Quantifiers

Bruno Dutertre
SRI International

Leonardo de Moura
Microsoft Research

Verification Tools need Quantifiers

Modeling the Runtime

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

Verification Tools need Quantifiers

Frame Axioms

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

Verification Tools need Quantifiers

User provided assertions

$$\forall\ i,j\!: i \leq j \Rightarrow \mathrm{read}(a,i) \leq \mathrm{read}(b,j)$$

Verification Tools need Quantifiers

Extra Theories

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

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

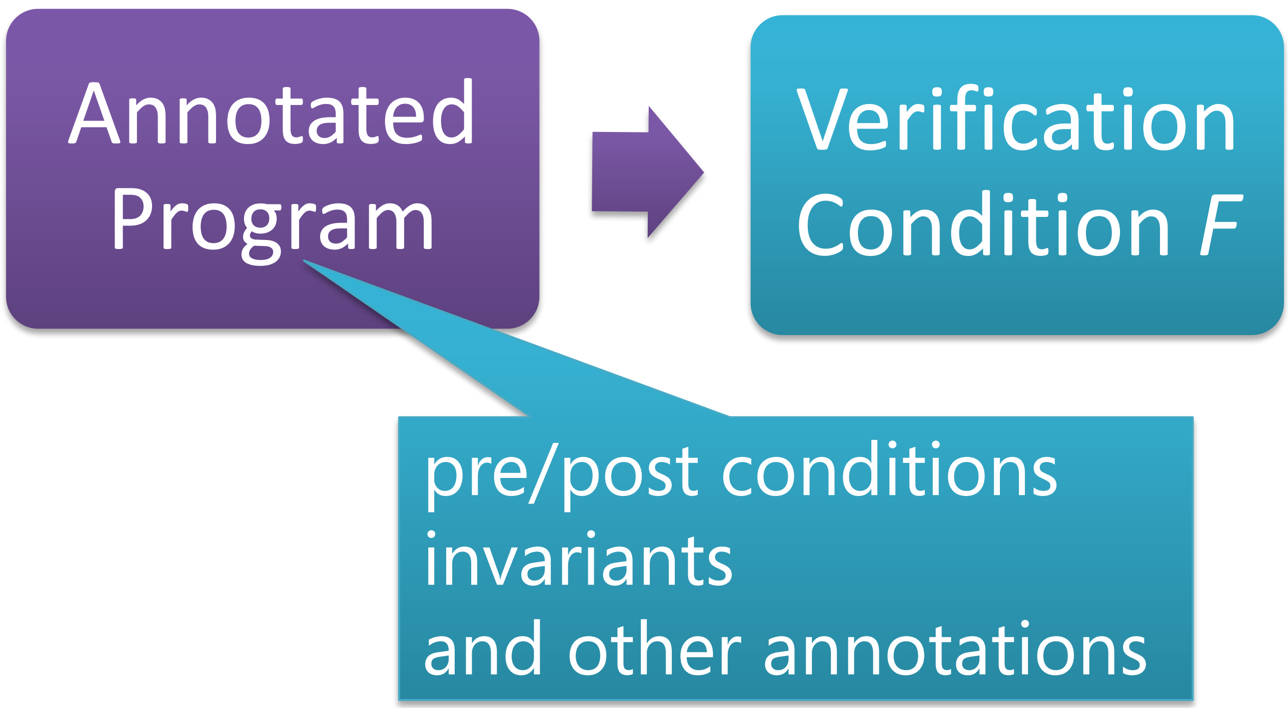
$\forall$ x,y: p(x,y), p(y,x) $\Rightarrow$ x = y

