SAT-Solving in a Nutshell

Prof. Dr. Erika Ábrahám

Theory of Hybrid Systems
Informatik 2

WS 09/10

1. Propositional logic, theories, normal forms
2. Propositional SAT-solving
3. SMT-solving

1. Propositional logic, theories, normal forms
2. Propositional SAT-solving
3. SMT-solving

- Abstract grammar:

$$\varphi \;:=\; \texttt{prop} \;\mid\; (\neg\varphi) \;\mid\; (\varphi \land \varphi)$$

  with $\texttt{prop} \in \texttt{Prop}$.

- Syntactic sugar:

$$\begin{array}{ccccccl}
 & & \bot & & & := & (a \land \neg a) \\
 & & \top & & & := & (a \lor \neg a) \\
(& \varphi_1 & \lor & \varphi_2 &) & := & \neg((\neg\varphi_1) \land (\neg\varphi_2)) \\
(& \varphi_1 & \rightarrow & \varphi_2 &) & := & ((\neg\varphi_1) \lor \varphi_2) \\
(& \varphi_1 & \leftrightarrow & \varphi_2 &) & := & ((\varphi_1 \rightarrow \varphi_2) \land (\varphi_2 \rightarrow \varphi_1)) \\
(& \varphi_1 & \bigoplus & \varphi_2 &) & := & (\varphi_1 \leftrightarrow (\neg\varphi_2))
\end{array}$$

- $\models \subseteq (2^{\texttt{Prop}} \times \text{Formula})$ is defined recursively:

$$
\begin{array}{ll}
\alpha \models p & \text{iff } \alpha(p) = \text{true} \\
\alpha \models \neg\varphi & \text{iff } \alpha \not\models \varphi \\
\alpha \models \varphi_1 \wedge \varphi_2 & \text{iff } \alpha \models \varphi_1 \text{ and } \alpha \models \varphi_2 \\
\alpha \models \varphi_1 \vee \varphi_2 & \text{iff } \alpha \models \varphi_1 \text{ or } \alpha \models \varphi_2 \\
\alpha \models \varphi_1 \rightarrow \varphi_2 & \text{iff } \alpha \models \varphi_1 \text{ implies } \alpha \models \varphi_2 \\
\alpha \models \varphi_1 \leftrightarrow \varphi_2 & \text{iff } \alpha \models \varphi_2 \text{ iff } \alpha \models \varphi_2
\end{array}
$$

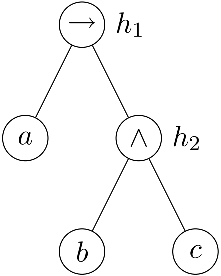
Propositional logic	$x \wedge (y \vee z)$
Equality	$(x = y \wedge \neg(y = z)) \rightarrow \neg(x = z)$
Linear arithmetic	$(2x + 3y \leq 5) \vee (x + 5y - 10z \geq 6)$
Bitvectors	$((a >> b) \& c) < c$
Arrays	$(i = j \wedge a[j] = 1) \rightarrow a[i] = 1$
Pointer	$p = q \wedge {}^*p = 5 \rightarrow {}^*q = 5$
Combined theories	$(i \leq j \wedge a[j] = 1) \rightarrow a[i] < 2$

Input for solvers:

- Negation Normal Form (NNF)
- Conjunctive Normal Form (CNF)

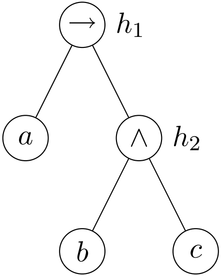
- Consider the formula
  $\phi = (a \rightarrow (b \wedge c))$

The Parse Tree:



- Associate a new auxiliary variable with each gate.
- Add constraints that define these new variables.
- Finally, enforce the root node.

