_1 \neq x_4, x_2 \neq x_4, x_2 = x_3$.

Component Procedures : Use individual procedures to decide

whether $\phi_i \wedge \psi$ is satisfiable.

Return: If both return yes, return yes. No, otherwise.

Microsoft Research

Soundness

Each step is satisfiability preserving.

Say ϕ is satisfiable (in the combination).

- Purification: $\phi_1 \wedge \phi_2$ is satisfiable.
- Iteration: for some partition ψ, $\phi_1 \wedge \phi_2 \wedge \psi$ is satisfiable.
- Component procedures: $\phi_1 \wedge \psi$ and $\phi_2 \wedge \psi$ are both satisfiable in component theories.
- Therefore, if the procedure return unsatisfiable, then ϕ is unsatisfiable.

Microsoft Research

Completeness

Suppose the procedure returns satisfiable.

- Let ψ be the partition and A and B be models of $\mathcal{T}_1 \wedge \phi_1 \wedge \psi$ and $\mathcal{T}_2 \wedge \phi_2 \wedge \psi$.
- The component theories are stably infinite. So, assume the models are infinite (of same cardinality).
- Let h be a bijection between $|A|$ and $|B|$ such that $h(A(x)) = B(x)$ for each shared variable.
- Extend B to $\bar{B}$ by interpretations of symbols in Σ_1: $\bar{B}(f)(b_1, \ldots, b_n) = h(A(f)(h^{-1}(b_1), \ldots, h^{-1}(b_n)))$
- $\bar{B}$ is a model of: $\mathcal{T}_1 \wedge \phi_1 \wedge \mathcal{T}_2 \wedge \phi_2 \wedge \psi$

oft Research

NO deterministic procedure (for convex theories)

Instead of guessing, we can deduce the equalities to be shared.

Purification: no changes.

Interaction: Deduce an equality $x = y$:

$$\mathcal{T}_1 \vdash (\phi_1 \Rightarrow x = y)$$

Update $\phi_2 := \phi_2 \wedge x = y$. And vice-versa. Repeat until no further changes.

Component Procedures : Use individual procedures to decide whether ϕ_i is satisfiable.

Remark: $\mathcal{T}_i \vdash (\phi_i \Rightarrow x = y)$ iff $\phi_i \wedge x \neq y$ is not satisfiable in $\mathcal{T}_i$.

NO deterministic procedure Completeness

Assume the theories are convex.

- Suppose ϕ_i is satisfiable.
- Let E be the set of equalities $x_j = x_k$ $(j \neq k)$ such that, $\mathcal{T}_i \not\vdash \phi_i \Rightarrow x_j = x_k$.
- By convexity, $\mathcal{T}_i \not\vdash \phi_i \Rightarrow \bigvee_E x_j = x_k$.
- $\phi_i \wedge \bigwedge_E x_j \neq x_k$ is satisfiable.
- The proof now is identical to the nondeterministic case.
- Sharing equalities is sufficient, because a theory $\mathcal{T}_1$ can assume that $x^B \neq y^B$ whenever $x = y$ is not implied by $\mathcal{T}_2$ and vice versa.

Microsoft Research

NO procedure: Example

$b + 2 = c$, $f(read(write(a,b,3), c-2)) \neq f(c-b+1)$

Arithmetic

$b + 2 = c$,

$v_1 \equiv 3$,

$v_3 \equiv c-2$,

$v_5 \equiv c-b+1$

Arrays

$v_2 \equiv write(a, b, v_1)$,

$v_4 \equiv read(v_2, v_3)$

EUF

$v_6 \equiv f(v_4)$,

$v_7 \equiv f(v_5)$,

$v_6 \neq v_7$

NO procedure: Example

$b + 2 = c, f(read(write(a,b,3), c-2)) \neq f(c-b+1)$

Arithmetic

$b + 2 = \mathbf{c},$

$v_1 \equiv 3,$

$v_3 \equiv c-2,$

$v_5 \equiv c-b+1$

Arrays

$v_2 \equiv write(a, b, v_1),$

$v_4 \equiv read(v_2, v_3)$

EUF

$v_6 \equiv f(v_4),$

$v_7 \equiv f(v_5),$

$v_6 \neq v_7$

Substituting c

NO procedure: Example

$b + 2 = c, f(read(write(a,b,3), c\text{-}2)) \neq f(c\text{-}b+1)$

Arithmetic

$b + 2 = c,$

$v_1 \equiv 3,$

$\mathbf{v_3 \equiv b},$

$v_5 \equiv 3$

Arrays

$v_2 \equiv write(a, b, v_1),$

$v_4 \equiv read(v_2, v_3),$

EUF

$v_6 \equiv f(v_4),$

$v_7 \equiv f(v_5),$

$v_6 \neq v_7$

Propagating $v_3 = b$

NO procedure: Example

$b + 2 = c,\ f(read(write(a,b,3), c\text{-}2)) \neq f(c\text{-}b+1)$

Arithmetic

$b + 2 = c,$

$v_1 \equiv 3,$

$v_3 \equiv b,$

$v_5 \equiv 3$

Arrays

$\mathbf{v_2} \equiv write(a, \mathbf{b}, v_1),$

$v_4 \equiv read(\mathbf{v_2}, \mathbf{v_3}),$

$v_3 = b$

EUF

$v_6 \equiv f(v_4),$

$v_7 \equiv f(v_5),$

$v_6 \neq v_7,$

$v_3 = b$

Deducing $v_4 = v_1$

NO procedure: Example

$b + 2 = c, f(read(write(a,b,3), c\text{-}2)) \neq f(c\text{-}b+1)$

Arithmetic

$b + 2 = c,$

$v_1 \equiv 3,$

$v_3 \equiv b,$

$v_5 \equiv 3$

Arrays

$v_2 \equiv write(a, b, v_1),$

$v_4 \equiv read(v_2, v_3),$

$v_3 = b,$

$\mathbf{v_4 = v_1}$

EUF

$v_6 \equiv f(v_4),$

$v_7 \equiv f(v_5),$

$v_6 \neq v_7,$

$v_3 = b$

Propagating $v_4 = v_1$

Microsoft Research

NO procedure: Example

$b + 2 = c, f(read(write(a,b,3), c-2)) \neq f(c-b+1)$

Arithmetic

$b + 2 = c,$

$\mathbf{v_1 \equiv 3},$

$v_3 \equiv b,$

$\mathbf{v_5 \equiv 3},$

$v_4 = v_1$

Arrays

$v_2 \equiv write(a, b, v_1),$

$v_4 \equiv read(v_2, v_3),$

$v_3 = b,$

$v_4 = v_1$

EUF

$v_6 \equiv f(v_4),$

$v_7 \equiv f(v_5),$

$v_6 \neq v_7,$

$v_3 = b,$

$v_4 = v_1$

Propagating $v_5 = v_1$

NO procedure: Example

$b + 2 = c,\ f(read(write(a,b,3),\ c-2)) \neq f(c-b+1)$

Arithmetic

$b + 2 = c,$

$v_1 \equiv 3,$

$v_3 \equiv b,$

$v_5 \equiv 3,$

$v_4 = v_1$

Arrays

$v_2 \equiv write(a,\ b,\ v_1),$

$v_4 \equiv read(v_2,\ v_3),$

$v_3 = b,$

$v_4 = v_1$

EUF

$v_6 \equiv f(\mathbf{v_4}),$

$v_7 \equiv f(\mathbf{v_5}),$

$v_6 \neq v_7,$

$v_3 = b,$

$\mathbf{v_4 = v_1},$

$\mathbf{v_5 = v_1}$

Congruence: $v_6 = v_7$

NO procedure: Example

$b + 2 = c, f(read(write(a,b,3), c-2)) \neq f(c-b+1)$

Arithmetic

$b + 2 = c,$

$v_1 \equiv 3,$

$v_3 \equiv b,$

$v_5 \equiv 3,$

$v_4 = v_1$

Unsatisfiable

Arrays

$v_2 \equiv write(a, b, v_1),$

$v_4 \equiv read(v_2, v_3),$

$v_3 = b,$

$v_4 = v_1$

EUF

$v_6 \equiv f(v_4),$

$v_7 \equiv f(v_5),$

$\mathbf{v_6 \neq v_7},$

$v_3 = b,$

$v_4 = v_1,$

$v_5 = v_1,$

$\mathbf{v_6 = v_7}$

Microsoft Research

NO deterministic procedure

Deterministic procedure may fail for non-convex theories.