Verification Tools need Quantifiers

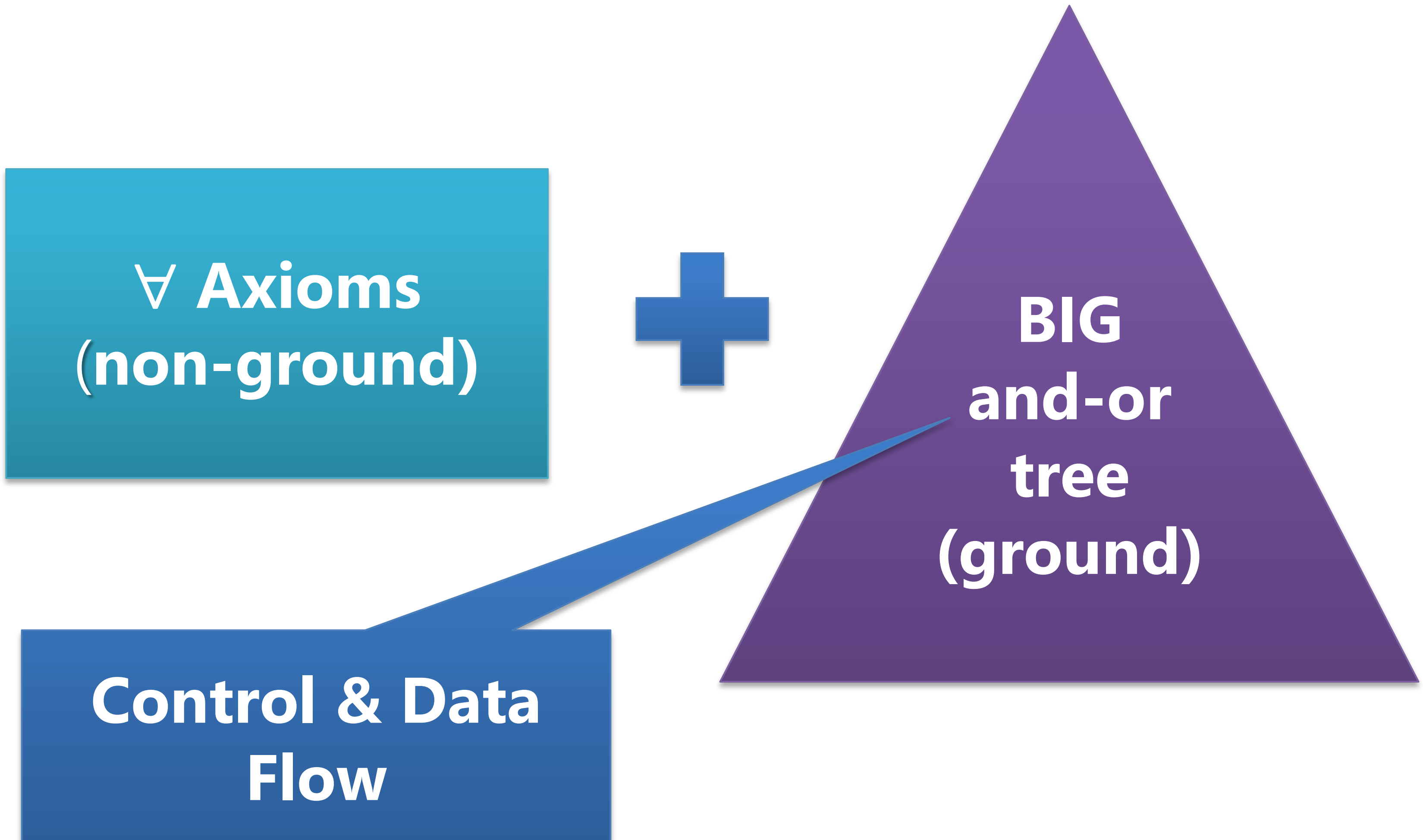
Main Challenge

Solver must be fast is satisfiable instances

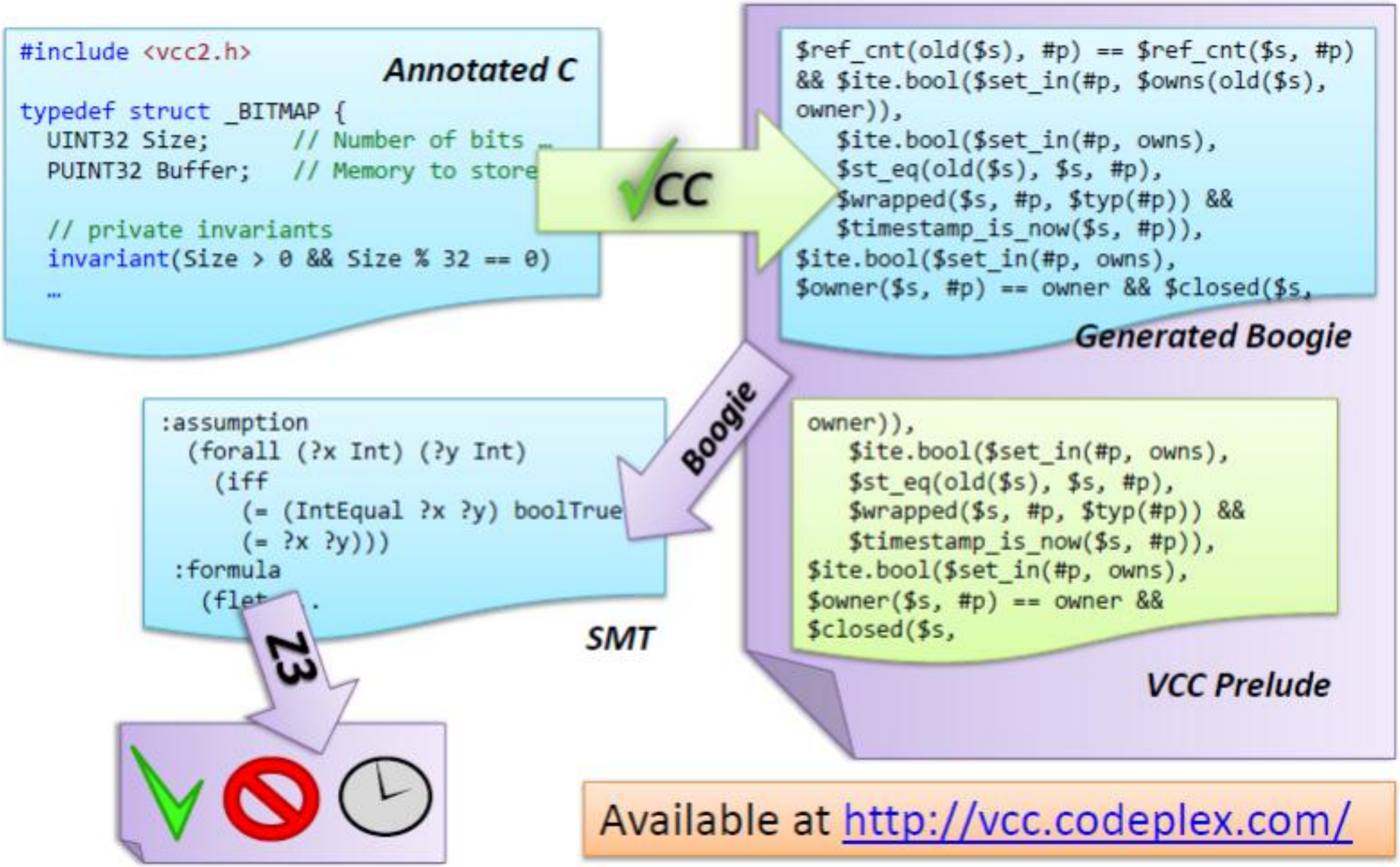
Verifying Compilers



Verification Condition: Structure



VCC: Verifying C Compiler



Available at http://vcc.codeplex.com/

BAD NEWS

First-order logic (FOL) is semi-decidable

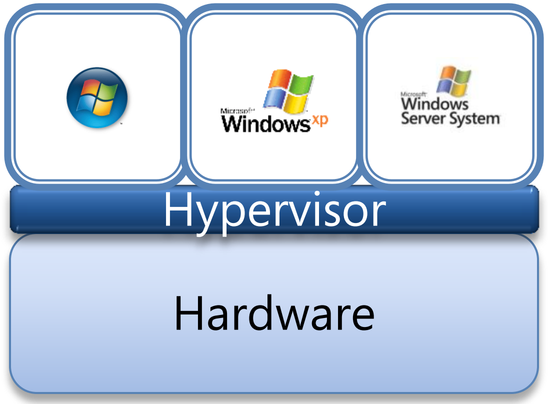
Quantifiers + EUF

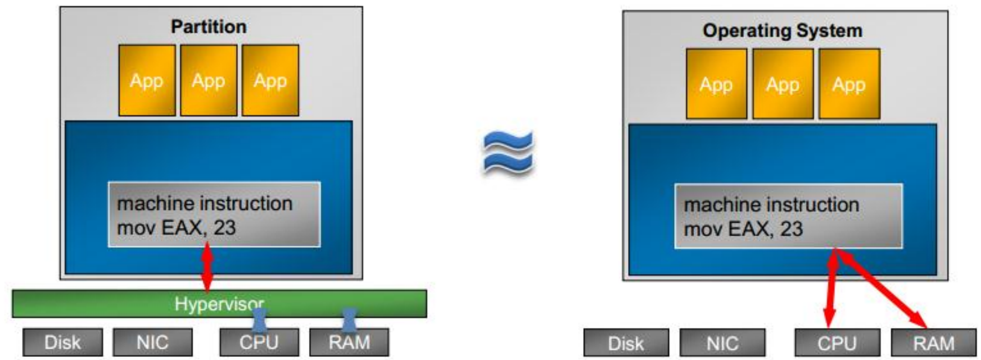
BAD NEWS

FOL + Linear Integer Arithmetic is undecidable

Quantifiers + EUF + LIA

Hypervisor





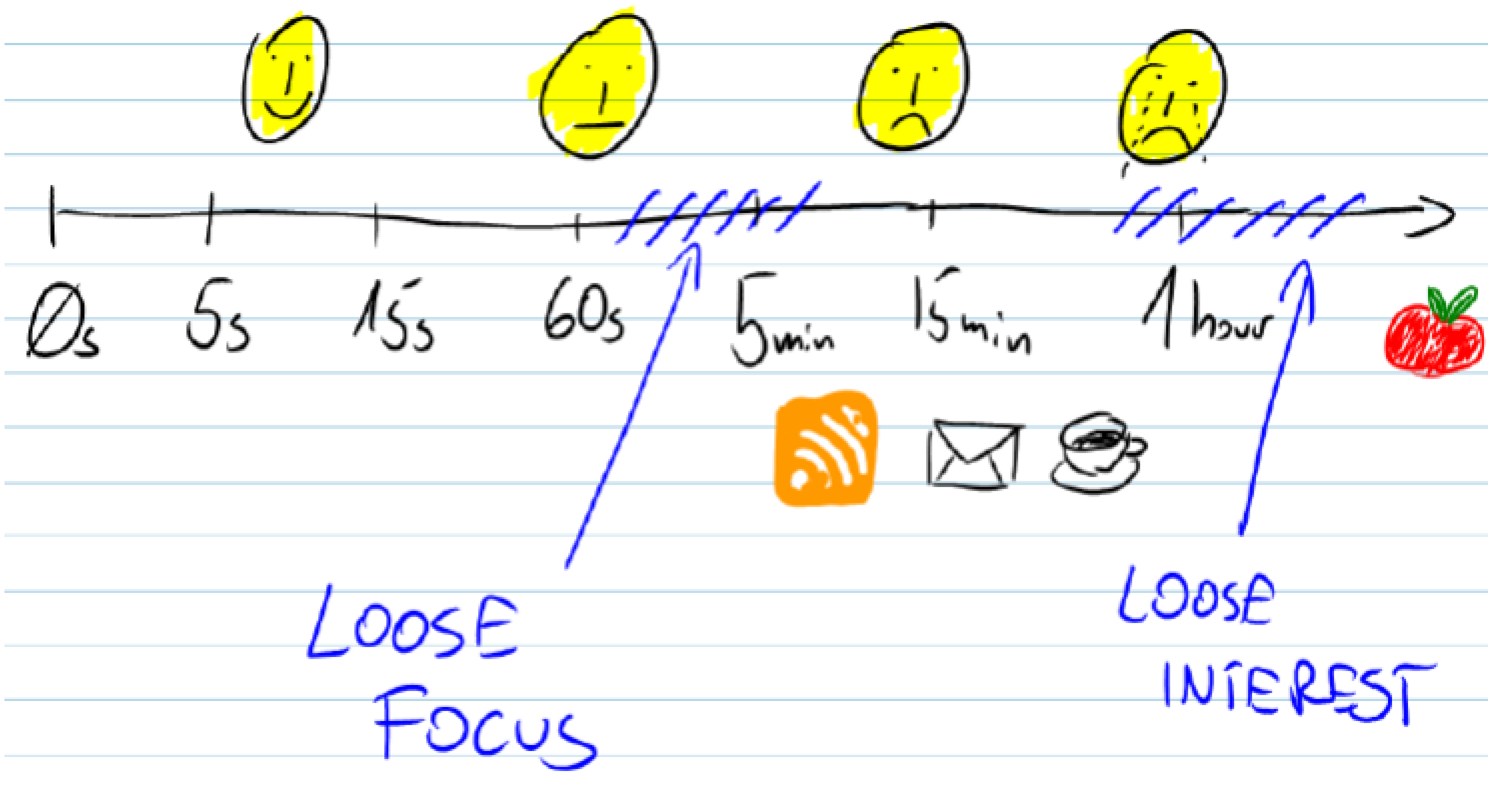
Challenges:

