SMT Solver

Zhaoguo Wang

Adapted From:

ETH Zurich program verification, Lecture 2, 3
(https://www.pm.inf.ethz.ch/education/courses/program-verification.html)
https://theory.stanford.edu/~nikolaj/programmingz3.html
<<Solving SAT and SAT Modulo Theories: From an Abstract Davis–Putnam–Logemann–Loveland Procedure to DPLL(T)>>

Predicate Logic（谓词逻辑）

1.1 Propositional Logic

Step 0. Proof.

Step 1. Convert program into mathematical formula.

Step 2. Ask the computer to solve the formula.

1.2 First Order Logic

Step 1. Convert it into first order logic formula.

Step 2. Ask the computer to solve the formula.

1.3 Auto-active Proof

Step 1. Axiom system

Step 2. Ask the computer to check the invariants

Outline – This Lecture

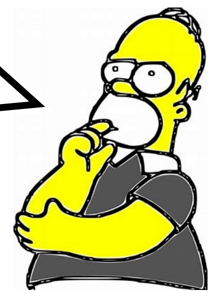
- SMT Solver
  - From DPLL to DPLL(T)
  - EUF

SMT

DEFINITION

The SMT problem (abbreviated satisfiability modulo theories) is the problem of determining if there exists an interpretation that satisfies a given predicate logic formula.

We can use SAT solvers to solve SAT problems automatically. Can we solve SMT problems automatically?



MiniSmt Computational Logic

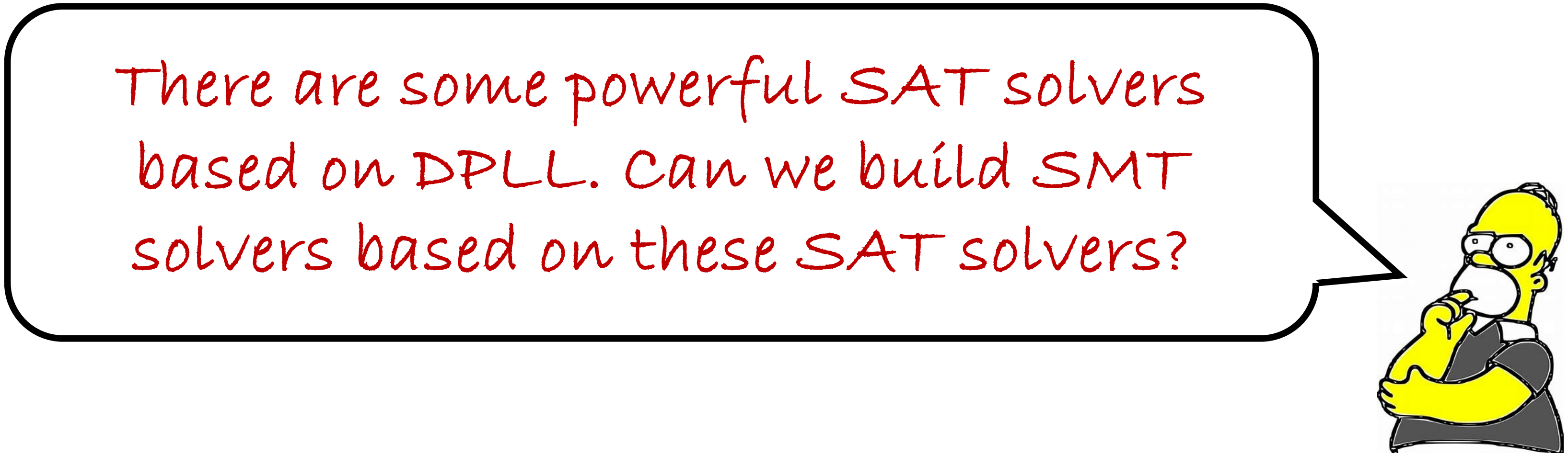
MathSAT 5
An SMT Solver for Formal Verification & More

SMT solver can do it!

Z3

CVC3

CVC4



We only consider the case of *quantifier-free formulas*.

A Quick Recap

$$\emptyset \parallel \overline{1} \vee \overline{2},\ \ 2 \vee 3,\ \ \overline{1} \vee \overline{3} \vee 4,\ \ 2 \vee \overline{3} \vee \overline{4},\ \ 1 \vee 4 \quad \Longrightarrow \quad \text{(Decide)}$$

A Quick Recap

$$\emptyset \;\|\; \overline{1}\vee\overline{2},\;\; 2\vee 3,\;\; \overline{1}\vee\overline{3}\vee 4,\;\; 2\vee\overline{3}\vee\overline{4},\;\; 1\vee 4 \quad\Longrightarrow\quad \text{(Decide)}$$
$$1^{\mathsf{d}} \;\|\; \overline{1}\vee\overline{2},\;\; 2\vee 3,\;\; \overline{1}\vee\overline{3}\vee 4,\;\; 2\vee\overline{3}\vee\overline{4},\;\; 1\vee 4 \quad\Longrightarrow\quad \text{(UnitPropagate)}$$

A Quick Recap

$$
\begin{array}{rcllllll}
\emptyset & \| & \overline{1}\vee\overline{2}, & 2\vee 3, & \overline{1}\vee\overline{3}\vee 4, & 2\vee\overline{3}\vee\overline{4}, & 1\vee 4 & \Longrightarrow \quad \text{(Decide)} \\
1^{\mathsf{d}} & \| & \overline{1}\vee\overline{2}, & 2\vee 3, & \overline{1}\vee\overline{3}\vee 4, & 2\vee\overline{3}\vee\overline{4}, & 1\vee 4 & \Longrightarrow \quad \text{(UnitPropagate)} \\
1^{\mathsf{d}}\ \overline{2} & \| & \overline{1}\vee\overline{2}, & 2\vee 3, & \overline{1}\vee\overline{3}\vee 4, & 2\vee\overline{3}\vee\overline{4}, & 1\vee 4 & \Longrightarrow \quad \text{(UnitPropagate)}
\end{array}
$$

A Quick Recap

$$
\begin{array}{rcllllllll}
\emptyset & \| & \overline{1}\vee\overline{2}, & 2\vee 3, & \overline{1}\vee\overline{3}\vee 4, & 2\vee\overline{3}\vee\overline{4}, & 1\vee 4 & \Longrightarrow & \text{(Decide)} \\
1^{\mathsf{d}} & \| & \overline{1}\vee\overline{2}, & 2\vee 3, & \overline{1}\vee\overline{3}\vee 4, & 2\vee\overline{3}\vee\overline{4}, & 1\vee 4 & \Longrightarrow & \text{(UnitPropagate)} \\
1^{\mathsf{d}}\ \overline{2} & \| & \overline{1}\vee\overline{2}, & 2\vee 3, & \overline{1}\vee\overline{3}\vee 4, & 2\vee\overline{3}\vee\overline{4}, & 1\vee 4 & \Longrightarrow & \text{(UnitPropagate)} \\
1^{\mathsf{d}}\ \overline{2}\ 3 & \| & \overline{1}\vee\overline{2}, & 2\vee 3, & \overline{1}\vee\overline{3}\vee 4, & 2\vee\overline{3}\vee\overline{4}, & 1\vee 4 & \Longrightarrow & \text{(UnitPropagate)}
\end{array}
$$

A Quick Recap

$$
\begin{array}{rcl}
\emptyset \;\|\; \overline{1}\vee\overline{2},\;\; 2\vee 3,\;\; \overline{1}\vee\overline{3}\vee 4,\;\; 2\vee\overline{3}\vee\overline{4},\;\; 1\vee 4 & \Longrightarrow & (\text{Decide}) \\
1^{\mathsf{d}} \;\|\; \overline{1}\vee\overline{2},\;\; 2\vee 3,\;\; \overline{1}\vee\overline{3}\vee 4,\;\; 2\vee\overline{3}\vee\overline{4},\;\; 1\vee 4 & \Longrightarrow & (\text{UnitPropagate}) \\
1^{\mathsf{d}}\;\overline{2} \;\|\; \overline{1}\vee\overline{2},\;\; 2\vee 3,\;\; \overline{1}\vee\overline{3}\vee 4,\;\; 2\vee\overline{3}\vee\overline{4},\;\; 1\vee 4 & \Longrightarrow & (\text{UnitPropagate}) \\
1^{\mathsf{d}}\;\overline{2}\;3 \;\|\; \overline{1}\vee\overline{2},\;\; 2\vee 3,\;\; \overline{1}\vee\overline{3}\vee 4,\;\; 2\vee\overline{3}\vee\overline{4},\;\; 1\vee 4 & \Longrightarrow & (\text{UnitPropagate}) \\
1^{\mathsf{d}}\;\overline{2}\;3\;4 \;\|\; \overline{1}\vee\overline{2},\;\; 2\vee 3,\;\; \overline{1}\vee\overline{3}\vee 4,\;\; 2\vee\overline{3}\vee\overline{4},\;\; 1\vee 4 & \Longrightarrow & (\text{Backtrack})
\end{array}
$$

A Quick Recap

$$
\begin{array}{rcllllllr}
\emptyset & \| & \overline{1}\lor\overline{2}, & 2\lor 3, & \overline{1}\lor\overline{3}\lor 4, & 2\lor\overline{3}\lor\overline{4}, & 1\lor 4 & \Longrightarrow & \text{(Decide)} \\
1^{\mathsf{d}} & \| & \overline{1}\lor\overline{2}, & 2\lor 3, & \overline{1}\lor\overline{3}\lor 4, & 2\lor\overline{3}\lor\overline{4}, & 1\lor 4 & \Longrightarrow & \text{(UnitPropagate)} \\
1^{\mathsf{d}}\ \overline{2} & \| & \overline{1}\lor\overline{2}, & 2\lor 3, & \overline{1}\lor\overline{3}\lor 4, & 2\lor\overline{3}\lor\overline{4}, & 1\lor 4 & \Longrightarrow & \text{(UnitPropagate)} \\
1^{\mathsf{d}}\ \overline{2}\ 3 & \| & \overline{1}\lor\overline{2}, & 2\lor 3, & \overline{1}\lor\overline{3}\lor 4, & 2\lor\overline{3}\lor\overline{4}, & 1\lor 4 & \Longrightarrow & \text{(UnitPropagate)} \\
1^{\mathsf{d}}\ \overline{2}\ 3\ 4 & \| & \overline{1}\lor\overline{2}, & 2\lor 3, & \overline{1}\lor\overline{3}\lor 4, & 2\lor\overline{3}\lor\overline{4}, & 1\lor 4 & \Longrightarrow & \text{(Backtrack)} \\
\overline{1} & \| & \overline{1}\lor\overline{2}, & 2\lor 3, & \overline{1}\lor\overline{3}\lor 4, & 2\lor\overline{3}\lor\overline{4}, & 1\lor 4 & \Longrightarrow & \text{(UnitPropagate)}
\end{array}
$$

A Quick Recap

$$
\begin{array}{rclcl}
\emptyset & \| & \overline{1}\vee\overline{2},\ \ 2\vee 3,\ \ \overline{1}\vee\overline{3}\vee 4,\ \ 2\vee\overline{3}\vee\overline{4},\ \ 1\vee 4 & \Longrightarrow & (\mathsf{Decide}) \\
1^{\mathsf{d}} & \| & \overline{1}\vee\overline{2},\ \ 2\vee 3,\ \ \overline{1}\vee\overline{3}\vee 4,\ \ 2\vee\overline{3}\vee\overline{4},\ \ 1\vee 4 & \Longrightarrow & (\mathsf{UnitPropagate}) \\
1^{\mathsf{d}}\ \overline{2} & \| & \overline{1}\vee\overline{2},\ \ 2\vee 3,\ \ \overline{1}\vee\overline{3}\vee 4,\ \ 2\vee\overline{3}\vee\overline{4},\ \ 1\vee 4 & \Longrightarrow & (\mathsf{UnitPropagate}) \\
1^{\mathsf{d}}\ \overline{2}\ 3 & \| & \overline{1}\vee\overline{2},\ \ 2\vee 3,\ \ \overline{1}\vee\overline{3}\vee 4,\ \ 2\vee\overline{3}\vee\overline{4},\ \ 1\vee 4 & \Longrightarrow & (\mathsf{UnitPropagate}) \\
1^{\mathsf{d}}\ \overline{2}\ 3\ 4 & \| & \overline{1}\vee\overline{2},\ \ 2\vee 3,\ \ \overline{1}\vee\overline{3}\vee 4,\ \ 2\vee\overline{3}\vee\overline{4},\ \ 1\vee 4 & \Longrightarrow & (\mathsf{Backtrack}) \\
\overline{1} & \| & \overline{1}\vee\overline{2},\ \ 2\vee 3,\ \ \overline{1}\vee\overline{3}\vee 4,\ \ 2\vee\overline{3}\vee\overline{4},\ \ 1\vee 4 & \Longrightarrow & (\mathsf{UnitPropagate}) \\
\overline{1}\ 4 & \| & \overline{1}\vee\overline{2},\ \ 2\vee 3,\ \ \overline{1}\vee\overline{3}\vee 4,\ \ 2\vee\overline{3}\vee\overline{4},\ \ 1\vee 4 & \Longrightarrow & (\mathsf{Decide})
\end{array}
$$