$0 \leq a \leq 1$, $0 \leq b \leq 1$, $0 \leq c \leq 1$,

$f(a) \neq f(b)$,

$f(a) \neq f(c)$,

$f(b) \neq f(c)$

Microsoft Research

Combining Procedures in Practice

Propagate all implied equalities.

- Deterministic Nelson-Oppen.
- Complete only for convex theories.
- It may be expensive for some theories.

Delayed Theory Combination.

- Nondeterministic Nelson-Oppen.
- Create set of interface equalities $(x = y)$ between shared variables.
- Use SAT solver to guess the partition.
- Disadvantage: the number of additional equality literals is quadratic in the number of shared variables.

Combining Procedures in Practice

Common to these methods is that they are pessimistic about which equalities are propagated.

Model-based Theory Combination

- Optimistic approach.
- Use a candidate model M_i for one of the theories $\mathcal{T}_i$ and propagate all equalities implied by the candidate model, hedging that other theories will agree.

$$\textbf{if } M_i \models \mathcal{T}_i \cup \Gamma_i \cup \{u = v\} \textbf{ then } \text{propagate } u = v \ .$$

- If not, use backtracking to fix the model.
- It is cheaper to enumerate equalities that are implied in a particular model than of all models.

Microsoft Research

Example

$$x = f(y-1), f(x) \neq f(y), 0 \leq x \leq 1, 0 \leq y \leq 1$$

Purifying

Example

$$x = f(z), f(x) \neq f(y), 0 \leq x \leq 1, 0 \leq y \leq 1, z = y - 1$$

Microsoft Research

Example

$\mathcal{T}_E$			$\mathcal{T}_A$	
Literals	*Eq. Classes*	*Model*	*Literals*	*Model*
$x = f(z)$	$\{x, f(z)\}$	$E(x) = *_1$	$0 \le x \le 1$	$A(x) = 0$
$f(x) \neq f(y)$	$\{y\}$	$E(y) = *_2$	$0 \le y \le 1$	$A(y) = 0$
	$\{z\}$	$E(z) = *_3$	$z = y - 1$	$A(z) = -1$
	$\{f(x)\}$	$E(f) = \{*_1 \mapsto *_4,$		
	$\{f(y)\}$	$*_2 \mapsto *_5,$		
		$*_3 \mapsto *_1,$		
		else $\mapsto *_6\}$		

Assume x = y

Example

$\mathcal{T}_E$			$\mathcal{T}_A$	
Literals	*Eq. Classes*	*Model*	*Literals*	*Model*
$x = f(z)$	$\{x, y, f(z)\}$	$E(x) = *_1$	$0 \leq x \leq 1$	$A(x) = 0$
$f(x) \neq f(y)$	$\{z\}$	$E(y) = *_1$	$0 \leq y \leq 1$	$A(y) = 0$
$x = y$	$\{f(x), f(y)\}$	$E(z) = *_2$	$z = y - 1$	$A(z) = -1$
		$E(f) = \{*_1 \mapsto *_3,$	$x = y$	
		$*_2 \mapsto *_1,$		
		else $\mapsto *_4\}$		

Unsatisfiable

Example

$\mathcal{T}_E$			$\mathcal{T}_A$	
Literals	*Eq. Classes*	*Model*	*Literals*	*Model*
$x = f(z)$	$\{x, f(z)\}$	$E(x) = *_1$	$0 \leq x \leq 1$	$A(x) = 0$
$f(x) \neq f(y)$	$\{y\}$	$E(y) = *_2$	$0 \leq y \leq 1$	$A(y) = 0$
$x \neq y$	$\{z\}$	$E(z) = *_3$	$z = y - 1$	$A(z) = -1$
	$\{f(x)\}$	$E(f) = \{*_1 \mapsto *_4,$	$x \neq y$	
	$\{f(y)\}$	$*_2 \mapsto *_5,$		
		$*_3 \mapsto *_1,$		
		else $\mapsto *_6\}$		

Backtrack, and assert $x \neq y$.

$\mathcal{T}_A$ model need to be fixed.

Example

$\mathcal{T}_E$			$\mathcal{T}_A$	
Literals	*Eq. Classes*	*Model*	*Literals*	*Model*
$x = f(z)$	$\{x, f(z)\}$	$E(x) = *_1$	$0 \leq x \leq 1$	$A(x) = 0$
$f(x) \neq f(y)$	$\{y\}$	$E(y) = *_2$	$0 \leq y \leq 1$	$A(y) = 1$
$x \neq y$	$\{z\}$	$E(z) = *_3$	$z = y - 1$	$A(z) = 0$
	$\{f(x)\}$	$E(f) = \{*_1 \mapsto *_4,$	$x \neq y$	
	$\{f(y)\}$	$*_2 \mapsto *_5,$		
		$*_3 \mapsto *_1,$		
		else $\mapsto *_6\}$		

Assume x = z

Example

$\mathcal{T}_E$			$\mathcal{T}_A$	
Literals	*Eq. Classes*	*Model*	*Literals*	*Model*
$x = f(z)$	$\{x, z,$	$E(x) = *_1$	$0 \leq x \leq 1$	$A(x) = 0$
$f(x) \neq f(y)$	$f(x), f(z)\}$	$E(y) = *_2$	$0 \leq y \leq 1$	$A(y) = 1$
$x \neq y$	$\{y\}$	$E(z) = *_1$	$z = y - 1$	$A(z) = 0$
$x = z$	$\{f(y)\}$	$E(f) = \{*_1 \mapsto *_1,$	$x \neq y$	
		$*_2 \mapsto *_3,$	$x = z$	
		else $\mapsto *_4\}$		

Satisfiable

Example

$\mathcal{T}_E$			$\mathcal{T}_A$	
Literals	*Eq. Classes*	*Model*	*Literals*	*Model*
$x = f(z)$	$\{x, z,$	$E(x) = *_1$	$0 \leq x \leq 1$	$A(x) = 0$
$f(x) \neq f(y)$	$f(x), f(z)\}$	$E(y) = *_2$	$0 \leq y \leq 1$	$A(y) = 1$
$x \neq y$	$\{y\}$	$E(z) = *_1$	$z = y - 1$	$A(z) = 0$
$x = z$	$\{f(y)\}$	$E(f) = \{*_1 \mapsto *_1,$	$x \neq y$	
		$*_2 \mapsto *_3,$	$x = z$	
		else $\mapsto *_4\}$		

Let h be the bijection between $|E|$ and $|A|$.

$$h = \{*_1 \mapsto 0, *_2 \mapsto 1, *_3 \mapsto -1, *_4 \mapsto 2, \ldots\}$$

Example

$\mathcal{T}_E$		$\mathcal{T}_A$	
Literals	*Model*	*Literals*	*Model*
$x = f(z)$	$E(x) = *_1$	$0 \leq x \leq 1$	$A(x) = 0$
$f(x) \neq f(y)$	$E(y) = *_2$	$0 \leq y \leq 1$	$A(y) = 1$
$x \neq y$	$E(z) = *_1$	$z = y - 1$	$A(z) = 0$
$x = z$	$E(f) = \{*_1 \mapsto *_1,$	$x \neq y$	$A(f) = \{0 \mapsto 0$
	$*_2 \mapsto *_3,$	$x = z$	$1 \mapsto -1$
	else $\mapsto *_4\}$		*else* $\mapsto 2\}$

Extending A using h.

$$h = \{*_1 \mapsto 0, *_2 \mapsto 1, *_3 \mapsto -1, *_4 \mapsto 2, \ldots\}$$

Model Mutation

Sometimes $M(x) = M(y)$ by accident.

$$\bigwedge_{i=1}^{N} f(x_i) \geq 0 \ \wedge \ x_i \geq 0$$

Model mutation: diversify the current model.