VCs have several Megabytes

Thousands universal quantifiers

Developers are willing at most 5 min per VC

Verification Attempt Time vs. Satisfaction and Productivity



By Michal Moskal (VCC Designer and Software Verification Expert)

NNF: Negation Normal Form

$$
\begin{aligned}
\mathit{NNF}(p) &= p \\
\mathit{NNF}(\neg p) &= \neg p \\
\mathit{NNF}(\neg\neg\phi) &= \mathit{NNF}(\phi) \\
\mathit{NNF}(\phi_0 \vee \phi_1) &= \mathit{NNF}(\phi_0) \vee \mathit{NNF}(\phi_1) \\
\mathit{NNF}(\neg(\phi_0 \vee \phi_1)) &= \mathit{NNF}(\neg\phi_0) \wedge \mathit{NNF}(\neg\phi_1) \\
\mathit{NNF}(\phi_0 \wedge \phi_1) &= \mathit{NNF}(\phi_0) \wedge \mathit{NNF}(\phi_1) \\
\mathit{NNF}(\neg(\phi_0 \wedge \phi_1)) &= \mathit{NNF}(\neg\phi_0) \vee \mathit{NNF}(\neg\phi_1) \\
\mathit{NNF}(\forall x : \phi) &= \forall x : \mathit{NNF}(\phi) \\
\mathit{NNF}(\neg(\forall x : \phi)) &= \exists x : \mathit{NNF}(\neg\phi) \\
\mathit{NNF}(\exists x : \phi) &= \exists x : \mathit{NNF}(\phi) \\
\mathit{NNF}(\neg(\exists x : \phi)) &= \forall x : \mathit{NNF}(\neg\phi)
\end{aligned}
$$

NNF: Negation Normal Form

Theorem: $F \Leftrightarrow \mathit{NNF}(F)$

Ex.: $\mathit{NNF}(\neg(p \land (\neg r \lor \forall x : q(x)))) = \neg p \lor (r \land \exists x : \neg q(x))$.

Skolemization

After NNF, Skolemization can be used to eliminate existential quantifiers.

$$\exists y : F[x, y] \;\leadsto\; F[x, f(x)]$$

Skolemization

The resultant formula is equisatisfiable.

Example:

$$\forall x : p(x) \Rightarrow \exists y : q(x, y)$$

$$\forall x : p(x) \Rightarrow q(x, f(x))$$

$\forall$ - Many Approaches

- Heuristic quantifier instantiation
- SMT + Saturation provers
- Complete quantifier instantiation
- Decidable fragments
- Model based quantifier instantiation
- Quantifier Elimination

Heuristic Quantifier Instantiation

E-matching (matching modulo equalities).

Example:

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

$a = g(b)$,

$b = c$,

$f(a) \neq c$

Pattern/Trigger

Heuristic Quantifier Instantiation

E-matching (matching modulo equalities).

Example:

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

a = g(b),

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

f(a) $\neq$ c

E-matching problem

Input: A set of ground equations E, a ground term t, and a pattern p, where p possibly contains variables.

Output: The set of substitutions β over the variables in p, such that:

$$E \models t = \beta(p)$$

Example:

$$\begin{aligned} E &\equiv \{a = f(b),\ a = f(c)\} \\ t &\equiv g(a) \\ p &\equiv g(f(x)) \\ R &\equiv \{\underbrace{\{x \mapsto b\}}_{\beta_1},\ \underbrace{\{x \mapsto c\}}_{\beta_2}\} \end{aligned}$$

Applying β_2: $a = f(b), a = f(c) \models g(a) = g(f(c))$

E-matching Challenge

Number of matches can be exponential

It is not refutationally complete

The real challenge is finding new matches:

- Incrementally during backtracking search
- Large database of patterns

EUF Solver: Review

$$f(g(a)) = c, c \neq f(g(b)), a = b$$

$$\begin{aligned}
F &= \{a \mapsto a,\ b \mapsto b,\ c \mapsto c,\ g(a) \mapsto g(a),\ g(b) \mapsto g(b) \\
&\quad f(g(a)) \mapsto f(g(a)),\ f(g(b)) \mapsto f(g(b))\} \\
D &= \{\} \\
\pi(a) &= \{g(a)\} \\
\pi(b) &= \{g(b)\} \\
\pi(g(a)) &= \{f(g(a))\} \\
\pi(g(b)) &= \{f(g(b))\}
\end{aligned}$$

EUF Solver: Review

$$f(g(a)) = c, c \neq f(g(b)), a = b$$

$$\begin{aligned}
F &= \{a \mapsto a,\ b \mapsto b,\ c \mapsto c,\ g(a) \mapsto g(a),\ g(b) \mapsto g(b) \\
&\quad f(g(a)) \mapsto f(g(a)),\ f(g(b)) \mapsto f(g(b))\} \\
D &= \{\} \\
\pi(a) &= \{g(a)\} \\
\pi(b) &= \{g(b)\} \\
\pi(g(a)) &= \{f(g(a))\} \\
\pi(g(b)) &= \{f(g(b))\}
\end{aligned}$$

Merge equivalence classes of $f(g(a))$ and c.

EUF Solver: Review

$$f(g(a)) = c, c \neq f(g(b)), a = b$$

$$\begin{aligned}
F &= \{a \mapsto a,\ b \mapsto b,\ c \mapsto c,\ g(a) \mapsto g(a),\ g(b) \mapsto g(b) \\
&\quad f(g(a)) \mapsto c,\ f(g(b)) \mapsto f(g(b))\} \\
D &= \{\} \\
\pi(a) &= \{g(a)\} \\
\pi(b) &= \{g(b)\} \\
\pi(g(a)) &= \{f(g(a))\} \\
\pi(g(b)) &= \{f(g(b))\}
\end{aligned}$$

EUF Solver: Review

$$f(g(a)) = c, c \neq f(g(b)), a = b$$

$$
\begin{aligned}
F &= \{a \mapsto a,\ b \mapsto b,\ c \mapsto c,\ g(a) \mapsto g(a),\ g(b) \mapsto g(b) \\
&\quad f(g(a)) \mapsto c,\ f(g(b)) \mapsto f(g(b))\} \\
D &= \{\} \\
\pi(a) &= \{g(a)\} \\
\pi(b) &= \{g(b)\} \\
\pi(g(a)) &= \{f(g(a))\} \\
\pi(g(b)) &= \{f(g(b))\}
\end{aligned}
$$

Add disequality

EUF Solver: Review

$$f(g(a)) = c, c \neq f(g(b)), a = b$$

$$
\begin{aligned}
F &= \{a \mapsto a,\ b \mapsto b,\ c \mapsto c,\ g(a) \mapsto g(a),\ g(b) \mapsto g(b) \\
&\quad\ f(g(a)) \mapsto c,\ f(g(b)) \mapsto f(g(b))\} \\
D &= \{c \neq f(g(b))\} \\
\pi(a) &= \{g(a)\} \\
\pi(b) &= \{g(b)\} \\
\pi(g(a)) &= \{f(g(a))\} \\
\pi(g(b)) &= \{f(g(b))\}
\end{aligned}
$$

EUF Solver: Review

$$f(g(a)) = c, c \neq f(g(b)), a = b$$

$$\begin{aligned}
F &= \{a \mapsto a,\ b \mapsto b,\ c \mapsto c,\ g(a) \mapsto g(a),\ g(b) \mapsto g(b) \\
&\quad f(g(a)) \mapsto c,\ f(g(b)) \mapsto f(g(b))\} \\
D &= \{c \neq f(g(b))\} \\
\pi(a) &= \{g(a)\} \\
\pi(b) &= \{g(b)\} \\
\pi(g(a)) &= \{f(g(a))\} \\
\pi(g(b)) &= \{f(g(b))\}
\end{aligned}$$

Merge equivalence classes of a and b.