A Quick Recap

$$
\begin{array}{rcl}
\emptyset \parallel \overline{1}\vee\overline{2},\ \ 2\vee 3,\ \ \overline{1}\vee\overline{3}\vee 4,\ \ 2\vee\overline{3}\vee\overline{4},\ \ 1\vee 4 & \Longrightarrow & \text{(Decide)} \\
1^{\mathsf{d}} \parallel \overline{1}\vee\overline{2},\ \ 2\vee 3,\ \ \overline{1}\vee\overline{3}\vee 4,\ \ 2\vee\overline{3}\vee\overline{4},\ \ 1\vee 4 & \Longrightarrow & \text{(UnitPropagate)} \\
1^{\mathsf{d}}\ \overline{2} \parallel \overline{1}\vee\overline{2},\ \ 2\vee 3,\ \ \overline{1}\vee\overline{3}\vee 4,\ \ 2\vee\overline{3}\vee\overline{4},\ \ 1\vee 4 & \Longrightarrow & \text{(UnitPropagate)} \\
1^{\mathsf{d}}\ \overline{2}\ 3 \parallel \overline{1}\vee\overline{2},\ \ 2\vee 3,\ \ \overline{1}\vee\overline{3}\vee 4,\ \ 2\vee\overline{3}\vee\overline{4},\ \ 1\vee 4 & \Longrightarrow & \text{(UnitPropagate)} \\
1^{\mathsf{d}}\ \overline{2}\ 3\ 4 \parallel \overline{1}\vee\overline{2},\ \ 2\vee 3,\ \ \overline{1}\vee\overline{3}\vee 4,\ \ 2\vee\overline{3}\vee\overline{4},\ \ 1\vee 4 & \Longrightarrow & \text{(Backtrack)} \\
\overline{1} \parallel \overline{1}\vee\overline{2},\ \ 2\vee 3,\ \ \overline{1}\vee\overline{3}\vee 4,\ \ 2\vee\overline{3}\vee\overline{4},\ \ 1\vee 4 & \Longrightarrow & \text{(UnitPropagate)} \\
\overline{1}\ 4 \parallel \overline{1}\vee\overline{2},\ \ 2\vee 3,\ \ \overline{1}\vee\overline{3}\vee 4,\ \ 2\vee\overline{3}\vee\overline{4},\ \ 1\vee 4 & \Longrightarrow & \text{(Decide)} \\
\overline{1}\ 4\ \overline{3}^{\mathsf{d}} \parallel \overline{1}\vee\overline{2},\ \ 2\vee 3,\ \ \overline{1}\vee\overline{3}\vee 4,\ \ 2\vee\overline{3}\vee\overline{4},\ \ 1\vee 4 & \Longrightarrow & \text{(UnitPropagate)}
\end{array}
$$

A Quick Recap

$$
\begin{array}{rcllllllr}
\emptyset & \| & \overline{1}\vee\overline{2}, & 2\vee 3, & \overline{1}\vee\overline{3}\vee 4, & 2\vee\overline{3}\vee\overline{4}, & 1\vee 4 & \Longrightarrow & \text{(Decide)} \\
1^{\mathsf{d}} & \| & \overline{1}\vee\overline{2}, & 2\vee 3, & \overline{1}\vee\overline{3}\vee 4, & 2\vee\overline{3}\vee\overline{4}, & 1\vee 4 & \Longrightarrow & \text{(UnitPropagate)} \\
1^{\mathsf{d}}\ \overline{2} & \| & \overline{1}\vee\overline{2}, & 2\vee 3, & \overline{1}\vee\overline{3}\vee 4, & 2\vee\overline{3}\vee\overline{4}, & 1\vee 4 & \Longrightarrow & \text{(UnitPropagate)} \\
1^{\mathsf{d}}\ \overline{2}\ 3 & \| & \overline{1}\vee\overline{2}, & 2\vee 3, & \overline{1}\vee\overline{3}\vee 4, & 2\vee\overline{3}\vee\overline{4}, & 1\vee 4 & \Longrightarrow & \text{(UnitPropagate)} \\
1^{\mathsf{d}}\ \overline{2}\ 3\ 4 & \| & \overline{1}\vee\overline{2}, & 2\vee 3, & \overline{1}\vee\overline{3}\vee 4, & 2\vee\overline{3}\vee\overline{4}, & 1\vee 4 & \Longrightarrow & \text{(Backtrack)} \\
\overline{1} & \| & \overline{1}\vee\overline{2}, & 2\vee 3, & \overline{1}\vee\overline{3}\vee 4, & 2\vee\overline{3}\vee\overline{4}, & 1\vee 4 & \Longrightarrow & \text{(UnitPropagate)} \\
\overline{1}\ 4 & \| & \overline{1}\vee\overline{2}, & 2\vee 3, & \overline{1}\vee\overline{3}\vee 4, & 2\vee\overline{3}\vee\overline{4}, & 1\vee 4 & \Longrightarrow & \text{(Decide)} \\
\overline{1}\ 4\ \overline{3}^{\mathsf{d}} & \| & \overline{1}\vee\overline{2}, & 2\vee 3, & \overline{1}\vee\overline{3}\vee 4, & 2\vee\overline{3}\vee\overline{4}, & 1\vee 4 & \Longrightarrow & \text{(UnitPropagate)} \\
\overline{1}\ 4\ \overline{3}^{\mathsf{d}}\ 2 & \| & \overline{1}\vee\overline{2}, & 2\vee 3, & \overline{1}\vee\overline{3}\vee 4, & 2\vee\overline{3}\vee\overline{4}, & 1\vee 4 & &
\end{array}
$$

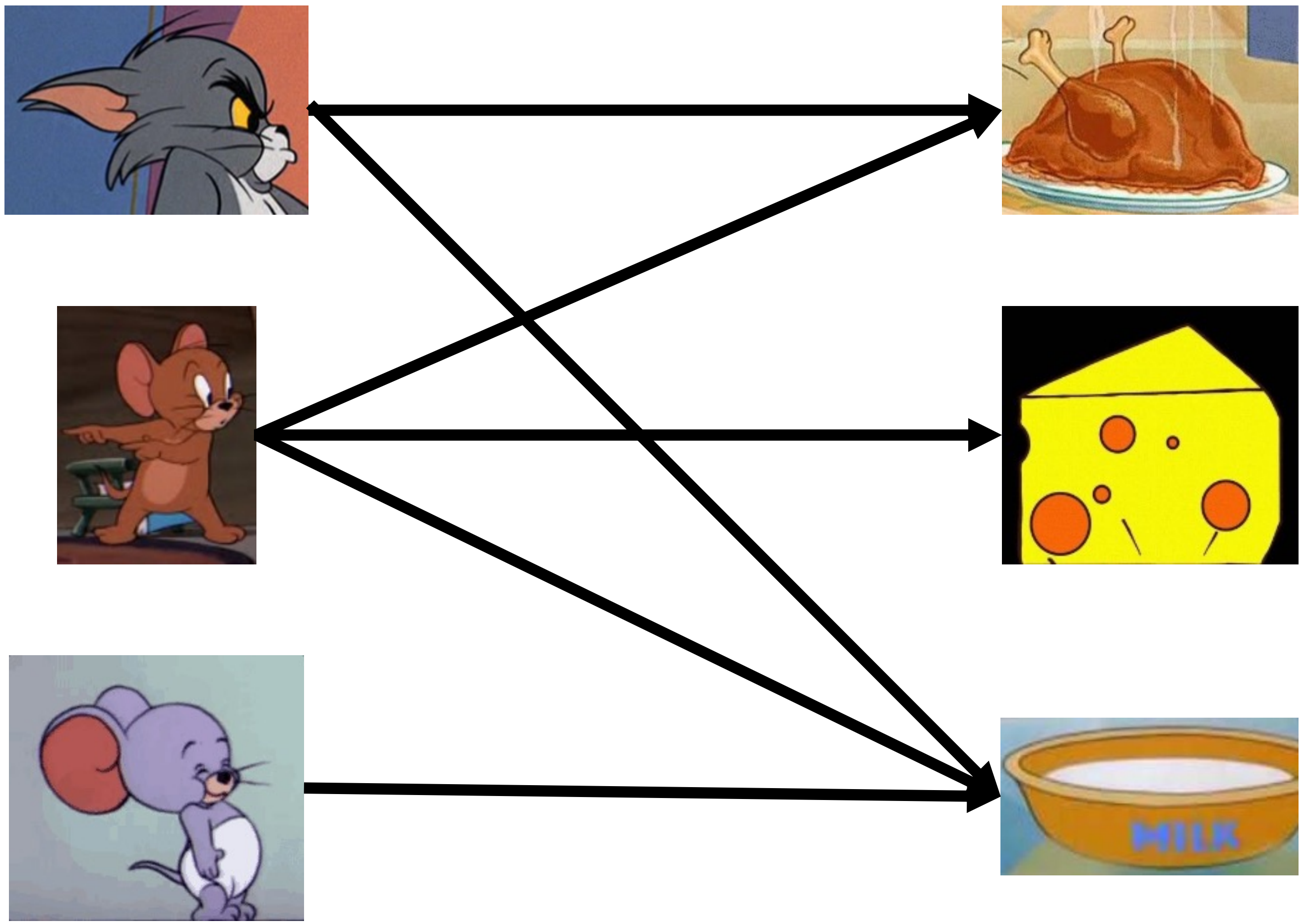
A Quick Recap

$$
\begin{array}{rcl}
\emptyset \;\|\; \bar{1}\vee\bar{2},\; 2\vee 3,\; \bar{1}\vee\bar{3}\vee 4,\; 2\vee\bar{3}\vee\bar{4},\; 1\vee 4 & \Longrightarrow & \text{(Decide)} \\
1^{\mathrm{d}} \;\|\; \bar{1}\vee\bar{2},\; 2\vee 3,\; \bar{1}\vee\bar{3}\vee 4,\; 2\vee\bar{3}\vee\bar{4},\; 1\vee 4 & \Longrightarrow & \text{(UnitPropagate)} \\
1^{\mathrm{d}}\,\bar{2} \;\|\; \bar{1}\vee\bar{2},\; 2\vee 3,\; \bar{1}\vee\bar{3}\vee 4,\; 2\vee\bar{3}\vee\bar{4},\; 1\vee 4 & \Longrightarrow & \text{(UnitPropagate)} \\
1^{\mathrm{d}}\,\bar{2}\,3 \;\|\; \bar{1}\vee\bar{2},\; 2\vee 3,\; \bar{1}\vee\bar{3}\vee 4,\; 2\vee\bar{3}\vee\bar{4},\; 1\vee 4 & \Longrightarrow & \text{(UnitPropagate)} \\
1^{\mathrm{d}}\,\bar{2}\,3\,4 \;\|\; \bar{1}\vee\bar{2},\; 2\vee 3,\; \bar{1}\vee\bar{3}\vee 4,\; 2\vee\bar{3}\vee\bar{4},\; 1\vee 4 & \Longrightarrow & \text{(Backtrack)} \\
\bar{1} \;\|\; \bar{1}\vee\bar{2},\; 2\vee 3,\; \bar{1}\vee\bar{3}\vee 4,\; 2\vee\bar{3}\vee\bar{4},\; 1\vee 4 & \Longrightarrow & \text{(UnitPropagate)} \\
\bar{1}\,4 \;\|\; \bar{1}\vee\bar{2},\; 2\vee 3,\; \bar{1}\vee\bar{3}\vee 4,\; 2\vee\bar{3}\vee\bar{4},\; 1\vee 4 & \Longrightarrow & \text{(Decide)} \\
\bar{1}\,4\,\bar{3}^{\mathrm{d}} \;\|\; \bar{1}\vee\bar{2},\; 2\vee 3,\; \bar{1}\vee\bar{3}\vee 4,\; 2\vee\bar{3}\vee\bar{4},\; 1\vee 4 & \Longrightarrow & \text{(UnitPropagate)} \\
\bar{1}\,4\,\bar{3}^{\mathrm{d}}\,2 \;\|\; \bar{1}\vee\bar{2},\; 2\vee 3,\; \bar{1}\vee\bar{3}\vee 4,\; 2\vee\bar{3}\vee\bar{4},\; 1\vee 4 & &
\end{array}
$$

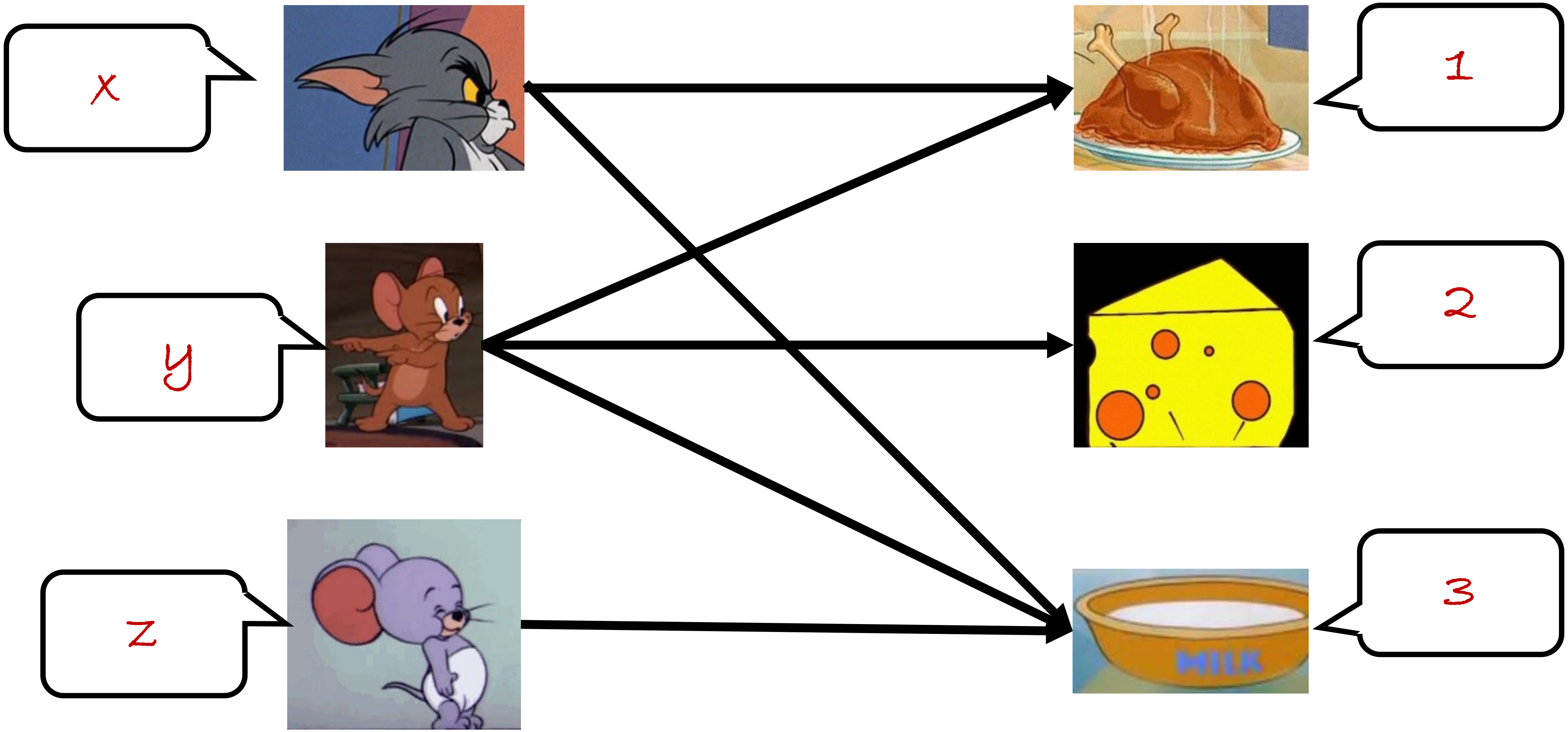
The input of a SAT solver is CNF and it uses DPLL to solve the problem.