Freedom Intervals

Model mutation without pivoting

For each non basic variable x_j compute $[L_j, U_j]$

Each row containing x_j enforces a limit on how much it can be increase and/or decreased without violating the bounds of the basic variable in the row.

Opportunistic Equality Propagation

We say a variable is fixed if the lower and upper bound are the same.

$$1 \leq x \leq 1$$

A polynomial P is fixed if all its variables are fixed.

Given a fixed polynomial P of the forma $2x_1 + x_2$,
we use M(P) to denote $2M(x_1) + M(x_2)$

Opportunistic Equality Propagation

FixedEq

$$l_i \le x_i \le u_i,\ l_j \le x_j \le u_j \implies x_i = x_j \ \text{ if } \ l_i = u_i = l_j = u_j$$

EqRow

$$x_i = x_j + P \implies x_i = x_j \ \text{ if } \ P \text{ is fixed, and } \mathsf{M}(P) = 0$$

EqOffsetRows

$$\begin{aligned} x_i &= x_k + P_1 \\ x_j &= x_k + P_2 \end{aligned} \implies x_i = x_j \ \text{ if } \ \begin{cases} P_1 \text{ and } P_2 \text{ are fixed, and} \\ \mathsf{M}(P_1) = \mathsf{M}(P_2) \end{cases}$$

EqRows

$$\begin{aligned} x_i &= P + P_1 \\ x_j &= P + P_2 \end{aligned} \implies x_i = x_j \ \text{ if } \ \begin{cases} P_1 \text{ and } P_2 \text{ are fixed, and} \\ \mathsf{M}(P_1) = \mathsf{M}(P_2) \end{cases}$$

Non-stably infinite theories in practice

Bit-vector theory is not stably-infinite.

How can we support it?

Solution: add a predicate *is-bv*(x) to the bit-vector theory (intuition: *is-bv*(x) is true iff x is a bitvector).

The result of the bit-vector operation *op*(x, y) is not specified if $\neg$*is-bv*(x) or $\neg$*is-bv*(y).

The new bit-vector theory is stably-infinite.

Reduction Functions

A reduction function reduces the satifiability problem for a complex theory into the satisfiability problem of a simpler theory.

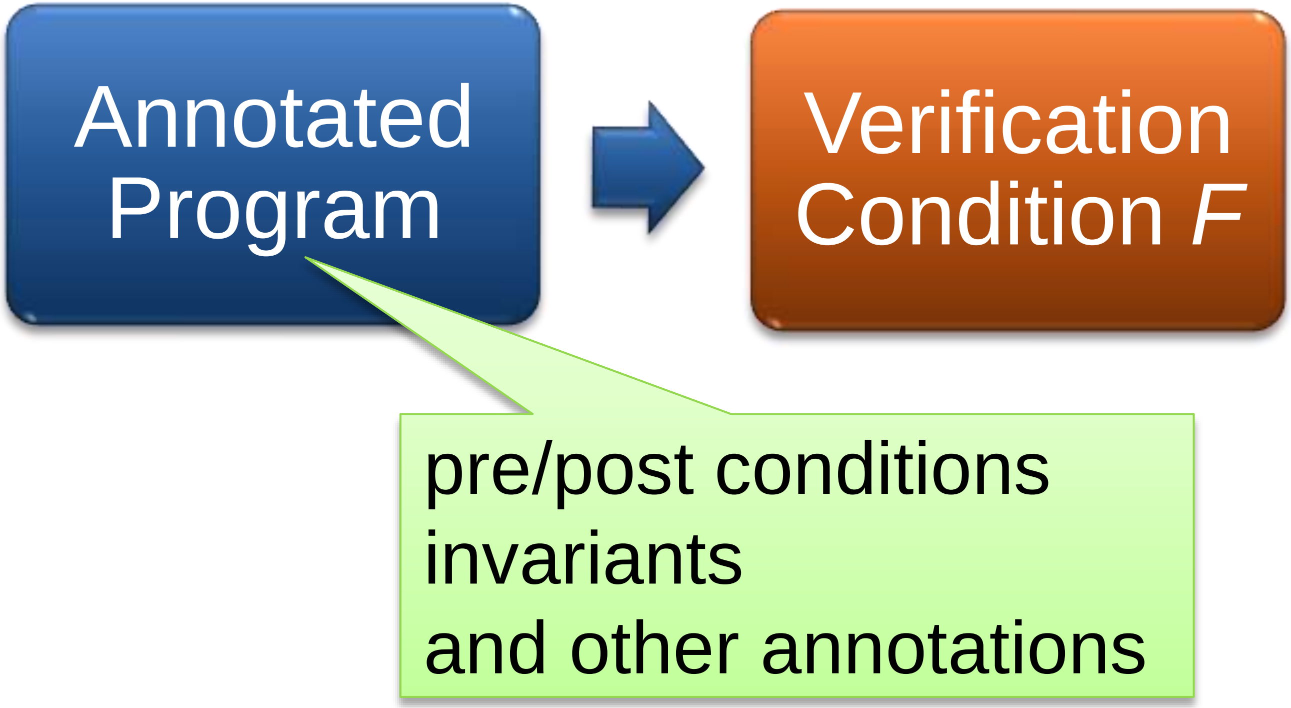
Ackermannization is a reduction function.

Reduction Functions

Theory of commutative functions.

- $\forall x, y. f(x, y) = f(y, x)$
- Reduction to EUF
- For every $f(a, b)$ in ϕ, do $\phi := \phi \wedge f(a, b) = f(b, a)$.

Verifying Compilers



Microsoft Research

Verification conditions: Structure

∀ Axioms
(non-ground)

BIG
and-or
tree
(ground)

Control & Data
Flow

Main Challenge

- Quantifiers, quantifiers, quantifiers, …
- Modeling the runtime

$$\forall\, h,o,f: \\ \quad IsHeap(h) \wedge o \neq null \wedge read(h, o, alloc) = t \\ \quad \Rightarrow \\ \quad read(h,o, f) = null \vee read(h, read(h,o,f),alloc) = t$$

Main Challenge

- Quantifiers, quantifiers, quantifiers, …
- Modeling the runtime
- Frame axioms

$\forall$ o, f:

$$o \neq null \wedge read(h_0, o, alloc) = t \Rightarrow$$
$$read(h_1,o,f) = read(h_0,o,f) \vee (o,f) \in M$$

Microsoft Research

Main Challenge

- Quantifiers, quantifiers, quantifiers, …
- Modeling the runtime
- Frame axioms
- User provided assertions

$\forall$ i,j: i $\leq$ j $\Rightarrow$ read(a,i) $\leq$ read(b,j)

Microsoft Research

Main Challenge

- Quantifiers, quantifiers, quantifiers, …
- Modeling the runtime
- Frame axioms
- User provided assertions
- Theories

$\forall$ x: p(x,x)

$\forall$ x,y,z: p(x,y), p(y,z) $\Rightarrow$ p(x,z)

$\forall$ x,y: p(x,y), p(y,x) $\Rightarrow$ x = y

Microsoft Research

Main Challenge

- Quantifiers, quantifiers, quantifiers, …
- Modeling the runtime
- Frame axioms
- User provided assertions
- Theories
- Solver must be fast in satisfiable instances.

We want to find bugs!

Microsoft Research

Some statistics

- Grand challenge: Microsoft Hypervisor
- 70k lines of dense C code
- VCs have several Mb
- Thousands of non ground clauses
- Developers are willing to wait at most 5 min per VC

Microsoft Research

Many Approaches

- Heuristic quantifier instantiation
- Combining SMT with Saturation provers
- Complete quantifier instantiation
- Decidable fragments
- Model based quantifier instantiation

Microsoft Research

E-matching & Quantifier instantiation

- SMT solvers use heuristic quantifier instantiation.
- E-matching (matching modulo equalities).
- Example:

$\forall\ x: f(g(x)) = x \ \{\ f(g(x))\ \}$

$a = g(b),$

$b = c,$

$f(a) \neq c$

Trigger

Microsoft Research

E-matching & Quantifier instantiation

- SMT solvers use heuristic quantifier instantiation.
- E-matching (matching modulo equalities).
- Example:

$\forall\ x: f(g(x)) = x\ \{\ f(g(x))\ \}$

$a = g(b)$,

$b = c$,

$f(a) \neq c$

$\xrightarrow{x=b}$ $f(g(b)) = b$

Equalities and ground terms come from the partial model M

E-matching: why do we use it?