- Need to satisfy:
  $(h_1 \leftrightarrow (a \rightarrow h_2))\wedge$
  $(h_2 \leftrightarrow (b \wedge c))\wedge$
  (h_1)



- Each gate encoding has a CNF representation with 3 or 4 clauses.

1. Propositional logic, theories, normal forms
2. Propositional SAT-solving
3. SMT-solving

Choose the next variable and value. Return false if all variables are assigned.

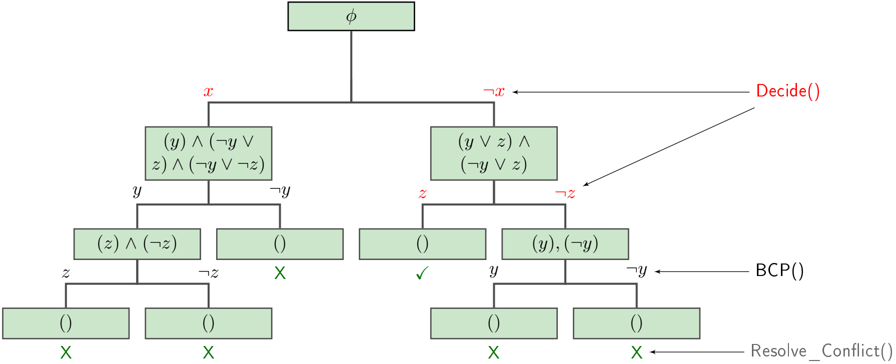
```
While (true)
 {
     if (!Decide()) return (SAT);
     while (!BCP())
          if(!Resolve_Conflict()) return (UNSAT);
 }
```

Boolean Constraint Propagation. Return false if reached a conflict.

Conflict resolution and backtracking. Return False if impossible.

Assume the CNF formula

$\phi \ : \ (x \vee y \vee z) \wedge (\neg x \vee y) \wedge (\neg y \vee z) \wedge (\neg x \vee \neg y \vee \neg z)$



- Decision
- Boolean Constraint Propagation
- Conflict resolution
- Backtracking

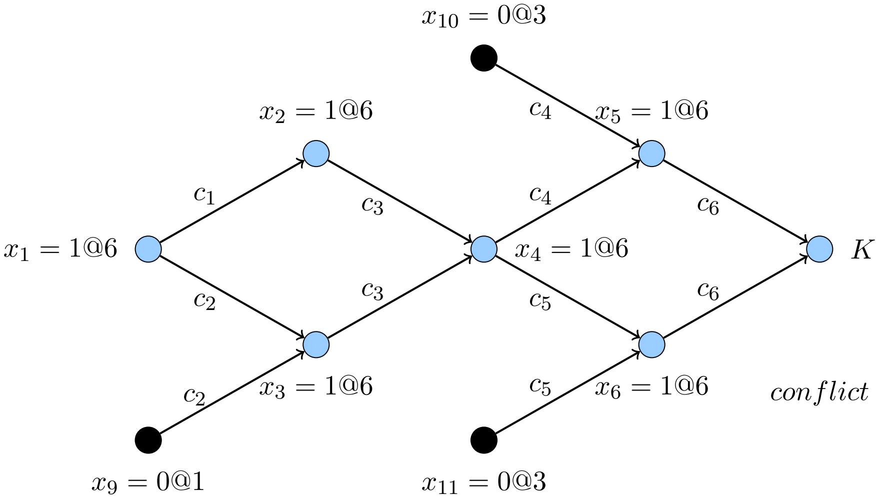
- A clause can be

Satisfied:	at least one literal is true
Unsatisfied:	all literals are false
	$\longrightarrow$ Conflict
Unit:	one literal is unassigned, the remaining literals are false
	$\longrightarrow$ Propagation
Unresolved:	all other cases