Eager SMT converts a theory into a SAT problem

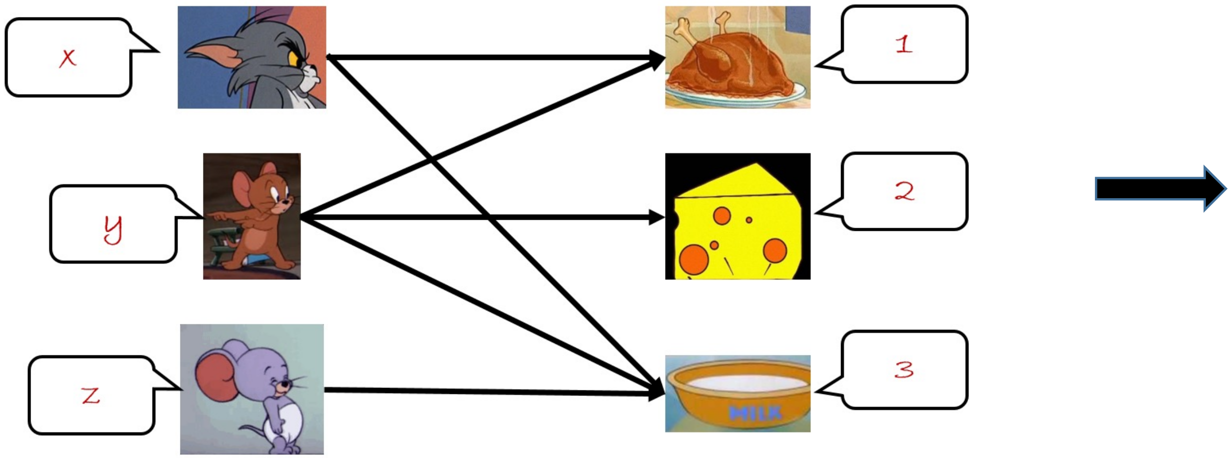
Eager SMT Techniques



Eager SMT Techniques



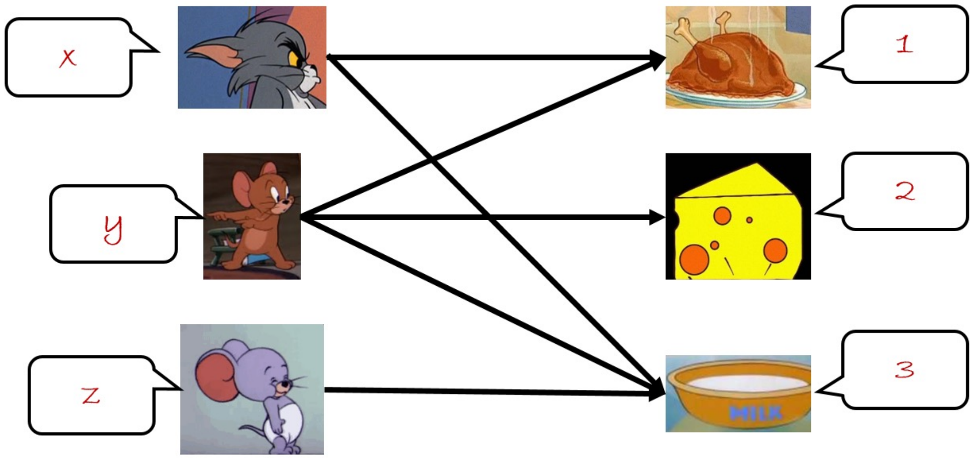
Eager SMT Techniques



$$(x = 1 \lor x = 3)\land$$
$$(y = 1 \lor y = 2 \lor y = 3)\land$$
$$(z = 3)\land$$
$$(x \neq y)\land$$
$$(x \neq z)\land$$
$$(y \neq z)$$

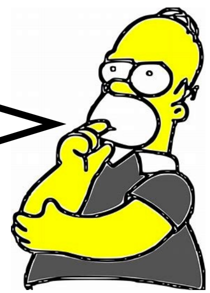
predicate logic

Eager SMT Techniques

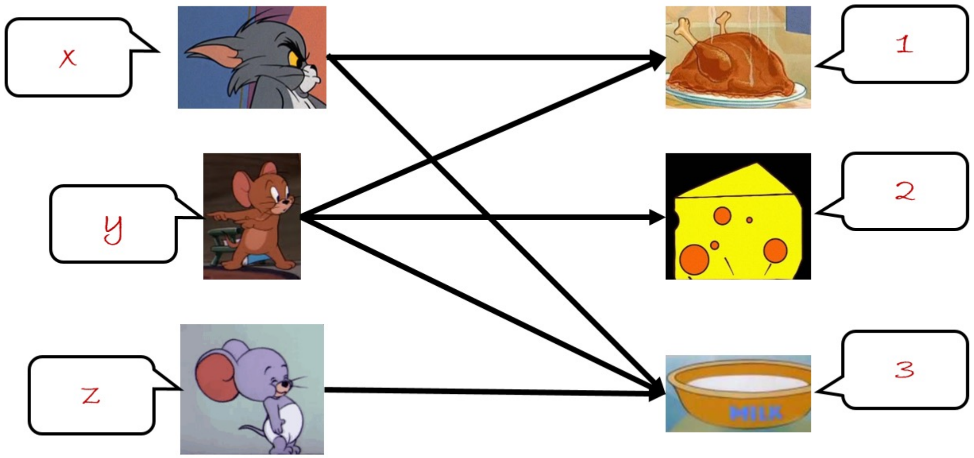


$$(x = 1 \vee x = 3) \wedge$$
$$(y = 1 \vee y = 2 \vee y = 3) \wedge$$
$$(z = 3) \wedge$$
$$(x \neq y) \wedge$$
$$(x \neq z) \wedge$$
$$(y \neq z)$$

The input formula can be translated using a satisfiability-preserving transformation into a propositional CNF formula.



Eager SMT Techniques

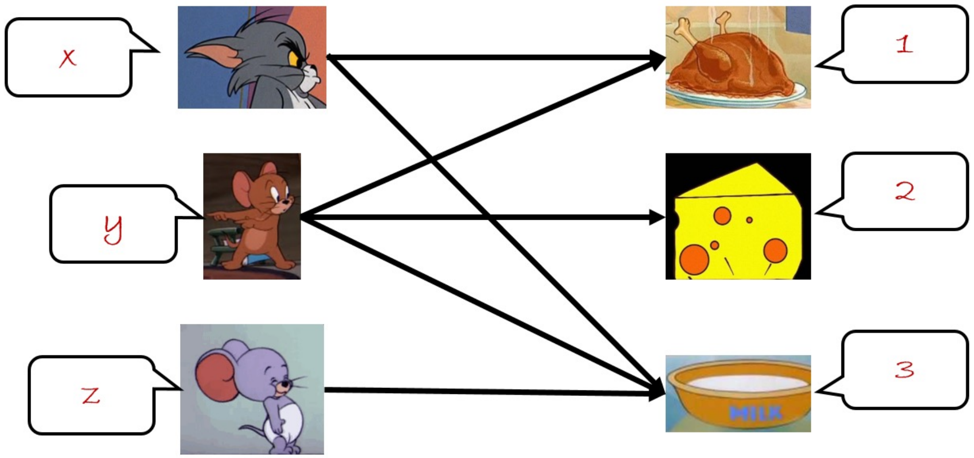


$$(x = 1 \lor x = 3) \land$$
$$(y = 1 \lor y = 2 \lor y = 3) \land$$
$$(z = 3) \land$$
$$(x \neq y) \land$$
$$(x \neq z) \land$$
$$(y \neq z)$$

Represent x, y and z by two variables each. They are similar to bits.

x	p1	p0
y	q1	q0
z	r1	r0

Eager SMT Techniques

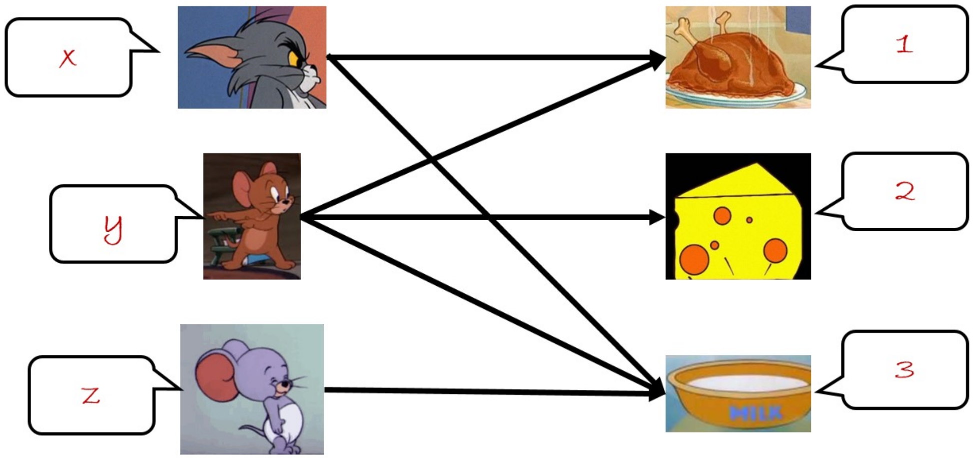


$$(x = 1 \lor x = 3)\land$$
$$(y = 1 \lor y = 2 \lor y = 3)\land$$
$$(z = 3)\land$$
$$(x \neq y)\land$$
$$(x \neq z)\land$$
$$(y \neq z)$$

$$((\neg p1 \land p0) \lor (p1 \land p0))\land$$
$$((\neg q1 \land q0) \lor (q1 \land \neg q0) \lor (q1 \land q0))\land$$
$$(r1 \land r0)\land$$
$$\neg((p1 \leftrightarrow q1) \land (p0 \leftrightarrow q0))\land$$
$$\neg((q1 \leftrightarrow r1) \land (q0 \leftrightarrow r0))\land$$
$$\neg((p1 \leftrightarrow r1) \land (p0 \leftrightarrow r0))$$

x	$\rightarrow$	p1	p0
y	$\rightarrow$	q1	q0
z	$\rightarrow$	r1	r0

Eager SMT Techniques



$$(x = 1 \lor x = 3) \land$$
$$(y = 1 \lor y = 2 \lor y = 3) \land$$
$$(z = 3) \land$$
$$(x \neq y) \land$$
$$(x \neq z) \land$$
$$(y \neq z)$$

$$((\neg p1 \land p0) \lor (p1 \land p0)) \land$$
$$((\neg q1 \land q0) \lor (q1 \land \neg q0) \lor (q1 \land q0)) \land$$
$$(r1 \land r0) \land$$
$$\neg((p1 \leftrightarrow q1) \land (p0 \leftrightarrow q0)) \land$$
$$\neg((q1 \leftrightarrow r1) \land (q0 \leftrightarrow r0)) \land$$
$$\neg((p1 \leftrightarrow r1) \land (p0 \leftrightarrow r0))$$

x → | p1 | p0 |
|---|---|

Leave the rest to powerful SAT solvers.

Eager SMT Techniques

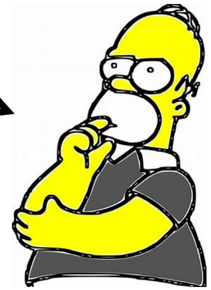
Eager SMT techniques require to encode problems to SAT:

1) Represent the state space

2) Convert the problem into the new representation

3) (Optional) Reduce redundancy

What if there are uninterpreted functions?



Eager SMT Techniques

x, y and z are bool type.

$$y = f(z) \land x = f(f(z)) \land \neg(x = f(y))$$

Function f is uninterpreted.

We want to encode it to SAT, including f.

Eager SMT Techniques

$$y = f(z) \wedge x = f(f(z)) \wedge \neg(x = f(y))$$

$$\Downarrow$$

Replace functions with fresh variables. Fresh variables are variables that never exist in original formula.

$$(y \leftrightarrow f_z) \wedge$$
$$(x \leftrightarrow f_{fz}) \wedge$$
$$\neg(x \leftrightarrow f_y)$$

Is the result equisatisfiable with the original formula?