- Integrates smoothly with DPLL.
- Software verification problems are big & shallow.
- Decides useful theories:
  - Arrays
  - Partial orders
  - …

Efficient E-matching

- E-matching is NP-Hard.
- In practice

Problem	Indexing Technique
Fast retrieval	E-matching code trees
Incremental E-Matching	Inverted path index

E-matching code trees

Trigger:

f(x1, g(x1, a), h(x2), b)

Compiler

Instructions:

1. init(f, 2)
2. check(r4, b, 3)
3. bind(r2, g, r5, 4)
4. compare(r1, r5, 5)
5. check(r6, a, 6)
6. bind(r3, h, r7, 7)
7. yield(r1, r7)

Similar triggers share several instructions.

Combine code sequences in a code tree

E-matching: Limitations

- E-matching needs ground seeds.

  $\forall x$: p(x),

  $\forall x$: not p(x)

E-matching: Limitations

- E-matching needs ground seeds.
- Bad user provided triggers:

  $\forall x\colon f(g(x))=x$ { $f(g(x))$ }

  $g(a) = c$,

  $g(b) = c$,

  $a \neq b$

Trigger is too restrictive

Microsoft Research

E-matching: Limitations

- E-matching needs ground seeds.
- Bad user provided triggers:

  $\forall x: f(g(x))=x \; \{ g(x) \}$

  $g(a) = c,$

  $g(b) = c,$

  $a \neq b$

More "liberal" trigger

Microsoft Research

E-matching: Limitations

- E-matching needs ground seeds.
- Bad user provided triggers:

$\forall x: f(g(x))=x \; \{ g(x) \}$

$g(a) = c,$

$g(b) = c,$

$a \neq b,$

$f(g(a)) = a,$

$f(g(b)) = b \quad \Longrightarrow \quad a=b$

Microsoft Research

E-matching: Limitations

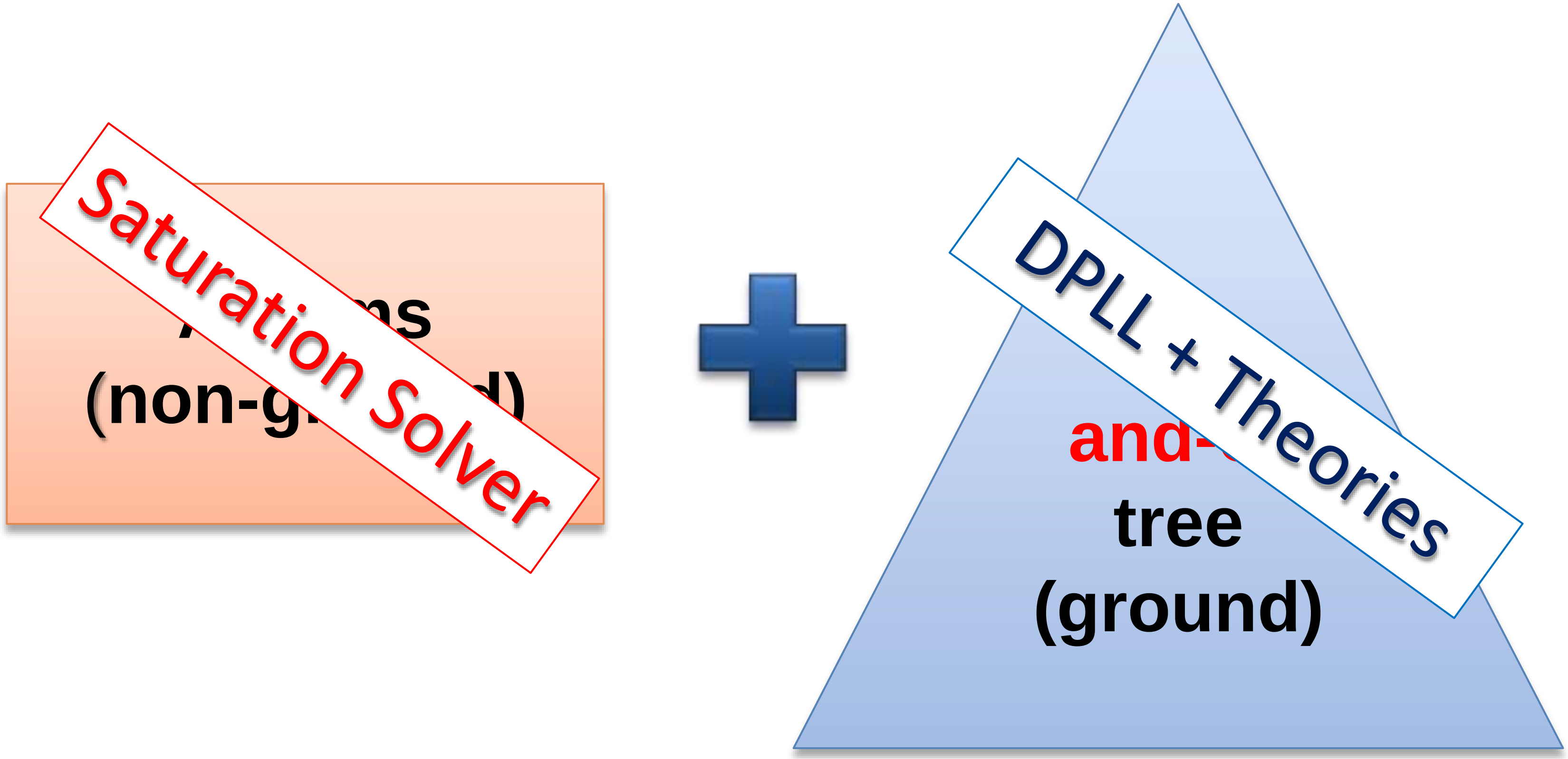
- E-matching needs ground seeds.
- Bad user provided triggers.
- It is not refutationally complete.

False positives

Microsoft Research

DPLL(Γ)

- Tight integration: DPLL + Saturation solver.



DPLL(Γ)

- Inference rule: $$\frac{C_1 \quad \ldots \quad C_n}{C}$$
- DPLL(Γ) is parametric.
- Examples:
  - Resolution
  - Superposition calculus
  - …

DPLL(Γ)

M | F

Partial model

Set of clauses

DPLL(Γ): Deduce I

$p(a) \mid p(a) \vee q(a),\ \forall x{:}\ \neg p(x) \vee r(x),\ \forall x{:}\ p(x) \vee s(x)$

Microsoft Research

DPLL(Γ): Deduce I

$p(a) \mid p(a) \vee q(a), \neg p(x) \vee r(x), p(x) \vee s(x)$

Microsoft Research

DPLL(Γ): Deduce I

$p(a) \mid p(a) \vee q(a), \neg p(x) \vee r(x), p(x) \vee s(x)$

$\Downarrow$ **Resolution**

$p(a) \mid p(a) \vee q(a), \neg p(x) \vee r(x), p(x) \vee s(x), r(x) \vee s(x)$

Microsoft Research

DPLL(Γ): Deduce II

- Using ground atoms from M:

$$M \mid F$$

- Main issue: backtracking.
- Hypothetical clauses:

$$H \triangleright C$$

Track literals from M used to derive C

(hypothesis) Ground literals

(regular) Clause

Microsoft Research

DPLL(Γ): Deduce II

$$p(a) \mid p(a) \lor q(a),\ \neg p(x) \lor r(x)$$

$$\frac{p(a),\quad \neg p(x) \lor r(x)}{r(a)}$$

$$p(a) \mid p(a) \lor q(a),\ \neg p(x) \lor r(x),\ p(a) \triangleright r(a)$$

Microsoft Research

DPLL(Γ): Backtracking

$p(a), r(a) \mid p(a) \vee q(a), \neg p(a) \vee \neg r(a), \ \mathbf{p(a)} \triangleright \mathbf{r(a)}, \ldots$