EUF Solver: Review

$$f(g(a)) = c, c \neq f(g(b)), a = b, g(a) = g(b)$$

$$\begin{aligned}
F &= \{a \mapsto a,\ b \mapsto a,\ c \mapsto c,\ g(a) \mapsto g(a),\ g(b) \mapsto g(b) \\
&\quad\ f(g(a)) \mapsto c,\ f(g(b)) \mapsto f(g(b))\} \\
D &= \{c \neq f(g(b))\} \\
\pi(a) &= \{g(a), g(b)\} \\
\pi(b) &= \{g(b)\} \\
\pi(g(a)) &= \{f(g(a))\} \\
\pi(g(b)) &= \{f(g(b))\}
\end{aligned}$$

EUF Solver: Review

$$f(g(a)) = c, c \neq f(g(b)), a = b, g(a) = g(b)$$

$$\begin{aligned}
F &= \{a \mapsto a,\ b \mapsto a,\ c \mapsto c,\ g(a) \mapsto g(a),\ g(b) \mapsto g(b) \\
&\quad f(g(a)) \mapsto c,\ f(g(b)) \mapsto f(g(b))\} \\
D &= \{c \neq f(g(b))\} \\
\pi(a) &= \{g(a), g(b)\} \\
\pi(b) &= \{g(b)\} \\
\pi(g(a)) &= \{f(g(a))\} \\
\pi(g(b)) &= \{f(g(b))\}
\end{aligned}$$

Merge equivalence classes of $g(a)$ and $g(b)$.

EUF Solver: Review

$$f(g(a)) = c, c \neq f(g(b)), a = b, g(a) = g(b), f(g(a)) = f(g(b))$$

$$\begin{aligned}
F &= \{a \mapsto a,\ b \mapsto a,\ c \mapsto c,\ g(a) \mapsto g(b),\ g(b) \mapsto g(b) \\
&\quad f(g(a)) \mapsto c,\ f(g(b)) \mapsto f(g(b))\} \\
D &= \{c \neq f(g(b))\} \\
\pi(a) &= \{g(a), g(b)\} \\
\pi(b) &= \{g(b)\} \\
\pi(g(a)) &= \{f(g(a))\} \\
\pi(g(b)) &= \{f(g(b)), f(g(a))\}
\end{aligned}$$

EUF Solver: Review

$f(g(a)) = c, c \neq f(g(b)), a = b, g(a) = g(b), f(g(a)) = f(g(b))$

$$
\begin{array}{rcl}
F & = & \{a \mapsto a,\ b \mapsto a,\ c \mapsto c,\ g(a) \mapsto g(b),\ g(b) \mapsto g(b) \\
 & & f(g(a)) \mapsto c,\ f(g(b)) \mapsto f(g(b))\} \\
D & = & \{c \neq f(g(b))\} \\
\pi(a) & = & \{g(a), g(b)\} \\
\pi(b) & = & \{g(b)\} \\
\pi(g(a)) & = & \{f(g(a))\} \\
\pi(g(b)) & = & \{f(g(b)), f(g(a))\}
\end{array}
$$

Merge equivalence classes of $f(g(a))$ and $f(g(b))$ $\rightsquigarrow$ **unsat.**

E-matching

$$\begin{array}{rcl}
\textit{match}(x, t, S) & = & \{\beta \cup \{x \mapsto t\} \mid \beta \in S, x \notin \textit{dom}(\beta)\} \cup \\
& & \{\beta \mid \beta \in S, F^*(\beta(x)) = F^*(t)\} \\
\textit{match}(c, t, S) & = & S \;\; \textit{if} \;\; F^*(c) = F^*(t) \\
\textit{match}(c, t, S) & = & \emptyset \;\; \textit{if} \;\; F^*(c) \neq F^*(t) \\
\textit{match}(f(p_1, \ldots, p_n), t, S) & = &
\end{array}$$

$$\bigcup_{F^*(f(t_1,\ldots,t_n))=F^*(t)} \textit{match}(p_n, t_n, \ldots, \textit{match}(p_1, t_1, S) \ldots)$$

$\textit{match}(p, t, \{\emptyset\})$ returns the desired set of substitutions.

E-matching: Example

$$
\begin{aligned}
F \;\; = \;\; & \{a \mapsto c,\ b \mapsto b,\ c \mapsto c,\ d \mapsto d, \\
& f(c,b) \mapsto f(c,b),\ \ f(g(a),b) \mapsto f(c,b), \\
& g(a) \mapsto c,\ \ g(b) \mapsto g(b),\ \ g(c) \mapsto c,\ \ g(d) \mapsto c, \\
& h(a,d) \mapsto b,\ \ h(c,a) \mapsto b\}
\end{aligned}
$$

E-match t and p:

$$
\begin{aligned}
t \;\; &= \;\; f(c,b) \\
p \;\; &= \;\; f(g(x), h(x,a))
\end{aligned}
$$

E-matching: Example

$$
\begin{aligned}
F \quad = \quad & \{a \mapsto c,\ b \mapsto b,\ c \mapsto c,\ d \mapsto d, \\
& f(c,b) \mapsto f(c,b),\ \ f(g(a),b) \mapsto f(c,b), \\
& g(a) \mapsto c,\ \ g(b) \mapsto g(b),\ \ g(c) \mapsto c,\ \ g(d) \mapsto c, \\
& h(a,d) \mapsto b,\ \ h(c,a) \mapsto b\}
\end{aligned}
$$

$$\textit{match}(f(g(x), h(x,a)), f(c,b), \{\emptyset\}) =$$

E-matching: Example

$$
\begin{aligned}
F \;=\; \{ & a \mapsto c,\ b \mapsto b,\ c \mapsto c,\ d \mapsto d, \\
& f(c,b) \mapsto f(c,b),\ \ f(g(a),b) \mapsto f(c,b), \\
& g(a) \mapsto c,\ \ g(b) \mapsto g(b),\ \ g(c) \mapsto c,\ \ g(d) \mapsto c, \\
& h(a,d) \mapsto b,\ \ h(c,a) \mapsto b \}
\end{aligned}
$$

$\mathit{match}(f(g(x), h(x,a)), f(c,b), \{\emptyset\}) =$

$\mathit{match}(g(x), c, \mathit{match}(h(x,a), b, \{\emptyset\}))$ for $f(c,b)$

$\cup$

$\mathit{match}(g(x), g(a), \mathit{match}(h(x,a), b, \{\emptyset\}))$ for $f(g(a),b)$

E-matching: Example

$$
\begin{aligned}
F \;=\; \{ & a \mapsto c,\ b \mapsto b,\ c \mapsto c,\ d \mapsto d, \\
& f(c,b) \mapsto f(c,b),\ \ f(g(a),b) \mapsto f(c,b), \\
& g(a) \mapsto c,\ \ g(b) \mapsto g(b),\ \ g(c) \mapsto c,\ \ g(d) \mapsto c, \\
& h(a,d) \mapsto b,\ \ h(c,a) \mapsto b \}
\end{aligned}
$$

$\textit{match}(f(g(x),h(x,a)), f(c,b), \{\emptyset\}) =$

$\textit{match}(g(x), c, \textit{match}(x, a, \textit{match}(a, d, \{\emptyset\}))$ for $h(a,d)$

$\cup$

$\textit{match}(x, c, \textit{match}(a, a, \{\emptyset\})))$ for $h(c,a)$

$\cup$

$\textit{match}(g(x), g(a), \textit{match}(h(x,a), b, \{\emptyset\}))$

E-matching: Example

$$F = \{a \mapsto c,\ b \mapsto b,\ c \mapsto c,\ d \mapsto d,$$
$$f(c,b) \mapsto f(c,b),\ \ f(g(a),b) \mapsto f(c,b),$$
$$g(a) \mapsto c,\ \ g(b) \mapsto g(b),\ \ g(c) \mapsto c,\ \ g(d) \mapsto c,$$
$$h(a,d) \mapsto b,\ \ h(c,a) \mapsto b\}$$

$\mathit{match}(f(g(x), h(x,a)), f(c,b), \{\emptyset\}) =$

$\mathit{match}(g(x), c, \mathit{match}(x, a, \mathit{match}(a, d, \{\emptyset\})))$ for $h(a,d)$

$\cup$

$\mathit{match}(x, c, \mathit{match}(a, a, \{\emptyset\})))$ for $h(c,a)$

$\cup$

$\mathit{match}(g(x), g(a), \mathit{match}(h(x,a), b, \{\emptyset\}))$

a and d are not in the same equivalence class.