Eager SMT Techniques

$x = f(f(z)) = f(y)$

$$y = f(z) \land x = f(f(z)) \land \neg(x = f(y))$$

UNSAT

$\Downarrow$

$$(y \leftrightarrow f_z) \land$$

$$(x \leftrightarrow f_{fz}) \land$$

$$\neg(x \leftrightarrow f_y)$$

Eager SMT Techniques

$x = f(f(z)) = f(y)$

$$y = f(z) \wedge x = f(f(z)) \wedge \neg(x = f(y))$$

UNSAT

$$\Downarrow$$

$$(y \leftrightarrow f_z) \wedge$$
$$(x \leftrightarrow f_{fz}) \wedge$$
$$\neg(x \leftrightarrow f_y)$$

$y = T, f_z = T, x = T$

$f_{fz} = T, f_y = F$

SAT

Eager SMT Techniques

x=f(f(z))=f(y)

$$y = f(z) \wedge x = f(f(z)) \wedge \neg(x = f(y))$$

UNSAT

FALSE

SAT

$$\Downarrow$$

$$(y \leftrightarrow f_z)\wedge$$

$$(x \leftrightarrow f_{fz})\wedge$$

$$\neg(x \leftrightarrow f_y)$$

$y = T, f_z = T, x = T$

$f_{fz} = T, f_y = F$

Without the relation between f(y) and f(f(z)): f(y)=f(f(z)).

Eager SMT Techniques

$$y = f(z) \wedge x = f(f(z)) \wedge \neg(x = f(y))$$

$$\Downarrow$$

$$\boxed{\begin{array}{l} ((y \leftrightarrow z) \rightarrow (f_y \leftrightarrow f_z))\wedge \\ ((y \leftrightarrow f_z) \rightarrow (f_y \leftrightarrow f_{fz})\wedge \\ ((z \leftrightarrow f_z) \rightarrow (f_z \leftrightarrow f_{fz}))\wedge \end{array}}$$

$$(y \leftrightarrow f_z)\wedge$$

$$(x \leftrightarrow f_{fz})\wedge$$

$$\neg(x \leftrightarrow f_y)$$

Property of function f:

$$a = b \rightarrow f(a) = f(b)$$

UNSAT

Eager SMT Techniques

We can always use the best available SAT solver off the shelf.

Eager SMT Techniques

We can always use the best available SAT solver off the shelf.

Issues:

Not very flexible to make them efficient,

Sophisticated translations are required for each theory.

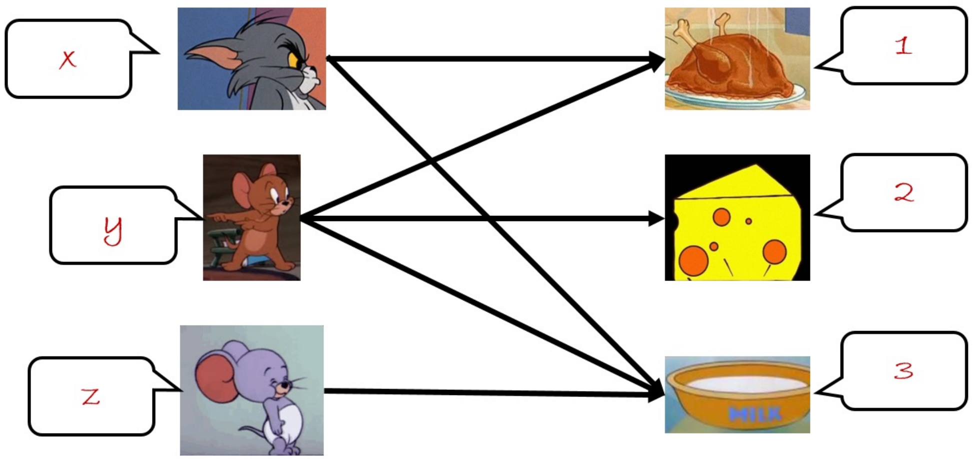
On many practical problems the translation process or the SAT solver run out of time or memory.

Lazy SMT integrates theory support into the propositional search

Lazy SMT Techniques

DEFINITION

An atomic formula in predicate logic is a formula without propositional connectives or quantifiers.



$$
\begin{aligned}
&(\boxed{x=1} \vee \boxed{x=3}) \wedge \\
&(\boxed{y=1} \vee \boxed{y=2} \vee \boxed{y=3}) \wedge \\
&(\boxed{z=3}) \wedge \\
&\neg(\boxed{x=y}) \wedge \\
&\neg(\boxed{x=z}) \wedge \\
&\neg(\boxed{y=z})
\end{aligned}
$$

atom formulas

Lazy SMT Techniques

$(x = 1 \vee x = 3)\wedge$	$(p1 \vee p2)\wedge$
$(y = 1 \vee y = 2 \vee y = 3)\wedge$	$(p3 \vee p4 \vee p5)\wedge$
$(z = 3)\wedge$	$(p6)\wedge$
$\neg(x = y)\wedge$	$\neg p7\wedge$
$\neg(x = z)\wedge$	$\neg p8\wedge$
$\neg(y = z)$	$\neg p9$

Step1: replace atomic formulas with fresh propositional variables.

Lazy SMT Techniques

$(x = 1 \vee x = 3) \wedge$
$(y = 1 \vee y = 2 \vee y = 3) \wedge$
$(z = 3) \wedge$
$\neg(x = y) \wedge$
$\neg(x = z) \wedge$
$\neg(y = z)$

$\longrightarrow$

$(p1 \vee p2) \wedge$
$(p3 \vee p4 \vee p5) \wedge$
$(p6) \wedge$
$\neg p7 \wedge$
$\neg p8 \wedge$
$\neg p9$

$\xrightarrow{\text{DPLL}}$ UNSAT

If it is unsatisfiable, the original formula is also unsatisfiable.

Step2: use SAT solvers to get assignments.

Lazy SMT Techniques

$(x = 1 \vee x = 3)\wedge$
$(y = 1 \vee y = 2 \vee y = 3)\wedge$
$(z = 3)\wedge$
$\neg(x = y)\wedge$
$\neg(x = z)\wedge$
$\neg(y = z)$

$\rightarrow$

$(p1 \vee p2)\wedge$
$(p3 \vee p4 \vee p5)\wedge$
$(p6)\wedge$
$\neg p7\wedge$
$\neg p8\wedge$
$\neg p9$

$\xrightarrow{\text{DPLL}}$

$p1 = T, p2 = F, p3 = T, p4 = F,$
$p5 = F, p6 = T, p7 = F, p8 = F,$
$p9 = F$

Step2: use SAT solvers to get assignments.

Lazy SMT Techniques

$(x = 1 \vee x = 3)\wedge$
$(y = 1 \vee y = 2 \vee y = 3)\wedge$
$(z = 3)\wedge$
$\neg(x = y)\wedge$
$\neg(x = z)\wedge$
$\neg(y = z)$

$\rightarrow$

$(p1 \vee p2)\wedge$
$(p3 \vee p4 \vee p5)\wedge$
$(p6)\wedge$
$\neg p7\wedge$
$\neg p8\wedge$
$\neg p9$

$\xrightarrow{\text{DPLL}}$

$p1 = T, p2 = F, p3 = T, p4 = F,$
$p5 = F, p6 = T, p7 = F, p8 = F,$
$p9 = F$

$\downarrow$

$x = 1, x \neq 3, y = 1, y \neq 2,$
$y \neq 3, z = 3, x \neq y, x \neq z,$
$y \neq z$

Step3: use T-solvers to check the satisfiability.

Lazy SMT Techniques

$$(x = 1 \vee x = 3)\wedge$$
$$(y = 1 \vee y = 2 \vee y = 3)\wedge$$
$$(z = 3)\wedge$$
$$\neg(x = y)\wedge$$
$$\neg(x = z)\wedge$$
$$\neg(y = z)$$

$$(p1 \vee p2)\wedge$$
$$(p3 \vee p4 \vee p5)\wedge$$
$$(p6)\wedge$$
$$\neg p7\wedge$$
$$\neg p8\wedge$$
$$\neg p9$$

DPLL

$$p1 = T, p2 = F, p3 = T, p4 = F,$$
$$p5 = F, p6 = T, p7 = F, p8 = F,$$
$$p9 = F$$

$$\boxed{x = 1}, x \neq 3, \boxed{y = 1}, y \neq 2,$$
$$y \neq 3, z = 3, \boxed{x \neq y}, x \neq z,$$
$$y \neq z$$

FALSE

Step3: use T-solvers to check the satisfiability.

Lazy SMT Techniques

$$(x = 1 \lor x = 3) \land$$
$$(y = 1 \lor y = 2 \lor y = 3) \land$$
$$(z = 3) \land$$
$$\neg(x = y) \land$$
$$\neg(x = z) \land$$
$$\neg(y = z)$$

$$(p1 \lor p2) \land$$
$$(p3 \lor p4 \lor p5) \land$$
$$(p6) \land$$
$$\neg p7 \land$$
$$\neg p8 \land$$
$$\neg p9$$

DPLL

$$p1 = T, p2 = F, p3 = T, p4 = F,$$
$$p5 = F, p6 = T, p7 = F, p8 = F,$$
$$p9 = F$$

$$\boxed{x = 1}, x \neq 3, \boxed{y = 1}, y \neq 2,$$
$$y \neq 3, z = 3, \boxed{x \neq y}, x \neq z,$$
$$y \neq z$$

FALSE

If it is unsatisfiable, tell SAT solver to give another assignment.

Lazy SMT Techniques

$(x = 1 \lor x = 3) \land$
$(y = 1 \lor y = 2 \lor y = 3) \land$
$(z = 3) \land$
$\neg(x = y) \land$
$\neg(x = z) \land$
$\neg(y = z)$

$(p1 \lor p2) \land$
$(p3 \lor p4 \lor p5) \land$
$(p6) \land$
$\neg p7 \land$
$\neg p8 \land$
$\neg p9$

DPLL

$p1 = T, p2 = F, p3 = T, p4 = F,$
$p5 = F, p6 = T, p7 = F, p8 = F,$
$p9 = F$

$\boxed{x = 1}, x \neq 3, \boxed{y = 1}, y \neq 2,$
$y \neq 3, z = 3, \boxed{x \neq y}, x \neq z,$
$y \neq z$

FALSE

But how to tell the SAT solver this assignment doesn't work?

Lazy SMT Techniques

$$(x = 1 \vee x = 3)\wedge$$
$$(y = 1 \vee y = 2 \vee y = 3)\wedge$$
$$(z = 3)\wedge$$
$$\neg(x = y)\wedge$$
$$\neg(x = z)\wedge$$
$$\neg(y = z)$$

$$(p1 \vee p2)\wedge$$
$$(p3 \vee p4 \vee p5)\wedge$$
$$(p6)\wedge$$
$$\neg p7\wedge$$
$$\neg p8\wedge$$
$$\neg p9$$

DPLL

$$p1 = T, p2 = F, p3 = T, p4 = F,$$
$$p5 = F, p6 = T, p7 = F, p8 = F,$$
$$p9 = F$$

$$\boxed{x = 1}, x \neq 3, \boxed{y = 1}, y \neq 2,$$
$$y \neq 3, z = 3, \boxed{x \neq y}, x \neq z,$$
$$y \neq z$$

FALSE

$$\neg(p1 \wedge \neg p2 \wedge p3 \wedge \neg p4\wedge$$
$$\neg p5 \wedge p6 \wedge \neg p7 \wedge \neg p8\wedge$$
$$\neg p9)$$

It may be redundant.
Clever T-solvers can
give simpler formulas.

Lazy SMT Techniques

$(x = 1 \vee x = 3)\wedge$

$(y = 1 \vee y = 2 \vee y = 3)\wedge$

$(z = 3)\wedge$

$\neg(x = y)\wedge$

$\neg(x = z)\wedge$

$\neg(y = z)$

$(p1 \vee p2)\wedge$

$(p3 \vee p4 \vee p5)\wedge$

$(p6)\wedge$

$\neg p7\wedge$

$\neg p8\wedge$

$\neg p9$

DPLL

$p1 = T, p2 = F, p3 = T, p4 = F,$

$p5 = F, p6 = T, p7 = F, p8 = F,$

$p9 = F$

$\boxed{x = 1}, x \neq 3, \boxed{y = 1}, y \neq 2,$

$y \neq 3, z = 3, \boxed{x \neq y}, x \neq z,$

$y \neq z$

FALSE

$\neg(p1 \wedge p3 \wedge \neg p7)$

It may be redundant.
Clever T-solvers can
give simpler formulas.