Microsoft Research

DPLL(Γ): Backtracking

$p(a), r(a) \mid p(a) \vee q(a), \neg p(a) \vee \neg r(a), p(a) \rhd r(a), \ldots$

p(a) is removed from M

$\neg p(a) \mid p(a) \vee q(a), \neg p(a) \vee \neg r(a), \ldots$

Microsoft Research

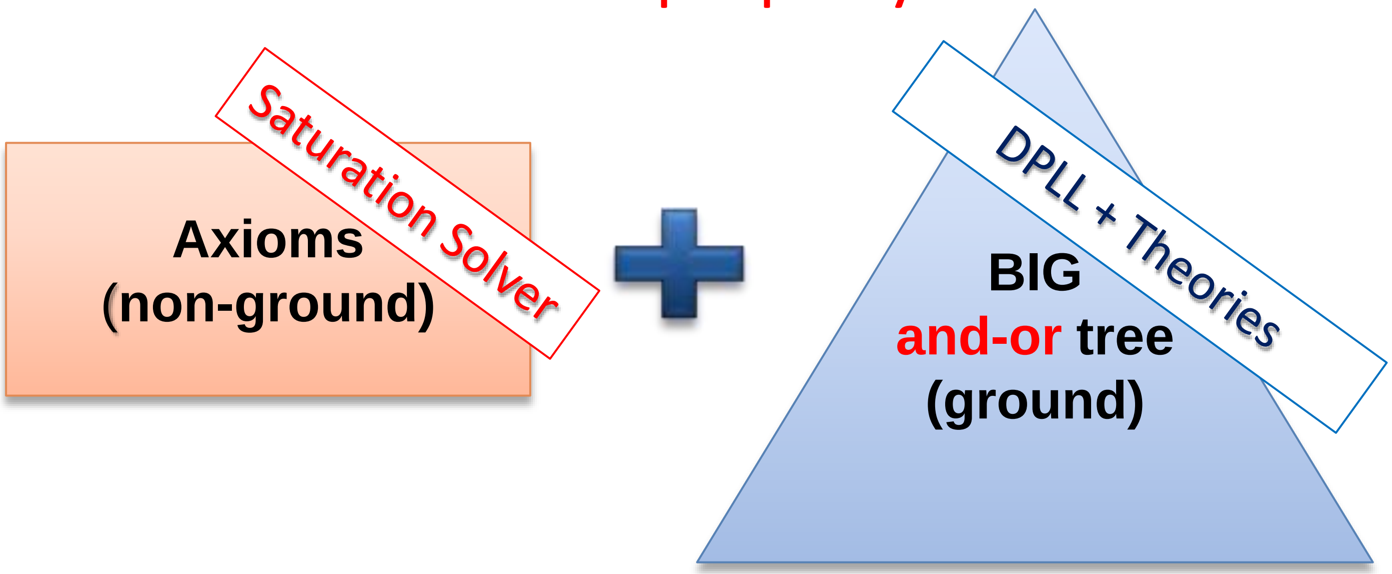
DPLL(Γ): Improvement

- Saturation solver ignores non-unit ground clauses.

$$p(a) \mid p(a) \lor q(a),\ \neg p(x) \lor r(x)$$

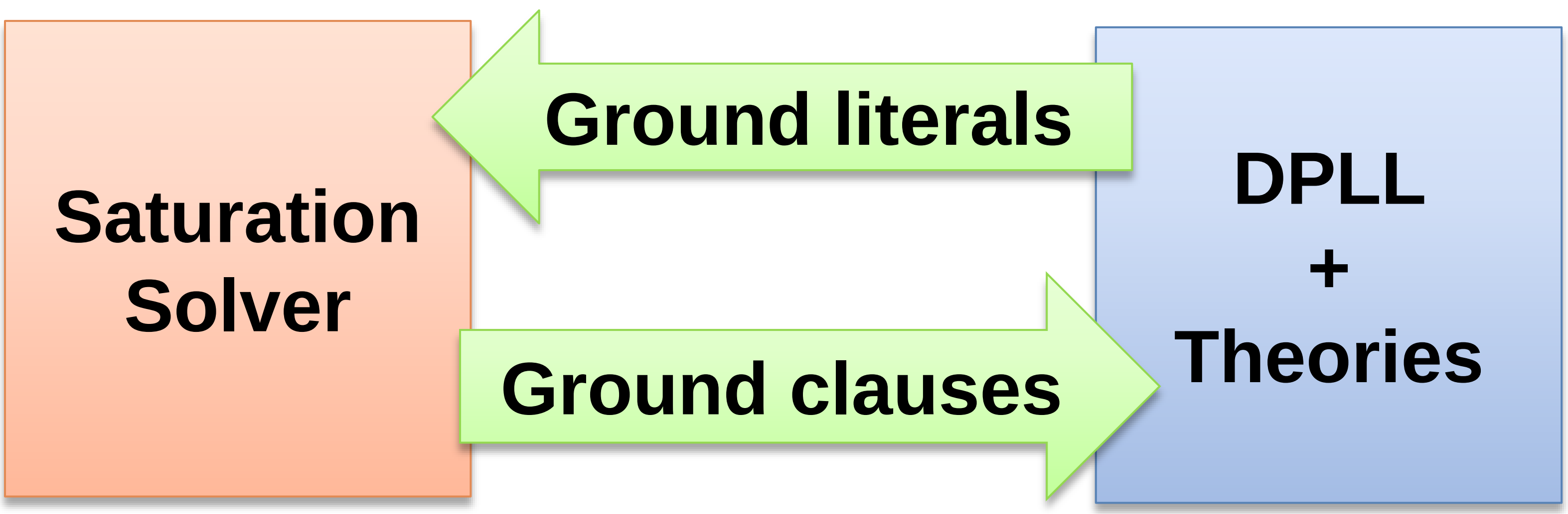
DPLL(Γ): Improvement

- Saturation solver ignores non-unit ground clauses.
- It is still refutanionally complete if:
  - Γ has the reduction property.



DPLL(Γ): Improvement

- Saturation solver ignores non-unit ground clauses.
- It is still refutanionally complete if:
  - Γ has the reduction property.



DPLL(Γ): Problem

- Interpreted symtbols

  $\neg$(f(a) > 2),   f(x) > 5

- It is refutationally complete if
  - Interpreted symbols only occur in ground clauses
  - Non ground clauses are variable inactive
  - “Good” ordering is used

Notation Remainder

$\forall x_1, x_2: \neg p(x_1, x_2) \vee f(x_1) = f(x_2) + 1,$
$p(a,b), a < b + 1$

Notation Remainder

$\neg p(x_1, x_2) \vee f(x_1) = f(x_2) + 1,$

$p(a,b),\ a < b + 1$

Microsoft Research

Essentially uninterpreted fragment

- Variables appear only as arguments of uninterpreted symbols.

$$f(g(x_1) + a) < g(x_1) \vee h(f(x_1), x_2) = 0$$

$$f(x_1+x_2) \leq f(x_1) + f(x_2)$$

Basic Idea

Given a set of formulas F,
build an equisatisfiable set of quantifier-free formulas F*

"Domain" of f is the set of ground terms A_f

$t \in A_f$ if there is a ground term f(t)

Suppose

1. We have a clause C[f(x)] containing f(x).
2. We have f(t).

→

Instantiate x with t: C[f(t)].