E-matching: Example

$$
\begin{aligned}
F \;=\; & \{a \mapsto c,\, b \mapsto b,\, c \mapsto c,\, d \mapsto d, \\
& f(c,b) \mapsto f(c,b),\;\; f(g(a),b) \mapsto f(c,b), \\
& g(a) \mapsto c,\;\; g(b) \mapsto g(b),\;\; g(c) \mapsto c,\;\; g(d) \mapsto c, \\
& h(a,d) \mapsto b,\;\; h(c,a) \mapsto b\}
\end{aligned}
$$

$$
\begin{aligned}
& \mathit{match}(f(g(x),h(x,a)), f(c,b), \{\emptyset\}) = \\
& \quad \mathit{match}(g(x), c, \mathit{match}(x, a, \emptyset) \\
& \qquad\qquad\qquad \cup \\
& \qquad\qquad\qquad \mathit{match}(x, c, \mathit{match}(a, a, \{\emptyset\}))) \\
& \quad \cup \\
& \quad \mathit{match}(g(x), g(a), \mathit{match}(h(x,a), b, \{\emptyset\}))
\end{aligned}
$$

E-matching: Example

$$
\begin{aligned}
F \;=\; & \{a \mapsto c,\ b \mapsto b,\ c \mapsto c,\ d \mapsto d, \\
& f(c,b) \mapsto f(c,b),\ \ f(g(a),b) \mapsto f(c,b), \\
& g(a) \mapsto c,\ \ g(b) \mapsto g(b),\ \ g(c) \mapsto c,\ \ g(d) \mapsto c, \\
& h(a,d) \mapsto b,\ \ h(c,a) \mapsto b\}
\end{aligned}
$$

$$
\begin{aligned}
& \mathit{match}(f(g(x), h(x,a)), f(c,b), \{\emptyset\}) = \\
& \quad \mathit{match}(g(x), c, \emptyset \\
& \qquad\qquad\qquad \cup \\
& \qquad\qquad\qquad \mathit{match}(x, c, \mathit{match}(a, a, \{\emptyset\}))) \\
& \quad \cup \\
& \quad \mathit{match}(g(x), g(a), \mathit{match}(h(x,a), b, \{\emptyset\}))
\end{aligned}
$$

E-matching: Example

$$F = \{a \mapsto c,\ b \mapsto b,\ c \mapsto c,\ d \mapsto d,$$

$$f(c,b) \mapsto f(c,b),\ \ f(g(a),b) \mapsto f(c,b),$$

$$g(a) \mapsto c,\ \ g(b) \mapsto g(b),\ \ g(c) \mapsto c,\ \ g(d) \mapsto c,$$

$$h(a,d) \mapsto b,\ \ h(c,a) \mapsto b\}$$

$$\mathit{match}(f(g(x), h(x,a)), f(c,b), \{\emptyset\}) =$$

$$\mathit{match}(g(x), c, \emptyset$$

$$\cup$$

$$\mathit{match}(x, c, \mathit{match}(a, a, \{\emptyset\})))$$

$$\cup$$

$$\mathit{match}(g(x), g(a), \mathit{match}(h(x,a), b, \{\emptyset\}))$$

E-matching: Example

$$\begin{aligned} F \;=\; & \{a \mapsto c,\ b \mapsto b,\ c \mapsto c,\ d \mapsto d, \\ & f(c,b) \mapsto f(c,b),\ \ f(g(a),b) \mapsto f(c,b), \\ & g(a) \mapsto c,\ \ g(b) \mapsto g(b),\ \ g(c) \mapsto c,\ \ g(d) \mapsto c, \\ & h(a,d) \mapsto b,\ \ h(c,a) \mapsto b\} \end{aligned}$$

$$\mathit{match}(f(g(x), h(x,a)), f(c,b), \{\emptyset\}) =$$

$$\mathit{match}(g(x), c, \emptyset$$

$$\cup$$

$$\mathit{match}(x, c, \color{red}{\mathit{match}(a, a, \{\emptyset\})}))$$

$$\cup$$

$$\mathit{match}(g(x), g(a), \mathit{match}(h(x,a), b, \{\emptyset\}))$$

$$\color{red}{F^*(a) = F^*(a)}$$

E-matching: Example

$$\begin{aligned}F \;=\; & \{a \mapsto c,\ b \mapsto b,\ c \mapsto c,\ d \mapsto d,\\ & f(c,b) \mapsto f(c,b),\ \ f(g(a),b) \mapsto f(c,b),\\ & g(a) \mapsto c,\ \ g(b) \mapsto g(b),\ \ g(c) \mapsto c,\ \ g(d) \mapsto c,\\ & h(a,d) \mapsto b,\ \ h(c,a) \mapsto b\}\end{aligned}$$

$$\mathit{match}(f(g(x),h(x,a)), f(c,b), \{\emptyset\}) =$$

$$\mathit{match}(g(x), c, \emptyset$$

$$\cup$$

$$\color{red}{\mathit{match}(x, c, \{\emptyset\}))}$$

$$\cup$$

$$\mathit{match}(g(x), g(a), \mathit{match}(h(x,a), b, \{\emptyset\}))$$

E-matching: Example

$$
\begin{aligned}
F \;=\; & \{a \mapsto c,\ b \mapsto b,\ c \mapsto c,\ d \mapsto d, \\
& f(c,b) \mapsto f(c,b),\ \ f(g(a),b) \mapsto f(c,b), \\
& g(a) \mapsto c,\ \ g(b) \mapsto g(b),\ \ g(c) \mapsto c,\ \ g(d) \mapsto c, \\
& h(a,d) \mapsto b,\ \ h(c,a) \mapsto b\}
\end{aligned}
$$

$$
\begin{aligned}
& \mathit{match}(f(g(x),h(x,a)), f(c,b), \{\emptyset\}) = \\
& \quad \mathit{match}(g(x), c, \emptyset \\
& \qquad\qquad\qquad \cup \\
& \qquad\qquad\qquad \{\{x \mapsto c\}\}) \\
& \quad \cup \\
& \quad \mathit{match}(g(x), g(a), \mathit{match}(h(x,a), b, \{\emptyset\}))
\end{aligned}
$$

E-matching: Example

$$
\begin{aligned}
F \;=\; & \{a \mapsto c,\ b \mapsto b,\ c \mapsto c,\ d \mapsto d, \\
& f(c,b) \mapsto f(c,b),\ \ f(g(a),b) \mapsto f(c,b), \\
& g(a) \mapsto c,\ \ g(b) \mapsto g(b),\ \ g(c) \mapsto c,\ \ g(d) \mapsto c, \\
& h(a,d) \mapsto b,\ \ h(c,a) \mapsto b\}
\end{aligned}
$$

$$
\begin{aligned}
& \mathit{match}(f(g(x),h(x,a)), f(c,b), \{\emptyset\}) = \\
& \quad \mathit{match}(g(x), c, \{\{x \mapsto c\}\}) \\
& \quad \cup \\
& \quad \mathit{match}(g(x), g(a), \mathit{match}(h(x,a), b, \{\emptyset\}))
\end{aligned}
$$