Lazy SMT Techniques

$(x = 1 \vee x = 3) \wedge$

$(y = 1 \vee y = 2 \vee y = 3) \wedge$

$(z = 3) \wedge$

$\neg(x = y) \wedge$

$\neg(x = z) \wedge$

$\neg(y = z)$

$(p1 \vee p2) \wedge$

$(p3 \vee p4 \vee p5) \wedge$

$(p6) \wedge$

$\neg p7 \wedge$

$\neg p8 \wedge$

$\neg p9$

DPLL

$p1 = T, p2 = F, \boxed{p3 = F} \boxed{p4 = T}$

$p5 = F, p6 = T, p7 = F, p8 = F,$

$p9 = F$

$x = 1, x \neq 3, y = 1, y \neq 2,$

$y \neq 3, z = 3, x \neq y, x \neq z,$

$y \neq z$

FALSE

$\neg(p1 \wedge p3 \wedge \neg p7)$

Lazy SMT Techniques

$$(x = 1 \lor x = 3) \land$$
$$(y = 1 \lor y = 2 \lor y = 3) \land$$
$$(z = 3) \land$$
$$\neg(x = y) \land$$
$$\neg(x = z) \land$$
$$\neg(y = z)$$

$$(p1 \lor p2) \land$$
$$(p3 \lor p4 \lor p5) \land$$
$$(p6) \land$$
$$\neg p7 \land$$
$$\neg p8 \land$$
$$\neg p9$$

DPLL

$$p1 = T, p2 = F, \boxed{p3 = F}, \boxed{p4 = T}$$
$$p5 = F, p6 = T, p7 = F, p8 = F,$$
$$p9 = F$$

$$\neg(p1 \land p3 \land \neg p7)$$

$$x = 1, x \neq 3, \boxed{y \neq 1}, \boxed{y = 2,}$$
$$y \neq 3, z = 3, x \neq y, x \neq z,$$
$$y \neq z$$

SAT

Lazy SMT Techniques

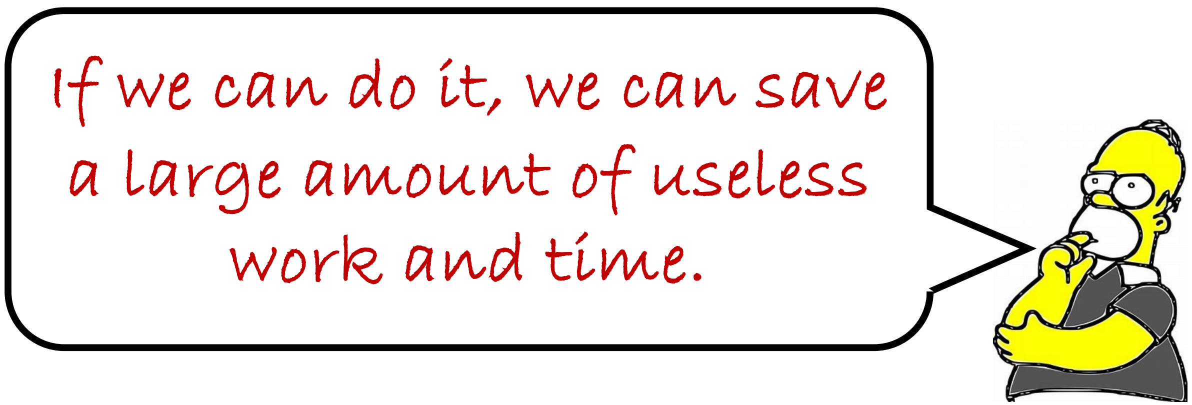
The main advantage of the lazy approach is that it can easily combine any SAT solver with any T-solver.

Several refinements exist that make the SMT procedure much more efficient when the SAT solver uses DPLL.

Incremental T-solver

In the original algorithm, T-solver checks unsatisfiability after DPLL finishes.

Can we discover errors earlier and notify SAT solver?



Incremental T-solver

$$(x = 1 \vee x = 3)\wedge$$
$$(y = 1 \vee y = 2 \vee y = 3)\wedge$$
$$(z = 3)\wedge$$
$$\neg(x = y)\wedge$$
$$\neg(x = z)\wedge$$
$$\neg(y = z)$$

$\Rightarrow$

$$(p1 \vee p2)\wedge$$
$$(p3 \vee p4 \vee p5)\wedge$$
$$(p6)\wedge$$
$$\neg p7\wedge$$
$$\neg p8\wedge$$
$$\neg p9$$

Incremental T-solver

$$(x = 1 \vee x = 3) \wedge \qquad (p1 \vee p2) \wedge$$
$$(y = 1 \vee y = 2 \vee y = 3) \wedge \qquad (p3 \vee p4 \vee p5) \wedge$$
$$(z = 3) \wedge \qquad (p6) \wedge$$
$$\neg(x = y) \wedge \qquad \neg p7 \wedge$$
$$\neg(x = z) \wedge \qquad \neg p8 \wedge$$
$$\neg(y = z) \qquad \neg p9$$

$$\emptyset \parallel \; 1 \vee 2,\; 3 \vee 4 \vee 5,\; 6,\; \overline{7},\; \overline{8},\; \overline{9} \quad \Longrightarrow (UnitProp)$$

Ignore pure literal rule here.

Incremental T-solver

$$
\begin{aligned}
&(x = 1 \vee x = 3)\wedge \\
&(y = 1 \vee y = 2 \vee y = 3)\wedge \\
&(z = 3)\wedge \\
&\neg(x = y)\wedge \\
&\neg(x = z)\wedge \\
&\neg(y = z)
\end{aligned}
\quad \Rightarrow \quad
\begin{aligned}
&(p1 \vee p2)\wedge \\
&(p3 \vee p4 \vee p5)\wedge \\
&(p6)\wedge \\
&\neg p7\wedge \\
&\neg p8\wedge \\
&\neg p9
\end{aligned}
$$

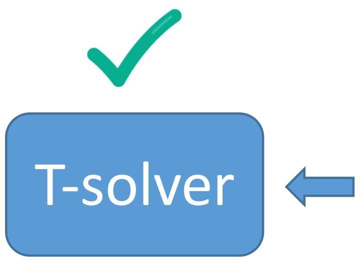
$$\emptyset \parallel 1 \vee 2,\ 3 \vee 4 \vee 5,\ 6,\ \overline{7},\ \overline{8},\ \overline{9} \Longrightarrow (UnitProp)$$

T-solver $\Leftarrow$ $$6\ \overline{7}\ \overline{8}\ \overline{9} \parallel 1 \vee 2,\ 3 \vee 4 \vee 5,\ 6,\ \overline{7},\ \overline{8},\ \overline{9} \Longrightarrow (Decide)$$

Incremental T-solver

$$
\begin{array}{lcl}
(x = 1 \lor x = 3)\land & & (p1 \lor p2)\land \\
(y = 1 \lor y = 2 \lor y = 3)\land & & (p3 \lor p4 \lor p5)\land \\
(z = 3)\land & & (p6)\land \\
\neg(x = y)\land & \Rightarrow & \neg p7\land \\
\neg(x = z)\land & & \neg p8\land \\
\neg(y = z) & & \neg p9
\end{array}
$$

$$
\begin{array}{rl}
\emptyset \parallel & 1 \lor 2,\ \ 3 \lor 4 \lor 5,\ \ 6,\ \ \overline{7},\ \ \overline{8},\ \ \overline{9} \implies (UnitProp) \\
6\ \ \overline{7}\ \ \overline{8}\ \ \overline{9} \parallel & 1 \lor 2,\ \ 3 \lor 4 \lor 5,\ \ 6,\ \ \overline{7},\ \ \overline{8},\ \ \overline{9} \implies (Decide) \\
6\ \ \overline{7}\ \ \overline{8}\ \ \overline{9}\ \ \overline{1}^{d} \parallel & 1 \lor 2,\ \ 3 \lor 4 \lor 5,\ \ 6,\ \ \overline{7},\ \ \overline{8},\ \ \overline{9} \implies (UnitProp)
\end{array}
$$



Incremental T-solver

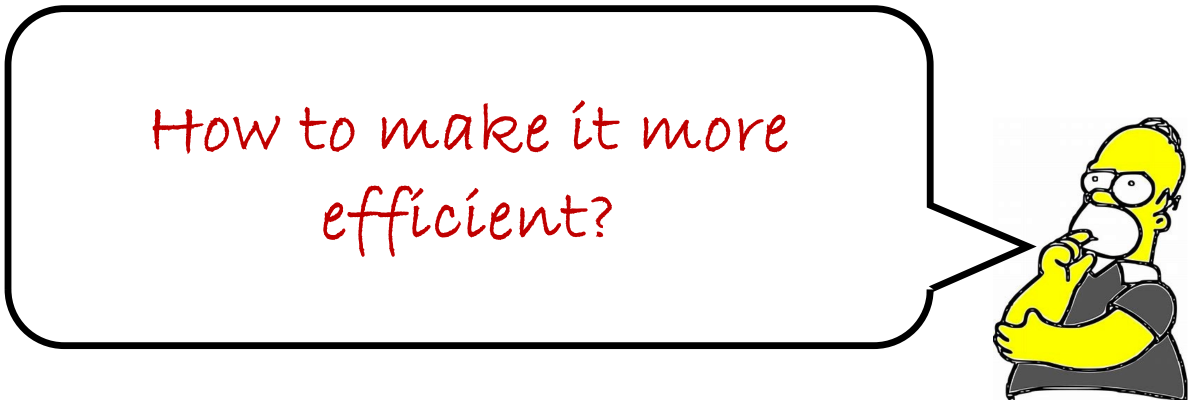
$$
\begin{array}{lcl}
(x = 1 \vee \boxed{x = 3})\wedge & & (p1 \vee p2)\wedge \\
(y = 1 \vee y = 2 \vee y = 3)\wedge & & (p3 \vee p4 \vee p5)\wedge \\
\boxed{(z = 3)}\wedge & \Longrightarrow & (p6)\wedge \\
\neg(x = y)\wedge & & \neg p7\wedge \\
\boxed{\neg(x = z)}\wedge & & \neg p8\wedge \\
\neg(y = z) & & \neg p9
\end{array}
$$

Restart the SAT solver.

$$
\begin{array}{rcll}
\emptyset & \| & 1 \vee 2,\ 3 \vee 4 \vee 5,\ 6,\ \overline{7},\ \overline{8},\ \overline{9} & \Longrightarrow (UnitProp) \\
6\ \overline{7}\ \overline{8}\ \overline{9} & \| & 1 \vee 2,\ 3 \vee 4 \vee 5,\ 6,\ \overline{7},\ \overline{8},\ \overline{9} & \Longrightarrow (Decide) \\
6\ \overline{7}\ \overline{8}\ \overline{9}\ \overline{1}^{d} & \| & 1 \vee 2,\ 3 \vee 4 \vee 5,\ 6,\ \overline{7},\ \overline{8},\ \overline{9} & \Longrightarrow (UnitProp) \\
\text{T-solver} \Leftarrow \boxed{6}\ \overline{7}\ \boxed{\overline{8}}\ \overline{9}\ \overline{1}^{d}\ \boxed{2} & \| & 1 \vee 2,\ 3 \vee 4 \vee 5,\ 6,\ \overline{7},\ \overline{8},\ \overline{9} &
\end{array}
$$

Incremental T-solver

It can be done fully eagerly, detecting unsatisfiability as soon as they are generated. But it may be much too expensive.



Incremental T-solver

It can be done fully eagerly, detecting unsatisfiability as soon as they are generated. But it may be much too expensive.

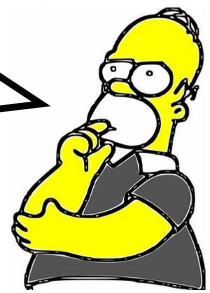
Detection can be done at regular intervals, for example, once every k literals added to the assignment.

Theory Propagation

$$
\begin{array}{rl}
\emptyset \parallel & 1 \vee 2,\ \ 3 \vee 4 \vee 5,\ \ 6,\ \ \overline{7},\ \ \overline{8},\ \ \overline{9} \implies (UnitProp)\\
6\ \ \overline{7}\ \ \overline{8}\ \ \overline{9} \parallel & 1 \vee 2,\ \ 3 \vee 4 \vee 5,\ \ 6,\ \ \overline{7},\ \ \overline{8},\ \ \overline{9} \implies (Decide)\\
6\ \ \overline{7}\ \ \overline{8}\ \ \overline{9}\ \ \overline{1}^d \parallel & 1 \vee 2,\ \ 3 \vee 4 \vee 5,\ \ 6,\ \ \overline{7},\ \ \overline{8},\ \ \overline{9} \implies (UnitProp)\\
6\ \ \overline{7}\ \ \overline{8}\ \ \overline{9}\ \ \overline{1}^d\ \ 2 \parallel & 1 \vee 2,\ \ 3 \vee 4 \vee 5,\ \ 6,\ \ \overline{7},\ \ \overline{8},\ \ \overline{9} \implies (UnitProp)
\end{array}
$$

T-solver

T-solver just checks satisfiability.
Can T-solver provide more
information to guide SAT solver to
produce assignment?



Theory Propagation

$$
\begin{aligned}
&(x = 1 \vee x = 3)\wedge \\
&(y = 1 \vee y = 2 \vee y = 3)\wedge \\
&(z = 3)\wedge \\
&\neg(x = y)\wedge \\
&\neg(x = z)\wedge \\
&\neg(y = z)
\end{aligned}
\quad \Rightarrow \quad
\begin{aligned}
&(p1 \vee p2)\wedge \\
&(p3 \vee p4 \vee p5)\wedge \\
&(p6)\wedge \\
&\neg p7\wedge \\
&\neg p8\wedge \\
&\neg p9
\end{aligned}
$$

$$\emptyset \parallel \; 1 \vee 2,\; 3 \vee 4 \vee 5,\; 6,\; \overline{7},\; \overline{8},\; \overline{9} \implies (UnitProp)$$

T-solver $\leftarrow$ $6\; \overline{7}\; \overline{8}\; \overline{9} \parallel \; 1 \vee 2,\; 3 \vee 4 \vee 5,\; 6,\; \overline{7},\; \overline{8},\; \overline{9}$

Theory Propagation

$x \neq 3$

$y \neq 3$

$$
\begin{aligned}
&(x = 1 \vee x = 3)\wedge && (p1 \vee p2)\wedge \\
&(y = 1 \vee y = 2 \vee y = 3)\wedge && (p3 \vee p4 \vee p5)\wedge \\
&(z = 3)\wedge && (p6)\wedge \\
&\neg(x = y)\wedge \quad \Rightarrow && \neg p7\wedge \\
&\neg(x = z)\wedge && \neg p8\wedge \\
&\neg(y = z) && \neg p9
\end{aligned}
$$

$$\emptyset \parallel \; 1 \vee 2,\; 3 \vee 4 \vee 5,\; 6,\; \overline{7},\; \overline{8},\; \overline{9} \implies (UnitProp)$$

$$6\ \overline{7}\ \overline{8}\ \overline{9} \parallel \; 1 \vee 2,\; 3 \vee 4 \vee 5,\; 6,\; \overline{7},\; \overline{8},\; \overline{9}$$