Example

F **F***

$g(x_1, x_2) = 0 \vee h(x_2) = 0,$

$g(f(x_1),b) + 1 \leq f(x_1),$

$h(c) = 1,$

$f(a) = 0$

Microsoft Research

Example

F

$g(x_1, x_2) = 0 \vee h(x_2) = 0,$
$g(f(x_1),b) + 1 \leq f(x_1),$
$h(c) = 1,$
$f(a) = 0$

F*

$h(c) = 1,$
$f(a) = 0$

Copy quantifier-free formulas

“Domains”:
A_f: { a }
A_g: { }
A_h: { c }

Example

F

$g(x_1, x_2) = 0 \vee h(x_2) = 0,$

$g(f(x_1),b) + 1 \leq f(x_1),$

$h(c) = 1,$

$f(a) = 0$

F*

$h(c) = 1,$

$f(a) = 0,$

"Domains":

$A_f : \{ a \}$

$A_g : \{ \}$

$A_h : \{ c \}$

Example

F

$g(x_1, x_2) = 0 \vee h(x_2) = 0,$

$g(f(x_1),b) + 1 \leq f(x_1),$

$h(c) = 1,$

$f(a) = 0$

$\Rightarrow$

F*

$h(c) = 1,$

$f(a) = 0,$

$g(f(a),b) + 1 \leq f(a)$

"Domains":

$A_f : \{ a \}$

$A_g : \{ [f(a), b] \}$

$A_h : \{ c \}$

Microsoft Research

Example

F

$g(x_1, x_2) = 0 \vee h(x_2) = 0,$
$g(f(x_1),b) + 1 \leq f(x_1),$
$h(c) = 1,$
$f(a) = 0$

F*

$h(c) = 1,$
$f(a) = 0,$
$g(f(a),b) + 1 \leq f(a),$

"Domains":

$A_f : \{ a \}$
$A_g : \{ [f(a), b] \}$
$A_h : \{ c \}$

Example

F

$g(x_1, x_2) = 0 \vee h(x_2) = 0,$

$g(f(x_1),b) + 1 \leq f(x_1),$

$h(c) = 1,$

$f(a) = 0$

F*

$h(c) = 1,$

$f(a) = 0,$

$g(f(a),b) + 1 \leq f(a),$

$g(f(a), b) = 0 \vee h(b) = 0$

"Domains":

A_f : { a }

A_g : { [f(a), b] }

A_h : { c, b }

Example

F

$g(x_1, x_2) = 0 \vee h(x_2) = 0,$

$g(f(x_1),b) + 1 \leq f(x_1),$

$h(c) = 1,$

$f(a) = 0$

F*

$h(c) = 1,$

$f(a) = 0,$

$g(f(a),b) + 1 \leq f(a),$

$g(f(a), b) = 0 \vee h(b) = 0$

"Domains":

$A_f : \{ a \}$

$A_g : \{ [f(a), b] \}$

$A_h : \{ c, b \}$

Microsoft Research

Example

F

$g(x_1, x_2) = 0 \vee h(x_2) = 0,$
$g(f(x_1),b) + 1 \leq f(x_1),$
$h(c) = 1,$
$f(a) = 0$

F*

$h(c) = 1,$
$f(a) = 0,$
$g(f(a),b) + 1 \leq f(a),$
$g(f(a), b) = 0 \vee h(b) = 0,$
$g(f(a), c) = 0 \vee h(c) = 0$

"Domains":

A_f : { a }
A_g : { [f(a), b], [f(a), c] }
A_h : { c, b }

Microsoft Research

Example

F

$g(x_1, x_2) = 0 \vee h(x_2) = 0,$

$g(f(x_1),b) + 1 \leq f(x_1),$

$h(c) = 1,$

$f(a) = 0$

F*

$h(c) = 1,$

$f(a) = 0,$

$g(f(a),b) + 1 \leq f(a),$

$g(f(a), b) = 0 \vee h(b) = 0,$

$g(f(a), c) = 0 \vee h(c) = 0$

M

$a \rightarrow 2,\ b \rightarrow 2,\ c \rightarrow 3$

$f \rightarrow \{\ 2 \rightarrow 0, \ldots\}$

$h \rightarrow \{\ 2 \rightarrow 0,\ 3 \rightarrow 1, \ldots\}$

$g \rightarrow \{\ [0,2] \rightarrow -1,\ [0,3] \rightarrow 0, \ldots\}$

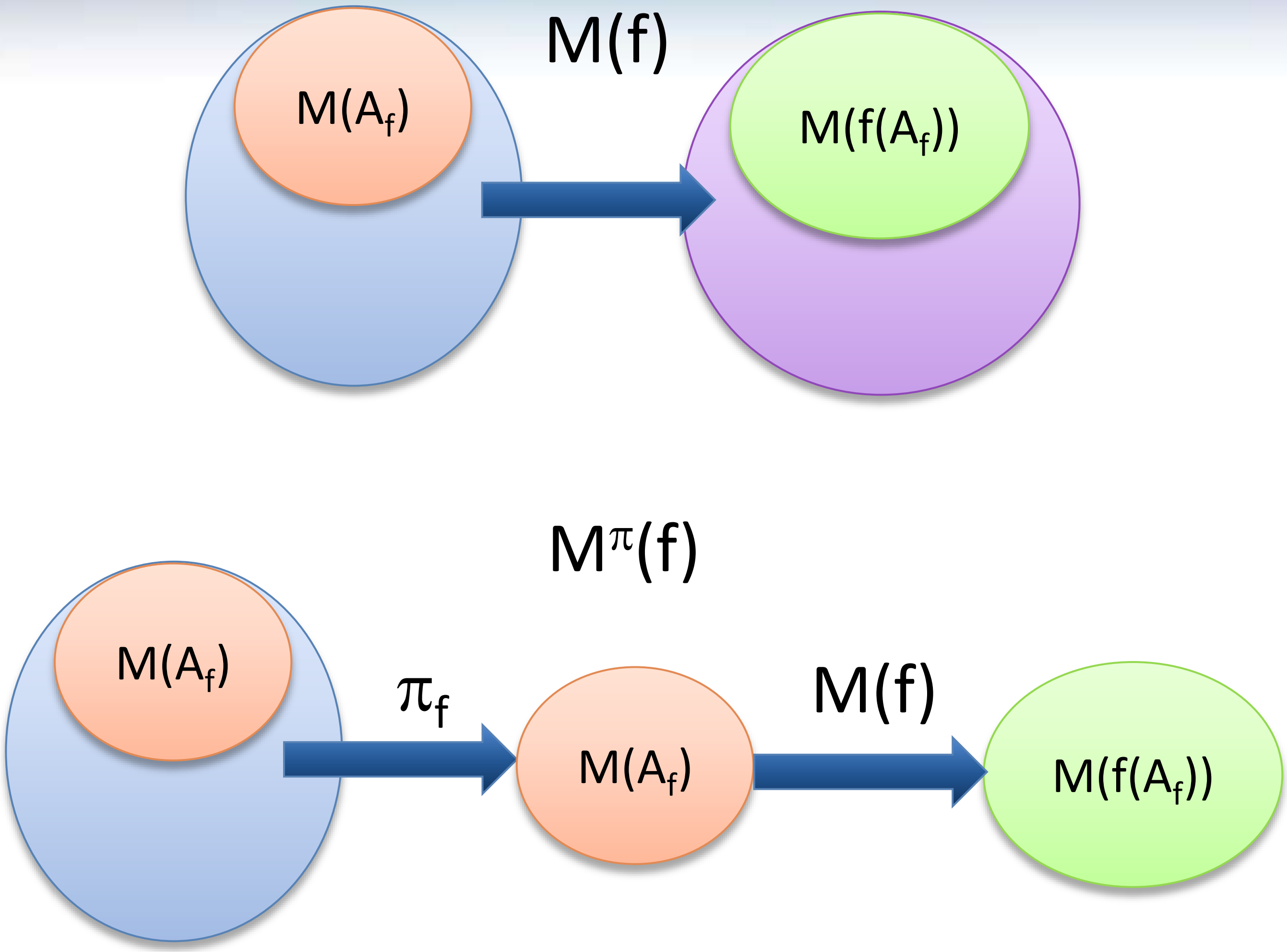
Basic Idea (cont.)

Given a model M for F^*,
Build a model M^π for F

Define a projection function π_f s.t.
range of π_f is $M(A_f)$, and
$\pi_f(v) = v \quad \text{if} \quad v \in M(A_f)$

Then,
$M^\pi(f)(v) = M(f)(\pi_f(v))$

Basic Idea (cont.)



Basic Idea (cont.)