E-matching: Example

$$
\begin{aligned}
F \;=\; \{ & a \mapsto c,\ b \mapsto b,\ c \mapsto c,\ d \mapsto d, \\
& f(c,b) \mapsto f(c,b),\ \ f(g(a),b) \mapsto f(c,b), \\
& g(a) \mapsto c,\ \ g(b) \mapsto g(b),\ \ g(c) \mapsto c,\ \ g(d) \mapsto c, \\
& h(a,d) \mapsto b,\ \ h(c,a) \mapsto b \}
\end{aligned}
$$

$\mathit{match}(f(g(x), h(x,a)), f(c,b), \{\emptyset\}) =$

$\mathit{match}(x, a, \{\{x \mapsto c\}\}) \cup$ for $g(a)$

$\mathit{match}(x, c, \{\{x \mapsto c\}\}) \cup$ for $g(c)$

$\mathit{match}(x, d, \{\{x \mapsto c\}\}) \cup$ for $g(d)$

$\mathit{match}(g(x), g(a), \mathit{match}(h(x,a), b, \{\emptyset\}))$

E-matching: Example

$$
\begin{aligned}
F \;=\; & \{a \mapsto c,\ b \mapsto b,\ c \mapsto c,\ d \mapsto d, \\
& f(c,b) \mapsto f(c,b),\ \ f(g(a),b) \mapsto f(c,b), \\
& g(a) \mapsto c,\ \ g(b) \mapsto g(b),\ \ g(c) \mapsto c,\ \ g(d) \mapsto c, \\
& h(a,d) \mapsto b,\ \ h(c,a) \mapsto b\}
\end{aligned}
$$

$$
\begin{aligned}
& \mathit{match}(f(g(x), h(x,a)), f(c,b), \{\emptyset\}) = \\
& \quad \{\{x \mapsto c\}\} \cup \\
& \quad \{\{x \mapsto c\}\} \cup \\
& \quad \emptyset \cup \\
& \quad \mathit{match}(g(x), g(a), \mathit{match}(h(x,a), b, \{\emptyset\}))
\end{aligned}
$$

E-matching: Example

$$\begin{aligned} F \;=\; & \{a \mapsto c,\ b \mapsto b,\ c \mapsto c,\ d \mapsto d, \\ & f(c,b) \mapsto f(c,b),\ \ f(g(a),b) \mapsto f(c,b), \\ & g(a) \mapsto c,\ \ g(b) \mapsto g(b),\ \ g(c) \mapsto c,\ \ g(d) \mapsto c, \\ & h(a,d) \mapsto b,\ \ h(c,a) \mapsto b\} \end{aligned}$$

$$\mathit{match}(f(g(x),h(x,a)),f(c,b),\{\emptyset\}) =$$

$$\{\{x \mapsto c\}\} \cup$$

$$\mathit{match}(g(x),g(a),\mathit{match}(h(x,a),b,\{\emptyset\}))$$

E-matching: Example

$$
\begin{aligned}
F \;=\; & \{a \mapsto c,\ b \mapsto b,\ c \mapsto c,\ d \mapsto d, \\
& f(c,b) \mapsto f(c,b),\ \ f(g(a),b) \mapsto f(c,b), \\
& g(a) \mapsto c,\ \ g(b) \mapsto g(b),\ \ g(c) \mapsto c,\ \ g(d) \mapsto c, \\
& h(a,d) \mapsto b,\ \ h(c,a) \mapsto b\}
\end{aligned}
$$

$$
\mathit{match}(f(g(x), h(x,a)), f(c,b), \{\emptyset\}) =
$$

$$
\{\{x \mapsto c\}\} \cup
$$

$$
\{\{x \mapsto c\}\}
$$

E-matching: Example

$$\begin{aligned} F \quad = \quad & \{a \mapsto c,\ b \mapsto b,\ c \mapsto c,\ d \mapsto d, \\ & f(c,b) \mapsto f(c,b),\ \ f(g(a),b) \mapsto f(c,b), \\ & g(a) \mapsto c,\ \ g(b) \mapsto g(b),\ \ g(c) \mapsto c,\ \ g(d) \mapsto c, \\ & h(a,d) \mapsto b,\ \ h(c,a) \mapsto b\} \end{aligned}$$

$$\mathit{match}(f(g(x), h(x,a)), f(c,b), \{\emptyset\}) =$$

$$\{\{x \mapsto c\}\}$$

Efficient E-matching

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

Bad user provided triggers:

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

$g(a) = c,$

$g(b) = c,$

$a \neq b$

Trigger is too restrictive

E-matching limitations

Bad user provided triggers:

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

$g(a) = c,$

$g(b) = c,$

$a \neq b$

More "liberal" trigger

E-matching limitations

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 $\Longrightarrow$ a=b

E-matching limitations

It is not refutationally complete

↓

False positives

E-matching: why do we use it?

Integrates smoothly with current SMT Solvers design.

Proof finding.

Software verification problems are big & shallow.

Decidable Fragments
&
Complete Quantifier Instatiation

$\forall$ + theories

There is no sound and refutationally complete procedure for
linear arithmetic + unintepreted function symbols

Model Generation

How to represent the model of satisfiable formulas?

Functor:

Given a model *M* for *T*

Generate a model *M'* for *F* (modulo *T*)

Example:

F: f(a) = 0 and a > b and f(b) > f(a) + 1

M':

Symbol	Interpretation
a	1
b	0
f	ite(x=1, 0, 2)

Model Generation

How to represent the model of satisfiable formulas?

Functor:

Given a model *M* for *T*

Generate a model *M'* for *F* (modu

Interpretation is given using *T*-symbols

Example:

F: f(a) = 0 and a > b and f(b) > f(a) + 1

M':

Symbol	Interpretation
a	1
b	0
f	ite(x=1, 0, 2)

Model Generation

How to represent the model of satisfiable formulas?

Functor:

Given a model *M* for *T*

Generate a model *M'* for *F* (modu

Non ground term (lambda expression)

Example:

F: f(a) = 0 and a > b and f(b) > f(a) + 1

M':

Symbol	Interpretati
a	1
b	0
f	ite(x=1, 0, 2)

Models as Functional Programs

```
(declare-fun f (Int Int) Int)
(declare-const a Int)
(declare-const b Int)

(assert (forall ((x Int)) (>= (f x x) (+ x a))))

(assert (< (f a b) a))
(assert (> a 0))
(check-sat)
(get-model)

(echo "evaluating (f (+ a 10) 20)...")
(eval (f (+ a 10) 20))
```

ask z3

```
sat
(model
  (define-fun b () Int
    2)
  (define-fun a () Int
    1)
  (define-fun f ((x!1 Int) (x!2 Int)) Int
    (ite (and (= x!1 1) (= x!2 2)) 0
      (+ 1 x!1)))
)
evaluating (f (+ a 10) 20)...
12
```

Model Checking

M':

Symbol	Interpretation
a	1
b	0
f	ite(x=1, 0, 2)

Is ∀x: f(x) ≥ 0 satisfied by *M'?*

Yes,
not (ite(k=1,0,2) ≥ 0) is unsatisfiable

Model Checking

M':

Symbol	Interpretation
a	1
b	0
f	ite(x=1, 0, 2)

Is $\forall x$: f(x) ≥ 0 satisfied by *M'*?