T-solver

Theory Propagation

$$x \neq 3$$
$$y \neq 3$$

$$
\begin{array}{lcl}
(x = 1 \lor x = 3)\land & & (p1 \lor p2)\land \\
(y = 1 \lor y = 2 \lor y = 3)\land & & (p3 \lor p4 \lor p5)\land \\
(z = 3)\land & & (p6)\land \\
\neg(x = y)\land & \Rightarrow & \neg p7\land \\
\neg(x = z)\land & & \neg p8\land \\
\neg(y = z) & & \neg p9
\end{array}
$$

$$\emptyset \parallel \; 1 \lor 2,\; 3 \lor 4 \lor 5,\; 6,\; \overline{7},\; \overline{8},\; \overline{9} \implies (UnitProp)$$

T-solver

$$6\ \overline{7}\ \overline{8}\ \overline{9} \parallel \; 1 \lor 2,\; 3 \lor 4 \lor 5,\; 6,\; \overline{7},\; \overline{8},\; \overline{9}$$

$$6\ \overline{7}\ \overline{8}\ \overline{9}\ \boxed{\overline{2}\ \overline{5}} \parallel \; 1 \lor 2,\; 3 \lor 4 \lor 5,\; 6,\; \overline{7},\; \overline{8},\; \overline{9}$$

Z3

$$(x = 1 \vee x = 3) \wedge$$
$$(y = 1 \vee y = 2 \vee y = 3) \wedge$$
$$(z = 3) \wedge$$
$$(x \neq y) \wedge$$
$$(x \neq z) \wedge$$
$$(y \neq z)$$

individuals

Predicate logic formulas

Solve it and return sat

```
from z3 import *
x = Int('x')
y = Int('y')
z = Int('z’)

s = Solver()
s.add(Or(x == 1, x == 3))
s.add(Or(y == 1, y == 2, y == 3))
s.add(z == 3)
s.add(x != y)
s.add(x != z)
s.add(y != z)

result = s.check()
print(result)
if result == sat:
    print(s.model())
```

Z3

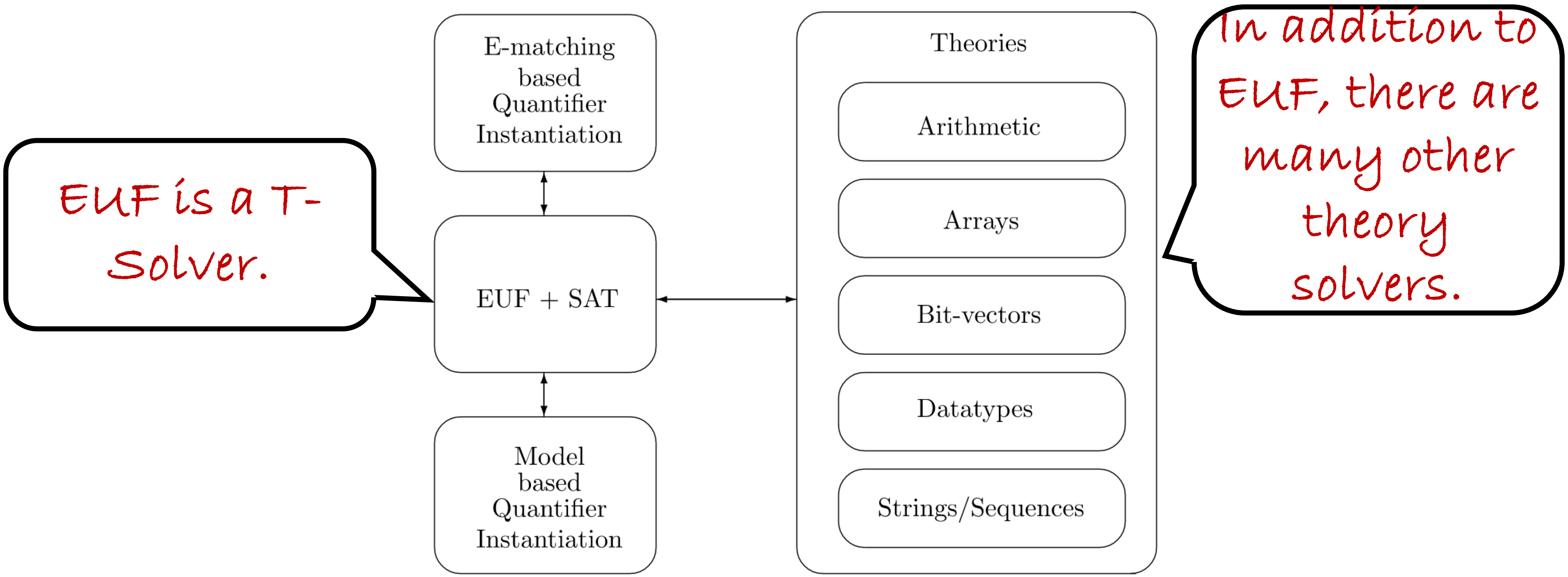
```python
from z3 import *
x = Int('x')
y = Int('y')
z = Int('z')

s = Solver()
s.add(Or(x == 1, x == 3))
s.add(Or(y == 1, y == 2, y == 3))
s.add(z == 3)
s.add(x != y)
s.add(x != z)
s.add(y != z)

result = s.check()
print(result)
if result == sat:
    print(s.model())
```

```
> python3 example.py

sat
[y = 2, z = 3, x = 1]
```



Architecture of Z3's SMT Core solver

Theory Solver

What does T-Solver looks Like?

EUF

The logic of **equality and uninterpreted function**, EUF, is a basic ingredient for (first-order) predicate logic.

$$a = b, b = c, d = e, b = s, d = t$$

EUF

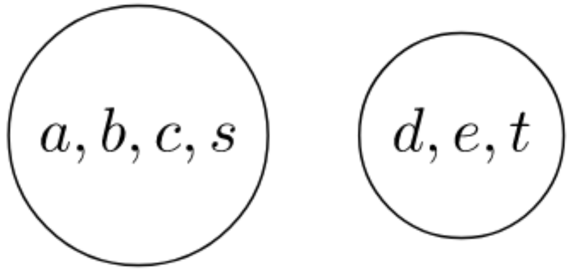
While formulas are not empty:

remove a=b;

merge(a, b);

$$a = b, b = c, d = e$$

$$b = s, d = t$$



EUF

While formulas are not empty:

        remove a=b;

        merge(a, b);

If find(a) = find(b) for some a≠b

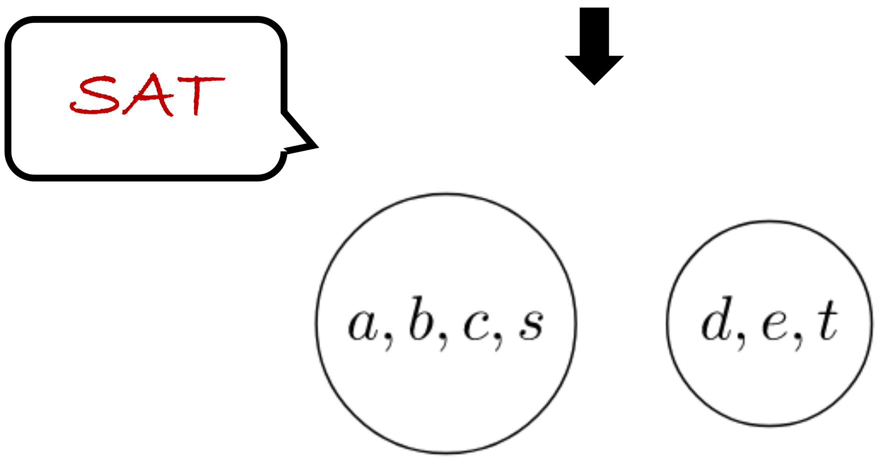
        return false;

else

        return true;

$$a = b, b = c, d = e$$

$$b = s, d = t$$



EUF

While formulas are not empty:

    remove a=b;

    merge(a, b);

If find(a) = find(b) for some a≠b

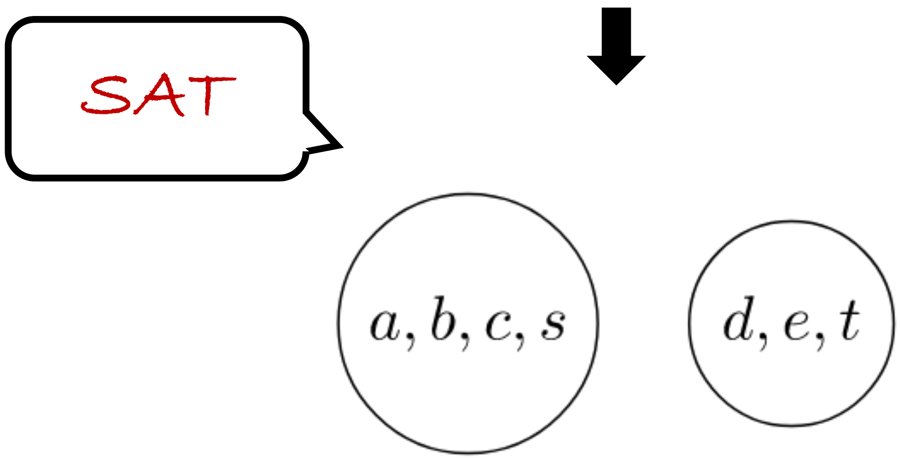
    return false;

else

    return true;

$$a = b, b = c, d = e$$

$$b = s, d = t, a \neq d$$



EUF

While formulas are not empty:

    remove a=b;

    merge(a, b);

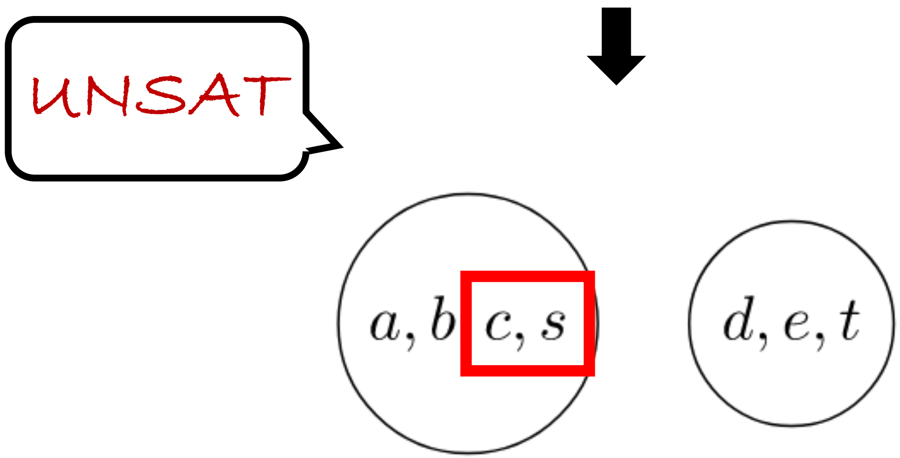
If find(a) = find(b) for some a≠b

    return false;

else

    return true;

$$a = b, b = c, d = e$$
$$b = s, d = t, \boxed{c \neq s}$$

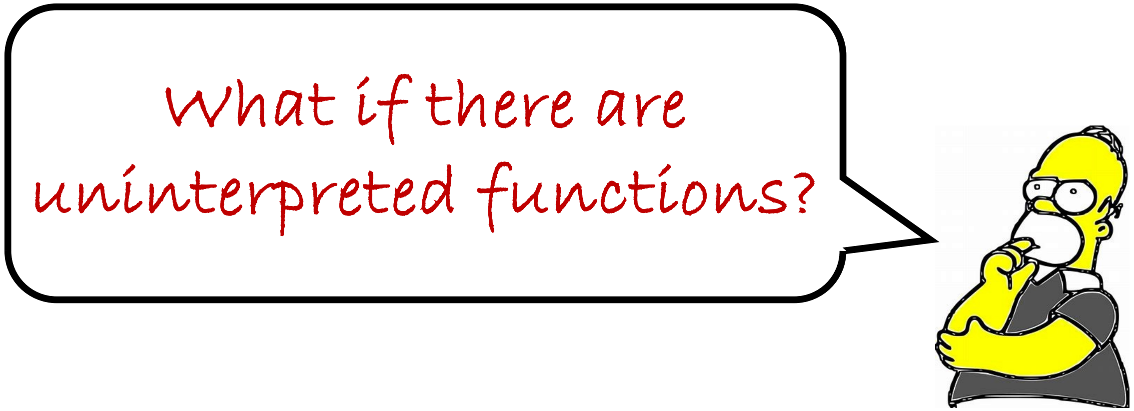


EUF

$$a = b, b = c, d = e$$

$$b = s, d = t$$

$$f(a, g(d)) \neq f(b, g(e))$$



EUF

THEOREM

Congruence rule:

$$x_1 = y_1, \dots, x_n = y_n \Rightarrow f(x_1, \dots, x_n) = f(y_1, \dots, y_n)$$

$a = b, b = c, d = e$

$b = s, d = t$

$f(a, g(d)) \neq f(b, g(e))$

We can construct the graph with congruence rule.