Given a model M for F*,
Build a model M^π for F

In our example, we have: h(b) and h(c)
$\rightarrow A_h = \{ b, c \}$, and $M(A_h) = \{ 2, 3 \}$

$$\pi_h = \{ 2 \rightarrow 2, 3 \rightarrow 3, \text{else} \rightarrow 3 \}$$

M(h) $\Rightarrow$ M^π(h)

$\{ 2 \rightarrow 0, 3 \rightarrow 1, \ldots \}$ $\Rightarrow$ $\{ 2 \rightarrow 0, 3 \rightarrow 1, \text{else} \rightarrow 1 \}$

$$M^\pi(h) = \lambda x.\ \text{if}(x=2, 0, 1)$$

Microsoft Research

Example

F

$g(x_1, x_2) = 0 \vee h(x_2) = 0,$
$g(f(x_1),b) + 1 \leq f(x_1),$
$h(c) = 1,$
$f(a) = 0$

F*

$h(c) = 1,$
$f(a) = 0,$
$g(f(a),b) + 1 \leq f(a),$
$g(f(a), b) = 0 \vee h(b) = 0,$
$g(f(a), c) = 0 \vee h(c) = 0$

M

$a \rightarrow 2, b \rightarrow 2, c \rightarrow 3$
$f \rightarrow \{ 2 \rightarrow 0, \ldots\}$
$h \rightarrow \{ 2 \rightarrow 0, 3 \rightarrow 1, \ldots\}$
$g \rightarrow \{ [0,2]\rightarrow -1, [0,3]\rightarrow 0, \ldots\}$

M^{π}

$a \rightarrow 2, b \rightarrow 2, c \rightarrow 3$
$f \rightarrow \lambda x.\ 2$
$h \rightarrow \lambda x.\ if(x=2, 0, 1)$
$g \rightarrow \lambda x,y.\ if(x=0 \wedge y=2,-1, 0)$

Microsoft Research

Example: Model Checking

$\mathbf{M}^{\pi}$

$a \rightarrow 2,\ b \rightarrow 2,\ c \rightarrow 3$

$f \rightarrow \lambda x.\ 2$

$h \rightarrow \lambda x.\ \text{if}(x=2, 0, 1)$

$g \rightarrow \lambda x,y.\ \text{if}(x=0 \wedge y=2,-1, 0)$

Does M^{π} satisfies?

$\forall x_1, x_2 : g(x_1, x_2) = 0 \vee h(x_2) = 0$

$\forall x_1, x_2\text{: if}(x_1=0 \wedge x_2=2,-1,0) = 0 \vee \text{if}(x_2=2,0,1) = 0$ is valid

$\exists x_1, x_2\text{: if}(x_1=0 \wedge x_2=2,-1,0) \neq 0 \wedge \text{if}(x_2=2,0,1) \neq 0$ is unsat

$\text{if}(s_1=0 \wedge s_2=2,-1,0) \neq 0 \wedge \text{if}(s_2=2,0,1) \neq 0$ is unsat

Microsoft Research

Why does it work?

Suppose M^{π} does not satisfy $C[f(x)]$.

Then for some value v,
$M^{\pi}\{x \rightarrow v\}$ falsifies $C[f(x)]$.

$M^{\pi}\{x \rightarrow \pi_f(v)\}$ also falsifies $C[f(x)]$.

But, there is a term $t \in A_f$ s.t. $M(t) = \pi_f(v)$
Moreover, we instantiated $C[f(x)]$ with t.

So, M must not satisfy $C[f(t)]$.
Contradiction: M is a model for F^*.

Microsoft Research

Refinement 1: Lazy construction

- F^* may be very big (or infinite).
- Lazy-construction
  - Build F^* incrementally, F^* is the limit of the sequence

$$F^0 \subset F^1 \subset \ldots \subset F^k \subset \ldots$$

  - If F^k is unsat then F is unsat.
  - If F^k is sat, then build (candidate) M^π
  - If M^π satisfies all quantifiers in F then return sat.

Microsoft Research

Refinement 2: Model-based instantiation

Suppose M^π does not satisfy a clause $C[f(x)]$ in F.

Add an instance $C[f(t)]$ which "blocks" this spurious model.
Issue: how to find t?

Use model checking,
and the "inverse" mapping π_f^{-1} from values to terms (in A_f).
$\pi_f^{-1}(v) = t$ if $M^\pi(t) = \pi_f(v)$

Microsoft Research

Model-based instantiation: Example

F

$\forall x_1: f(x_1) < 0,$
$f(a) = 1,$
$f(b) = -1$

F^0

$f(a) = 1,$
$f(b) = -1$

M^π

$a \rightarrow 2,\ b \rightarrow 3$
$f \rightarrow \lambda x.\ \text{if}(x = 2, 1, -1)$

Model Checking $\forall x_1: f(x_1) < 0$

not $\text{if}(s_1 = 2, 1, -1) < 0$

$s_1 \rightarrow 2$
$\pi_f^{-1}(2) = a$

F^1

$f(a) = 1,$
$f(b) = -1$
$f(a) < 0$

unsat

Microsoft Research

Infinite F*

- Is our procedure refutationally complete?

- FOL Compactness

  A set of sentences is unsatisfiable

  iff

  it contains an unsatisfiable finite subset.

- A theory T is a set of sentences, then apply compactness to $F^* \cup T$

Microsoft Research

Infinite F*: Example

F

$\forall x_1$: $f(x_1) < f(f(x_1))$,
$\forall x_1$: $f(x_1) < a$,
$1 < f(0)$.

Unsatisfiable

F*

$f(0) < f(f(0))$, $f(f(0)) < f(f(f(0)))$, ...
$f(0) < a$, $f(f(0)) < a$, ...
$1 < f(0)$

Every finite subset of F* is satisfiable.

Microsoft Research

Infinite F*: What is wrong?

- Theory of linear arithmetic T_Z is the set of all first-order sentences that are true in the standard structure Z.
- T_z has non-standard models.
- F and F* are satisfiable in a non-standard model.

- Alternative: a theory is a class of structures.
- Compactness does not hold.
- F and F* are still equisatisfiable.

Δ_F and Set Constraints

Given a clause $C_k[x_1, \ldots, x_n]$

Let

- $S_{k,i}$ be the set of ground terms used to instantiate x_i in clause $C_k[x_1, \ldots, x_n]$

How to characterize $S_{k,i}$?

F **j-th argument of f in C_k**	**Δ_F** **system of set constraints**
a ground term t	$t \in A_{f,j}$
$t[x_1, \ldots, x_n]$	$t[S_{k,1}, \ldots, S_{k,n}] \subseteq A_{f,j}$
x_i	$S_{k,i} = A_{f,j}$

Microsoft Research

Δ_F: Example

F

$g(x_1, x_2) = 0 \vee h(x_2) = 0,$
$g(f(x_1),b) + 1 \leq f(x_1),$
$h(c) = 1,$
$f(a) = 0$

Δ_F

$S_{1,1} = A_{g,1}, S_{1,2} = A_{g,2}, S_{1,2} = A_{h,1}$
$S_{2,1} = A_{f,1}, f(S_{2,1}) \subseteq A_{g,1}, b \in A_{g,2}$
$c \in A_{h,1}$
$a \in A_{f,1}$

Δ_F: least solution

$S_{1,1} = \{ f(a) \},\ S_{1,2} = \{ b, c \}$
$S_{2,1} = \{ a \}$

Use Δ_F to generate F*

Microsoft Research

Complexity

- Δ_F is stratified then the least solution (and F*) is finite

$t[S_{k,1}, \ldots, S_{k,n}] \subseteq A_{f,j}$	$\mathrm{level}(S_{k,i}) < \mathrm{level}(A_{f,j})$
$S_{k,i} = A_{f,j}$	$\mathrm{level}(S_{k,i}) = \mathrm{level}(A_{f,j})$

- New decidable fragment: NEXPTIME-Hard.
- The least solution of Δ_F is exponential in the worst case.

  $a \in S_1$, $b \in S_1$, $f_1(S_1, S_1) \subseteq S_2$, …, $f_n(S_n, S_n) \subseteq S_{n+1}$
- F* can be doubly exponential in the size of F.