Yes,
not (ite(k=1,0,2) ≥ 0) is unsatisfiable

Negated quantifier
Replaced *f* by its interpretation
Replaced *x* by fresh constant *k*

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***

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

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

$h(c) = 1,$

$f(a) = 0$

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

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

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

$h(c) = 1,$

$f(a) = 0$

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*

$h(c) = 1,$

$f(a) = 0,$

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

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

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”:

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

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

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

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

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

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

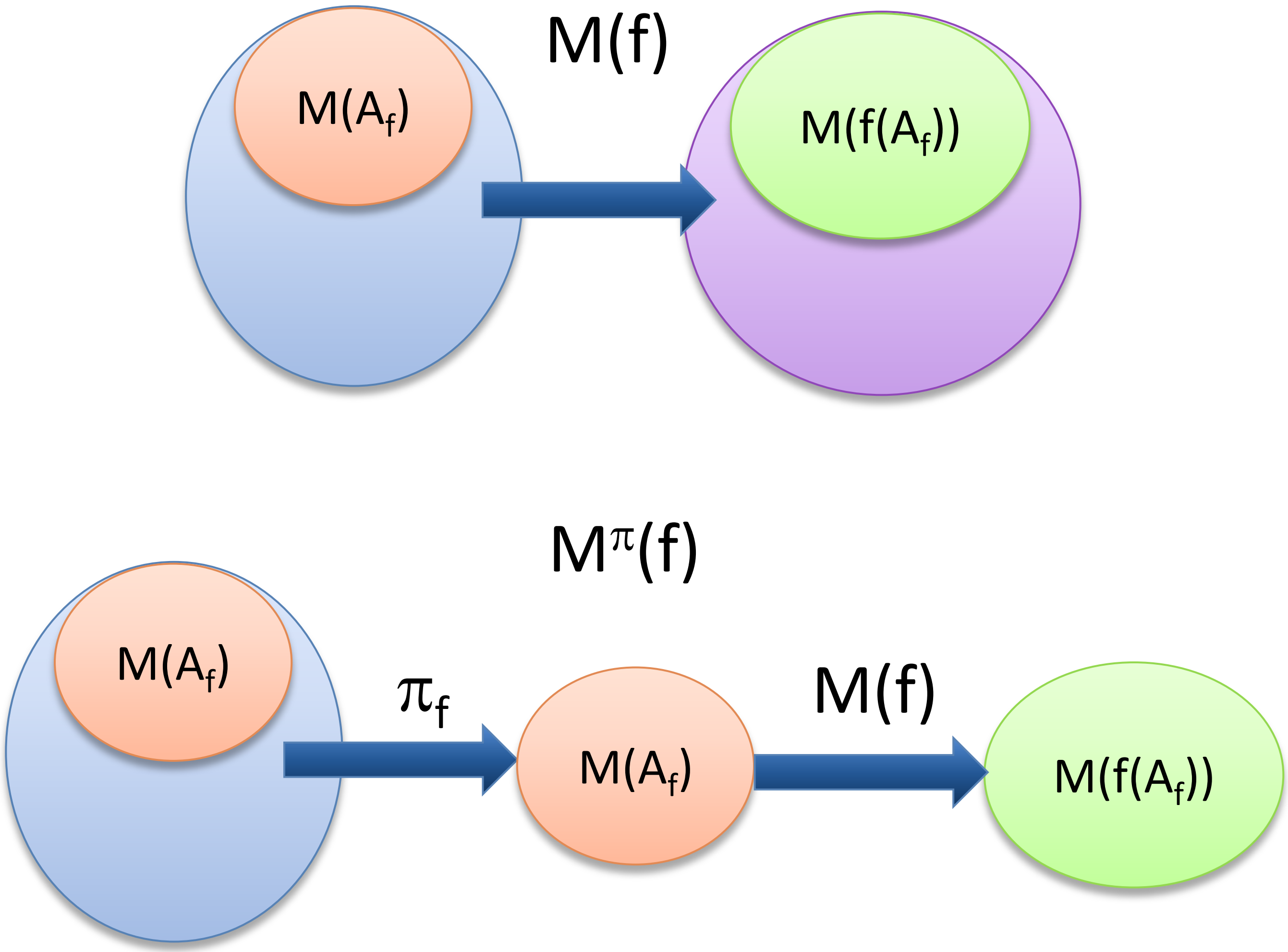
Basic Idea

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
$\pi_f(v) = v$ if $v \in M(A_f)$

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

Basic Idea



Basic Idea

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 $M(A_h) = \{ 2, 3 \}$

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

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

$\Rightarrow$

$M^\pi(h)$
$\{ 2 \rightarrow 0, 3 \rightarrow 1, \text{else} \rightarrow 1 \}$

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

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

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

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

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

Example : Model Checking

M$^{\pi}$

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

$f \to \lambda x.\ 2$

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

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

Does M$^{\pi}$ satisfies?

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

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

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

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

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^*.

Refinement: 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.

Refinement: 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 $M^\pi(t) = \pi_f(v)$

Example: Model-based instantiation

F

$\forall x_1: f(x_1) < 0,$

$f(a) = 1,$

$f(b) = -1$

F^0

$f(a) = 1,$

$f(b) = -1$

M^π

$a \rightarrow 2, b \rightarrow 3$

$f \rightarrow \lambda x.\ if(x = 2, 1, -1)$

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

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

$s_1 \rightarrow 2$

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

F^1

$f(a) = 1,$

$f(b) = -1$

$f(a) < 0$

unsat

Infinite F*

Is 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 $F^* \cup T$

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.

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.

Extensions

Shifting

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

Extensions

Many-sorted logic
Pseudo-Macros

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

Extensions

Online tutorial at:
http://rise4fun.com/z3/tutorial

Extensions

Online tutorial at:
http://rise4fun.com/z3/tutorial

Related work

Bernays-Schönfinkel class.

Stratified Many-Sorted Logic.

Array Property Fragment.

Local theory extensions.

SMT + Saturation

CDCL/DPLL : Review

M | F

Partial model

Set of clauses

CDCL/DPLL : Review

Guessing

$p \;|\; p \vee q, \neg q \vee r$

$p, \neg q \;|\; p \vee q, \neg q \vee r$

CDCL/DPLL : Review

Deducing

$p \;|\; p \vee q, \neg p \vee s$

$p, s \,|\, p \vee q, \neg p \vee s$

CDCL/DPLL : Review

Backtracking

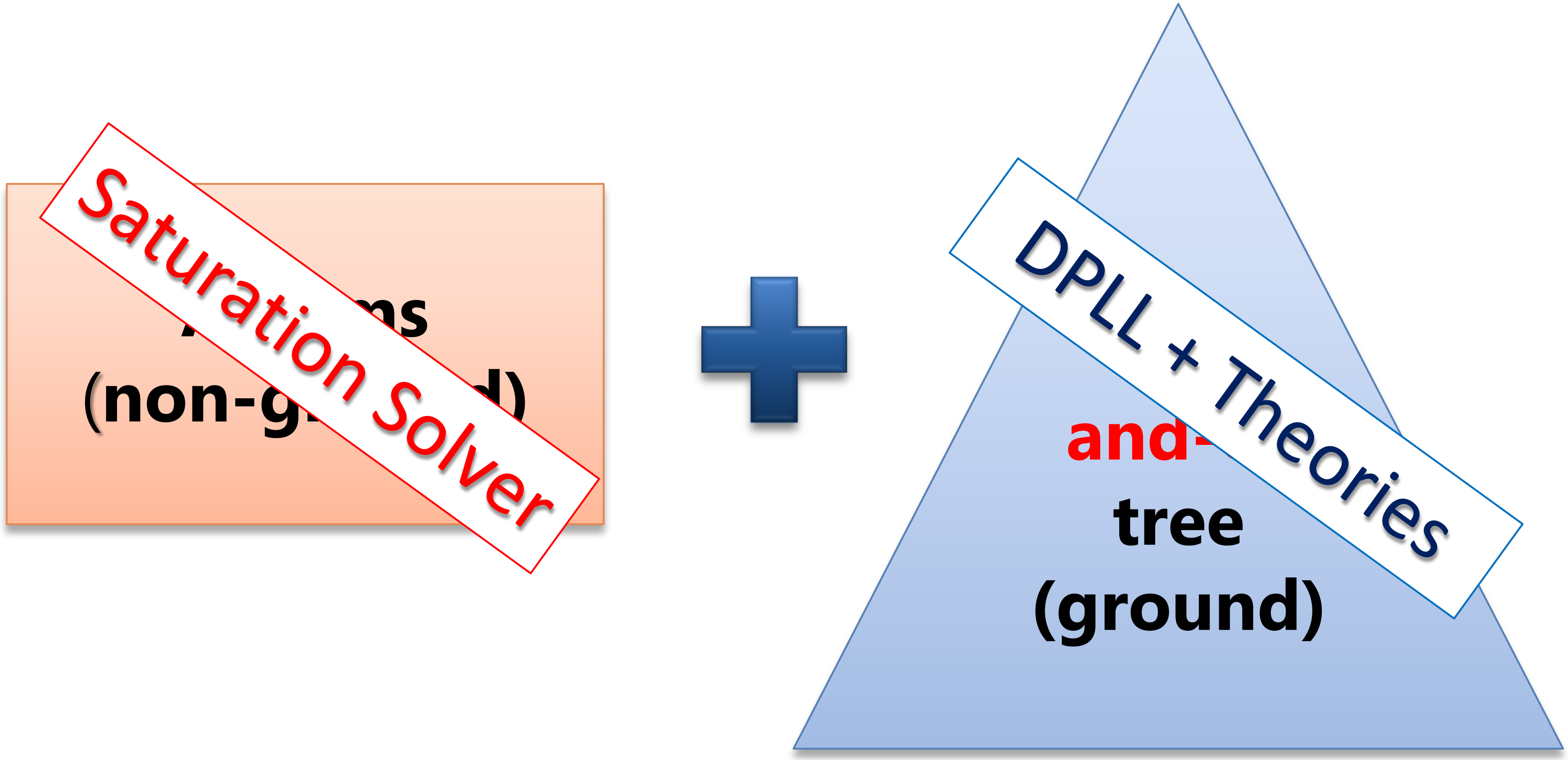
$p, \neg s, q \mid p \vee q, s \vee q, \neg p \vee \neg q$