EUF

1) introduce constants that can be used as shorthands for sub-terms.

$$a = b, b = c, d = e$$
$$b = s, d = t$$
$$f(a, g(d)) \neq f(b, g(e))$$

$$\Rightarrow$$

$$a = b, b = c, d = e$$
$$b = s, d = t, v3 \neq v4$$

$$v1 := g(e), v2 := g(d)$$
$$v3 := f(a, v2), v4 := f(b, v1)$$

EUF

2) Working bottom-up, the congruence rule dictates how to merge groups

$$a = b, b = c, d = e$$
$$b = s, d = t, v3 \neq v4$$
$$v1 := g(e), v2 := g(d)$$
$$v3 := f(a, v2), v4 := f(b, v1)$$

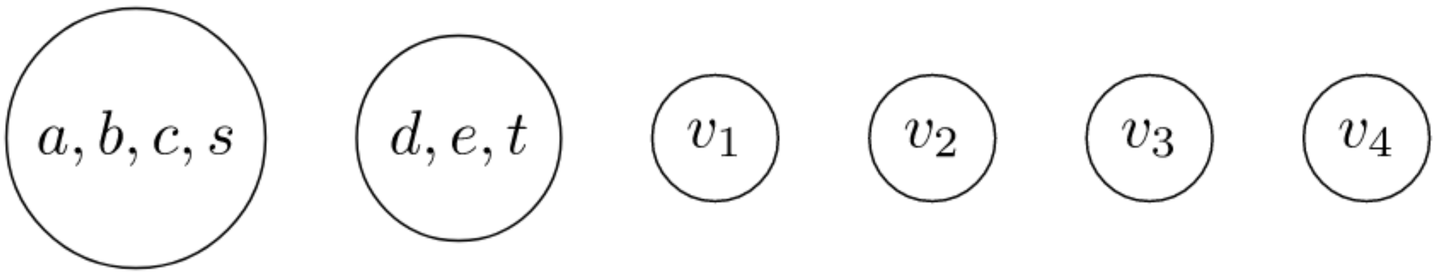
a, b, c, s $\quad$ d, e, t $\quad$ v_1 $\quad$ v_2 $\quad$ v_3 $\quad$ v_4

EUF

2) Working bottom-up, the congruence rule dictates how to merge groups

$$a = b, b = c, d = e$$
$$b = s, d = t, v3 \neq v4$$
$$v1 := g(e), v2 := g(d)$$
$$v3 := f(a, v2), v4 := f(b, v1)$$

$$e = d \Rightarrow g(e) = g(d)$$



EUF

2) Working bottom-up, the congruence rule dictates how to merge groups

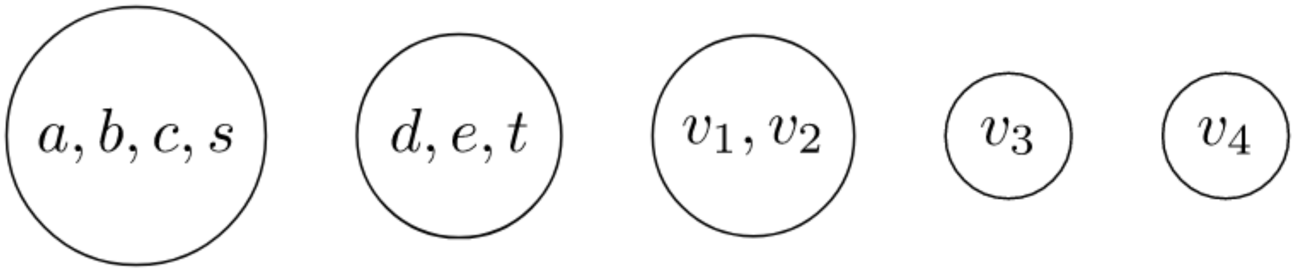
$$a = b, b = c, d = e$$

$$b = s, d = t, v3 \neq v4$$

$$v1 := g(e), v2 := g(d)$$

$$v3 := f(a, v2), v4 := f(b, v1)$$

$$e = d \Rightarrow g(e) = g(d)$$

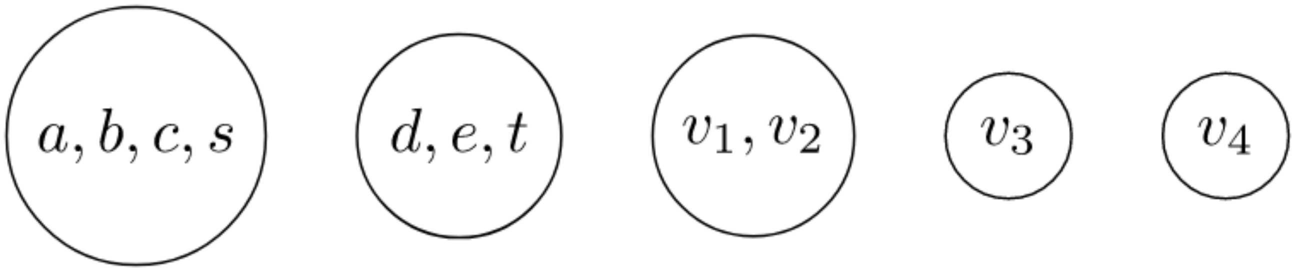


EUF

2) Working bottom-up, the congruence rule dictates how to merge groups

$$a = b, v2 = v1 \Rightarrow$$
$$f(a, v2) = f(b, v1)$$

$$a = b, b = c, d = e$$
$$b = s, d = t, v3 \neq v4$$
$$v1 := g(e), v2 := g(d)$$
$$v3 := f(a, v2), v4 := f(b, v1)$$



EUF

2) Working bottom-up, the congruence rule dictates how to merge groups

$$a = b, v2 = v1 \Rightarrow$$
$$f(a, v2) = f(b, v1)$$

$$a = b, b = c, d = e$$
$$b = s, d = t, v3 \neq v4$$
$$v1 := g(e), v2 := g(d)$$
$$v3 := f(a, v2), v4 := f(b, v1)$$

a, b, c, s $\quad$ d, e, t $\quad$ v_1, v_2 $\quad$ v_3, v_4

EUF

3) Check satisfiability like before

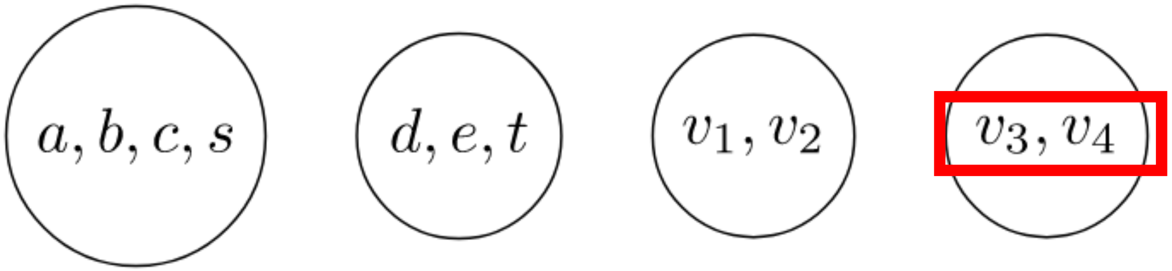
$$a = b, b = c, d = e$$

$$b = s, d = t, \boxed{v3 \neq v4}$$

$$v1 := g(e), v2 := g(d)$$

$$v3 := f(a, v2), v4 := f(b, v1)$$

UNSAT



All in One Example

$$a = b \wedge$$

$$c = f(a) \wedge$$

$$d = f(b) \wedge$$

$$\neg(c = d)$$

$\rightarrow$ $p_1 \wedge p_2 \wedge p_3 \wedge \neg p_4$ $\xrightarrow{\text{DPLL}}$

DPLL

$$p_1 \wedge p_2 \wedge p_3 \wedge \neg p_4$$

$$\emptyset \parallel p_1,\ p_2,\ p_3,\ \overline{p_4} \Rightarrow (PureLiteral)$$

$$p_1 \parallel p_1,\ p_2,\ p_3,\ \overline{p_4} \Rightarrow (PureLiteral)$$

$$p_1\ p_2 \parallel p_1,\ p_2,\ p_3,\ \overline{p_4} \Rightarrow (PureLiteral)$$

$$p_1\ p_2\ p_3 \parallel p_1,\ p_2,\ p_3,\ \overline{p_4} \Rightarrow (PureLiteral)$$

$$p_1\ p_2\ p_3,\ \overline{p_4} \parallel p_1,\ p_2,\ p_3,\ \overline{p_4}$$

All in One Example

$$a = b \wedge$$
$$c = f(a) \wedge$$
$$d = f(b) \wedge$$
$$\neg(c = d)$$

$$p_1 \wedge p_2 \wedge p_3 \wedge \neg p_4$$

DPLL

$$p_1 = T, p_2 = T,$$
$$p_3 = T, p_4 = F$$

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

EUF

$$a = b \land$$
$$c = f(a) \land$$
$$d = f(b) \land$$
$$\neg(c = d)$$

$$\Rightarrow$$

$$a = b, c = v_1, d = v_2, c \neq d,$$
$$v_1 := f(a), v_2 := f(b)$$

EUF

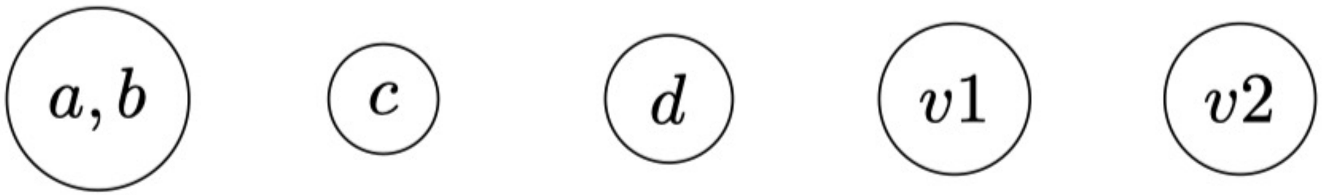
$$a = b, c = v_1, d = v_2, c \neq d,$$

$$v_1 := f(a), v_2 := f(b)$$

EUF

$$\boxed{a = b}, c = v_1, d = v_2, c \neq d,$$

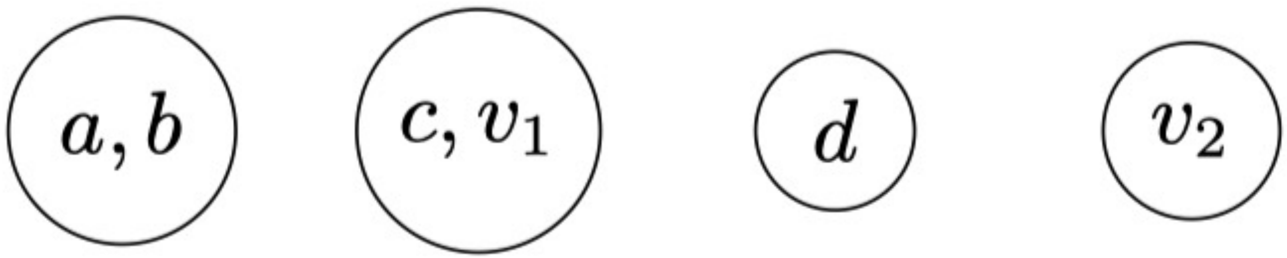
$$v_1 := f(a), v_2 := f(b)$$



EUF

$$a = b, \boxed{c = v_1}, d = v_2, c \neq d,$$

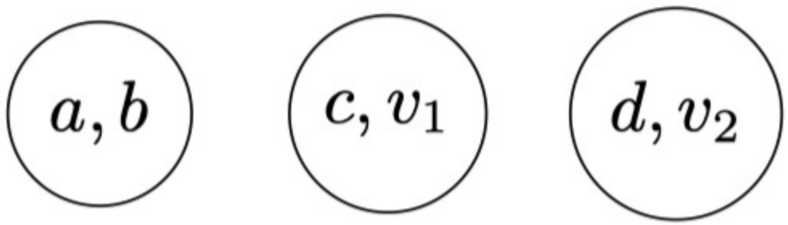
$$v_1 := f(a), v_2 := f(b)$$



EUF

$$a = b, c = v_1, \boxed{d = v_2}, c \neq d,$$

$$v_1 := f(a), v_2 := f(b)$$

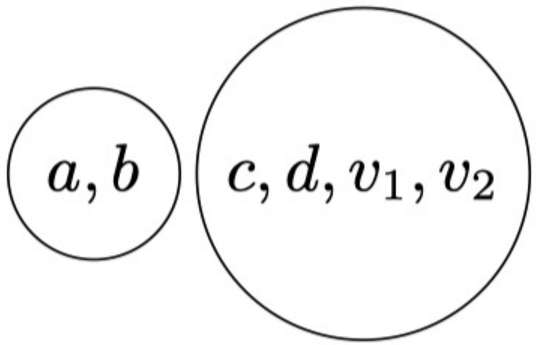


EUF

$$a = b \Rightarrow f(a) = f(b)$$

$$a = b, c = v_1, d = v_2, c \neq d,$$

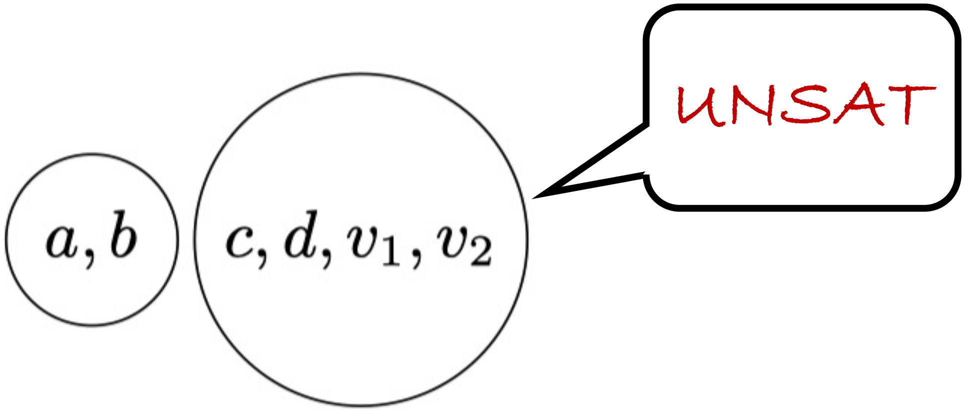
$$v_1 := f(a), v_2 := f(b)$$



EUF

$$a = b, c = v_1, d = v_2, \boxed{c \neq d,}$$

$$v_1 := f(a), v_2 := f(b)$$



ALL IN ONE

$$a = b \wedge$$
$$c = f(a) \wedge$$
$$d = f(b) \wedge$$
$$\neg(c = d)$$

$$p_1 \wedge p_2 \wedge p_3 \wedge \neg p_4 \xrightarrow{\text{DPLL}} p_1 = T, p_2 = T, p_3 = T, p_4 = F$$