Microsoft Research

Extensions

- Arithmetical literals: π_f must be monotonic.

Literal of C_k	Δ_F
$\neg(x_i \leq x_j)$	$S_{k,i} = S_{k,j}$
$\neg(x_i \leq t), \neg(t \leq x_i)$	$t \in S_{k,i}$
$x_i = t$	$\{t+1, t-1\} \subseteq S_{k,i}$

- Offsets:

j-th argument of f in C_k	Δ_F
$x_i + r$	$S_{k,i}+r \subseteq A_{f,j}$ $A_{f,j}+(-r) \subseteq S_{k,i}$

Microsoft Research

Extensions: Example

Shifting

$$\neg(0 \leq x_1) \vee \neg(x_1 \leq n) \vee f(x_1) = g(x_1+2)$$

More Extensions

- Many-sorted logic
- Pseudo-Macros

$0 \leq g(x_1) \vee f(g(x_1)) = x_1,$
$0 \leq g(x_1) \vee h(g(x_1)) = 2x_1,$
$g(a) < 0$

Microsoft Research

Conclusion

Powerful, mature, and versatile tools like SMT solvers can now be exploited in very useful ways.

The construction and application of satisfiability procedures is an active research area with exciting challenges.

SMT is hot at Microsoft.

Z3 is a new SMT solver.

Main applications:

- Test-case generation.
- Verifying compiler.
- Model Checking & Predicate Abstraction.

Microsoft Research

Books

- Bradley & Manna: The Calculus of Computation
- Kroening & Strichman: Decision Procedures, An Algorithmic Point of View
- Chapter in the Handbook of Satisfiability

Microsoft Research

Web Links

Z3:

http://research.microsoft.com/projects/z3

http://research.microsoft.com/~leonardo

- Slides & Papers

http://www.smtlib.org

http://www.smtcomp.org

References

[Ack54] W. Ackermann. Solvable cases of the decision problem. *Studies in Logic and the Foundation of Mathematics*, 1954

[ABC$^+$02] G. Audemard, P. Bertoli, A. Cimatti, A. Kornilowicz, and R. Sebastiani. A SAT based approach for solving formulas over boolean and linear mathematical propositions. In *Proc. of CADE'02*, 2002

[BDS00] C. Barrett, D. Dill, and A. Stump. A framework for cooperating decision procedures. In *17th International Conference on Computer-Aided Deduction*, volume 1831 of *Lecture Notes in Artificial Intelligence*, pages 79–97. Springer-Verlag, 2000

[BdMS05] C. Barrett, L. de Moura, and A. Stump. SMT-COMP: Satisfiability Modulo Theories Competition. In *Int. Conference on Computer Aided Verification (CAV'05)*, pages 20–23. Springer, 2005

[BDS02] C. Barrett, D. Dill, and A. Stump. Checking satisfiability of first-order formulas by incremental translation to SAT. In Ed Brinksma and Kim Guldstrand Larsen, editors, *Proceedings of the 14^{th} International Conference on Computer Aided Verification (CAV '02)*, volume 2404 of *Lecture Notes in Computer Science*, pages 236–249. Springer-Verlag, July 2002. Copenhagen, Denmark

[BBC$^+$05] M. Bozzano, R. Bruttomesso, A. Cimatti, T. Junttila, P. van Rossum, S. Ranise, and R. Sebastiani. Efficient satisfiability modulo theories via delayed theory combination. In *Int. Conf. on Computer-Aided Verification (CAV)*, volume 3576 of *LNCS*. Springer, 2005

[Chv83] V. Chvatal. *Linear Programming*. W. H. Freeman, 1983

References

[CG96] B. Cherkassky and A. Goldberg. Negative-cycle detection algorithms. In *European Symposium on Algorithms*, pages 349–363, 1996

[DLL62] M. Davis, G. Logemann, and D. Loveland. A machine program for theorem proving. *Communications of the ACM*, 5(7):394–397, July 1962

[DNS03] D. Detlefs, G. Nelson, and J. B. Saxe. Simplify: A theorem prover for program checking. Technical Report HPL-2003-148, HP Labs, 2003

[DST80] P. J. Downey, R. Sethi, and R. E. Tarjan. Variations on the Common Subexpression Problem. *Journal of the Association for Computing Machinery*, 27(4):758–771, 1980

[dMR02] L. de Moura and H. Rueß. Lemmas on demand for satisfiability solvers. In *Proceedings of the Fifth International Symposium on the Theory and Applications of Satisfiability Testing (SAT 2002)*. Cincinnati, Ohio, 2002

[DdM06] B. Dutertre and L. de Moura. Integrating simplex with DPLL($\mathcal{T}$). Technical report, CSL, SRI International, 2006

[dMB07b] L. de Moura and N. Bjørner. Efficient E-Matching for SMT solvers. In *CADE-21*, pages 183–198, 2007

References

[dMB07c] L. de Moura and N. Bjørner. Model Based Theory Combination. In *SMT'07*, 2007

[dMB07a] L. de Moura and N. Bjørner. Relevancy Propagation . Technical Report MSR-TR-2007-140, Microsoft Research, 2007

[dMB08a] L. de Moura and N. Bjørner. Z3: An Efficient SMT Solver. In *TACAS 08*, 2008

[dMB08c] L. de Moura and N. Bjørner. Engineering DPLL(T) + Saturation. In *IJCAR'08*, 2008

[dMB08b] L. de Moura and N. Bjørner. Deciding Effectively Propositional Logic using DPLL and substitution sets. In *IJCAR'08*, 2008

[GHN+04] H. Ganzinger, G. Hagen, R. Nieuwenhuis, A. Oliveras, and C. Tinelli. DPLL(T): Fast decision procedures. In R. Alur and D. Peled, editors, *Int. Conference on Computer Aided Verification (CAV 04)*, volume 3114 of *LNCS*, pages 175–188. Springer, 2004

[MSS96] J. Marques-Silva and K. A. Sakallah. GRASP - A New Search Algorithm for Satisfiability. In *Proc. of ICCAD'96*, 1996

[NO79] G. Nelson and D. C. Oppen. Simplification by cooperating decision procedures. *ACM Transactions on Programming Languages and Systems*, 1(2):245–257, 1979

[NO05] R. Nieuwenhuis and A. Oliveras. DPLL(T) with exhaustive theory propagation and its application to difference logic. In *Int. Conference on Computer Aided Verification (CAV'05)*, pages 321–334. Springer, 2005

References

[Opp80] D. Oppen. Reasoning about recursively defined data structures. *J. ACM*, 27(3):403–411, 1980

[PRSS99] A. Pnueli, Y. Rodeh, O. Shtrichman, and M. Siegel. Deciding equality formulas by small domains instantiations. *Lecture Notes in Computer Science*, 1633:455–469, 1999

[Pug92] William Pugh. The Omega test: a fast and practical integer programming algorithm for dependence analysis. In *Communications of the ACM*, volume 8, pages 102–114, August 1992

[RT03] S. Ranise and C. Tinelli. The smt-lib format: An initial proposal. In *Proceedings of the 1st International Workshop on Pragmatics of Decision Procedures in Automated Reasoning (PDPAR'03), Miami, Florida*, pages 94–111, 2003

[RS01] H. Ruess and N. Shankar. Deconstructing shostak. In *16th Annual IEEE Symposium on Logic in Computer Science*, pages 19–28, June 2001

[SLB03] S. Seshia, S. Lahiri, and R. Bryant. A hybrid SAT-based decision procedure for separation logic with uninterpreted functions. In *Proc. 40th Design Automation Conference*, pages 425–430. ACM Press, 2003

[Sho81] R. Shostak. Deciding linear inequalities by computing loop residues. *Journal of the ACM*, 28(4):769–779, October 1981

References

[dMB09] L. de Moura and N. Bjørner. Generalized and Efficient Array Decision Procedures. FMCAD, 2009.

[GdM09] Y. Ge and L. de Moura. Complete Quantifier Instantiation for quantified SMT formulas, CAV, 2009.