$\Downarrow$

$p, s \mid p \vee q, s \vee q, \neg p \vee \neg q$

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

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

DPLL(Γ) : Deduce I

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

DPLL(Γ) : Deduce I

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

Resolution

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

DPLL(Γ) : Deduce II

Using ground atoms from M:

M | F

Main issue: backtracking.

Hypothetical clauses:

$H \rhd C$

Track literals from M used to derive C

(hypothesis) Ground literals

(regular) Clause

DPLL(Γ) : Deduce II

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

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

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

DPLL(Γ) : Backtracking

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

DPLL(Γ) : Backtracking

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

p(a) is removed from M

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

DPLL(Γ) : Improvement

Saturation solver ignores non-unit ground clauses.

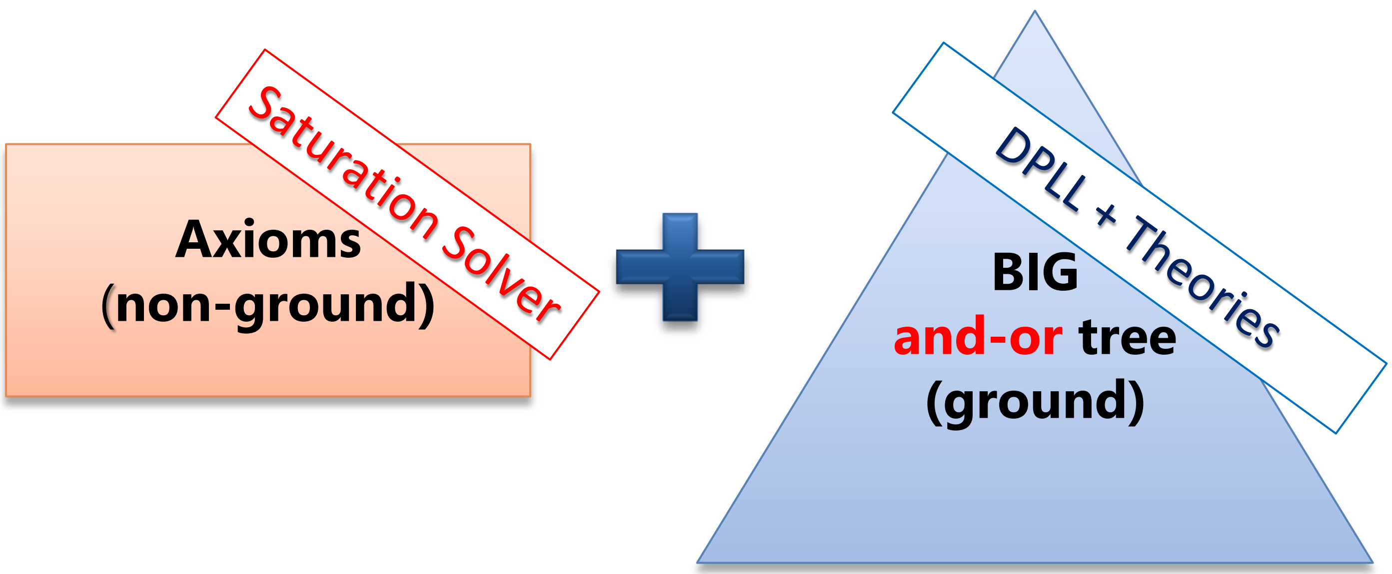
$$p(a) \mid \cancel{p(a) \lor q(a)}, \neg p(x) \lor r(x)$$

DPLL(Γ) : Improvement

Saturation solver ignores non-unit ground clauses.

It is still refutanionally complete if:

Γ has the reduction property.

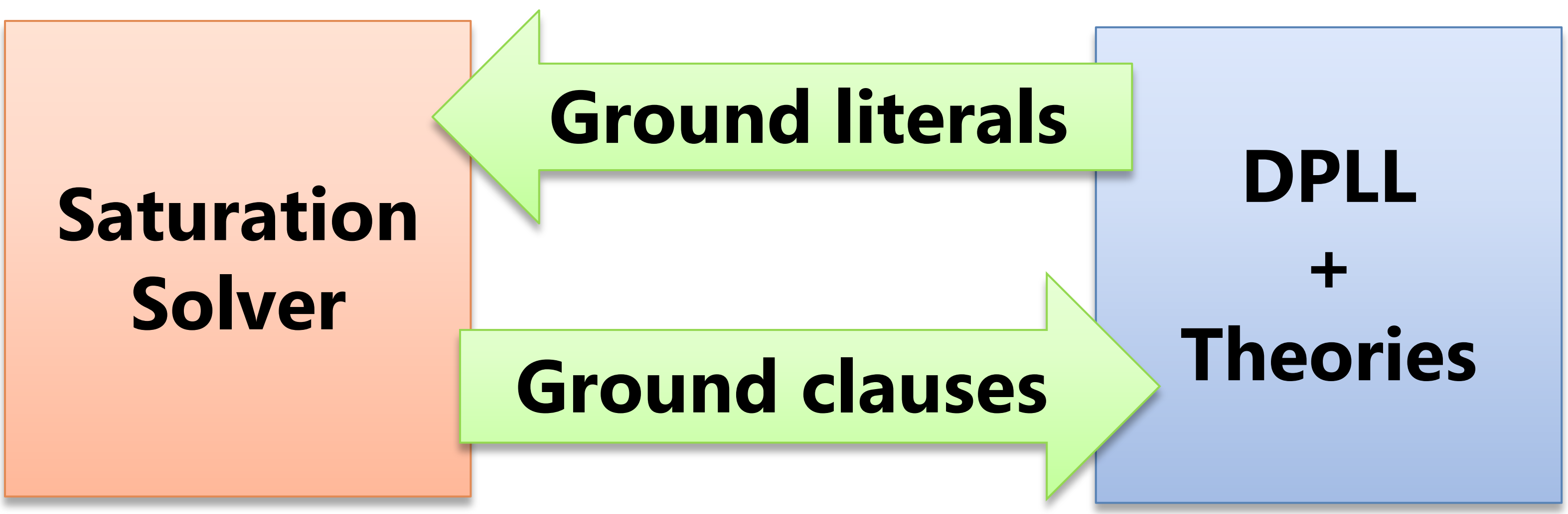


DPLL(Γ) : Improvement

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

Summary

E-matching

- proof finding
- fast
- shallow proofs in big formulas
- not refutationally complete
- regularly solves VCs with more than 5 Mb

Summary

Complete instantiation + MBQI

- decides several useful fragments
- model & proof finding
- slow
- complements E-matching

Summary

SMT + Saturation

- refutationally complete for pure first-order
- proof finding
- slow

Not covered

Quantifier elimination

- Fourier-Motzkin (Linear Real Arithmetic)
- Cooper (Linear Integer Arithmetic)
- CAD (Nonlinear Real Arithmetic)
- Algebraic Datatypes (Hodges)

Finite model finding

Many Decidable Fragments

Challenge

New and efficient procedures capable of producing models for satisfiable instances.