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

FALSE

$$\neg(p_1 \wedge p_2 \wedge p_3 \wedge \neg p_4)$$

DPLL

$$p_1 \wedge p_2 \wedge p_3 \wedge \neg p_4 \wedge (\neg p_1 \vee \neg p_2 \vee \neg p_3 \vee p_4)$$

$$\emptyset \parallel p_1,\ p_2,\ p_3,\ \overline{p_4},\ \overline{p_1} \vee \overline{p_2} \vee \overline{p_3} \vee p_4 \Rightarrow (Decide)$$

$$p_1^d \parallel p_1,\ p_2,\ p_3,\ \overline{p_4},\ \overline{p_1} \vee \overline{p_2} \vee \overline{p_3} \vee p_4 \Rightarrow (Decide)$$

$$p_1^d\ p_2^d \parallel p_1,\ p_2,\ p_3,\ \overline{p_4},\ \overline{p_1} \vee \overline{p_2} \vee \overline{p_3} \vee p_4 \Rightarrow (Decide)$$

$$p_1^d\ p_2^d\ p_3^d \parallel p_1,\ p_2,\ p_3,\ \overline{p_4},\ \overline{p_1} \vee \overline{p_2} \vee \overline{p_3} \vee p_4 \Rightarrow (UnitProp)$$

$$p_1^d\ p_2^d\ p_3^d\ p_4 \parallel p_1,\ p_2,\ p_3,\ \overline{p_4},\ \overline{p_1} \vee \overline{p_2} \vee \overline{p_3} \vee p_4 \Rightarrow (Backtrack)$$

$$p_1^d\ p_2^d\ \overline{p_3} \parallel p_1,\ p_2,\ p_3,\ \overline{p_4},\ \overline{p_1} \vee \overline{p_2} \vee \overline{p_3} \vee p_4 \Rightarrow (Backtrack)$$

$$p_1^d\ \overline{p_2} \parallel p_1,\ p_2,\ p_3,\ \overline{p_4},\ \overline{p_1} \vee \overline{p_2} \vee \overline{p_3} \vee p_4 \Rightarrow (Backtrack)$$

$$\overline{p_1} \parallel p_1,\ p_2,\ p_3,\ \overline{p_4},\ \overline{p_1} \vee \overline{p_2} \vee \overline{p_3} \vee p_4 \Rightarrow (Fail)$$

ALL IN ONE

$$a = b \wedge$$
$$c = f(a) \wedge$$
$$d = f(b) \wedge$$
$$\neg(c = d)$$

$$\rightarrow \quad p_1 \wedge p_2 \wedge p_3 \wedge \neg p_4 \xrightarrow{\text{DPLL}}$$

UNSAT

$$\neg(p_1 \wedge p_2 \wedge p_3 \wedge \neg p_4)$$

Second-Order Logic

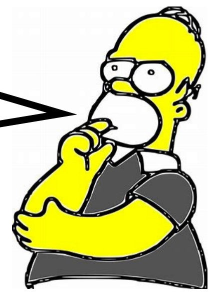
DEFINITION

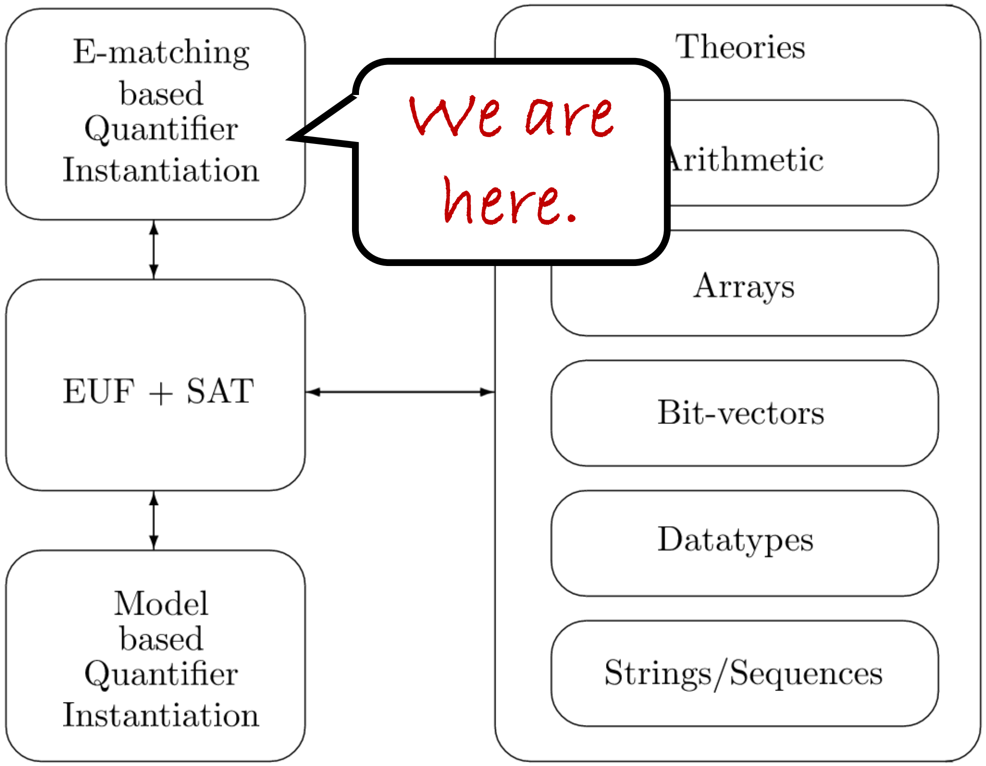
In a second-order logic formula, in addition to individuals, **predicates (functions) can also be bounded by quantifiers**. E.g.

$(\forall P)(\forall x)(P(x) \vee \neg P(x))$

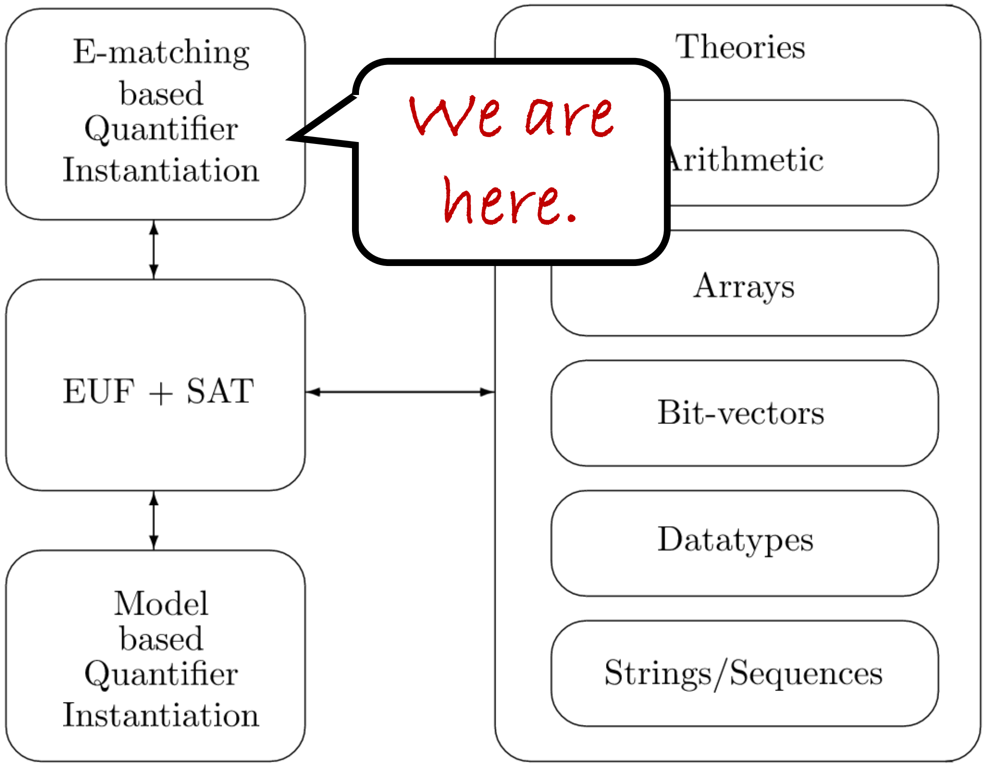
$$a = c \wedge a = f(b) \wedge d = f(e) \wedge b \neq e \wedge c = f(e)$$

The function "f" is bounded by an existential quantifier in the formula

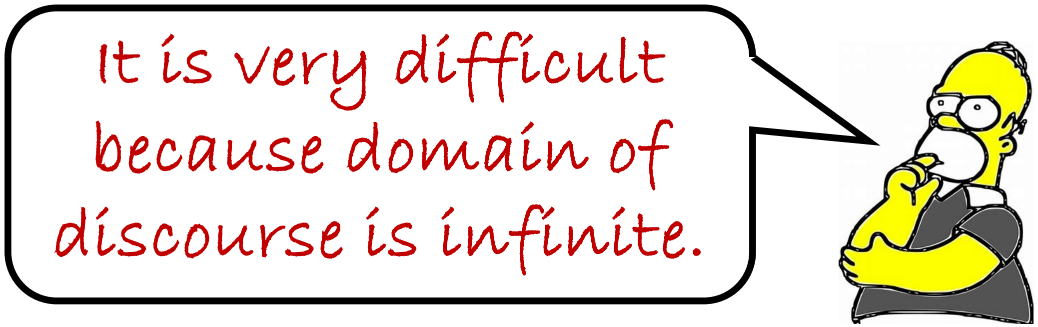




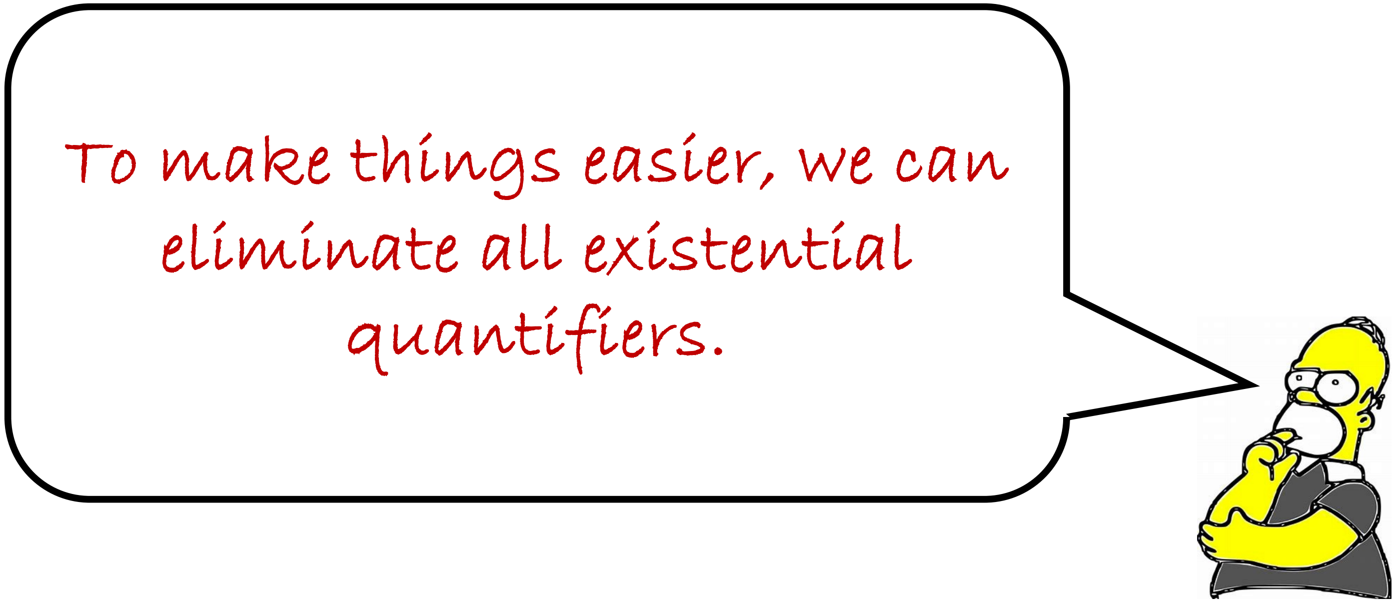
Architecture of Z3's SMT Core solver



Architecture of Z3's SMT Core solver



Quantifier


To make things easier, we can eliminate all existential quantifiers.

Quantifier

$$\neg(\forall x)(P(x)) = (\exists x)(\neg P(x))$$
$$\neg(\exists x)(P(x)) = (\forall x)(\neg P(x))$$

1) Move "not" inwards.
2) Use skolemization to eliminate existential quantifiers.
3) Now we obtain a formula with only positive universally-quantified literals.

There is no "not" before quantifiers.

A Quick Recap

$$(\forall x)(\exists y)(\exists z)S(a,b,x,y,z)$$

$$\Downarrow$$

y depends on x

$$(\forall x)(\exists z)S(a,b,x,f(x),z)$$

$$\Downarrow$$

$$(\forall x)S(a,b,x,f(x),g(x))$$

z depends on x

We don't need to move all the quantifiers to the front of all formulas. We directly do skolemization on quantifiers.

Quantifier

Now there are only universal quantifiers.

Intuitive approach: replace x with a

$$g(f(a)) = 0 \wedge (\forall x)(g(f(x)) = 1)$$

Quantifier

Now there are only universal quantifiers.

Intuitive approach: replace x with a

$$g(f(a)) = 0 \land \boxed{(\forall x)(g(f(x)) = 1)}$$

$$g(f(a)) = 1$$

Quantifier

Now there are only universal quantifiers.

Intuitive approach: replace x with a

$$g(f(a)) = 0 \wedge \boxed{(\forall x)(g(f(x)) = 1)}$$

$$g(f(a)) = 1$$

Theory Solvers

UNSAT

Thanks & Questions