- Example: C = $(x_1 \vee x_2 \vee x_3)$

x_1	x_2	x_3	C
1	0		satisfied
0	0	0	unsatisfied
0	0		unit
	0		unresolved

- Organize the search in the form of a decision tree
  - Each node corresponds to a decision
  - Definition: Decision Level (DL) is the depth of the node in the decision tree.
  - Notation: x =v @ d
    $x \in \{0,1\}$ is assigned to v at the decision level d


$x_{10} = 0@3$
$x_2 = 1@6$
$x_5 = 1@6$
c_4
c_1
c_3
c_4
c_6
$x_1 = 1@6$
$x_4 = 1@6$
K
c_2
c_3
c_5
c_6
c_2
$x_3 = 1@6$
c_5
$x_6 = 1@6$
$conflict$
$x_9 = 0@1$
$x_{11} = 0@3$

The resolution inference rule for CNF:

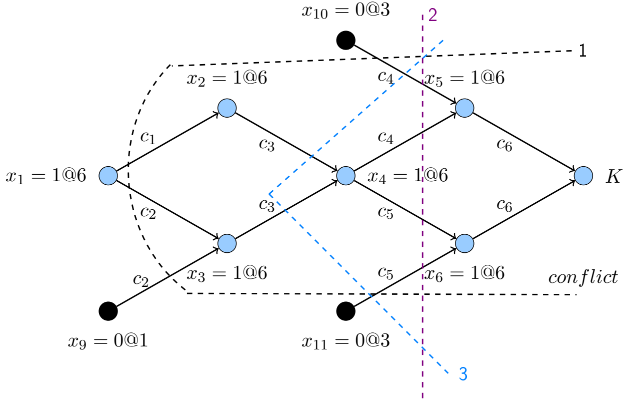
$$\frac{(l \lor l_1 \lor l_2 \lor ... \lor l_n) \qquad (\neg l \lor l'_1 \lor ... \lor l'_m)}{(l_1 \lor ... \lor l_n \lor l'_1 \lor ... \lor l'_m)} \text{ Resolution}$$

Example:

$$\frac{(a \lor b) \qquad (\neg a \lor c)}{(b \lor c)}$$

- Resolution is a sound and complete inference system for CNF.
- If the input formula is unsatisfiable, there exists a proof of the empty clause.

Apply resolution up in the implication tree until a UIP (Unique Implication Point) has been reached:



1.$(x_{10} \vee \neg x_1 \vee x_9 \vee x_{11})$

2.$(x_{10} \vee \neg x_4 \vee x_{11})$

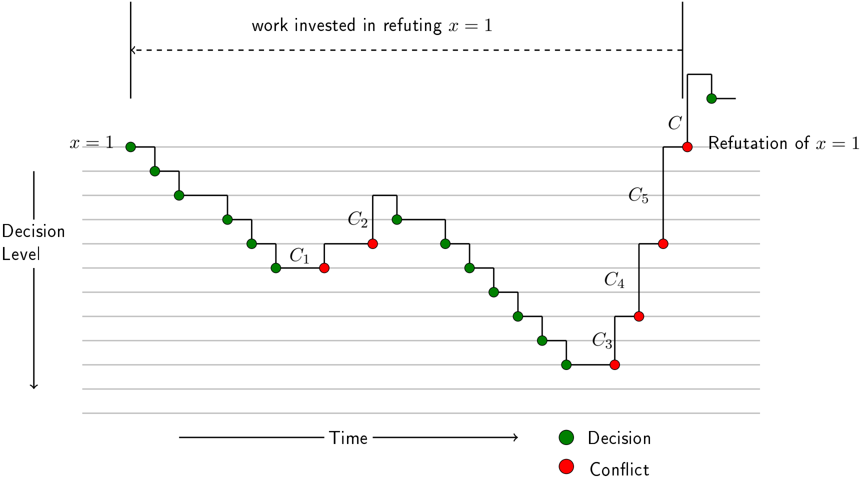
3.$(x_{10} \vee \neg x_2 \vee \neg x_3 \vee x_{11})$

$\vdots$

$\vdots$

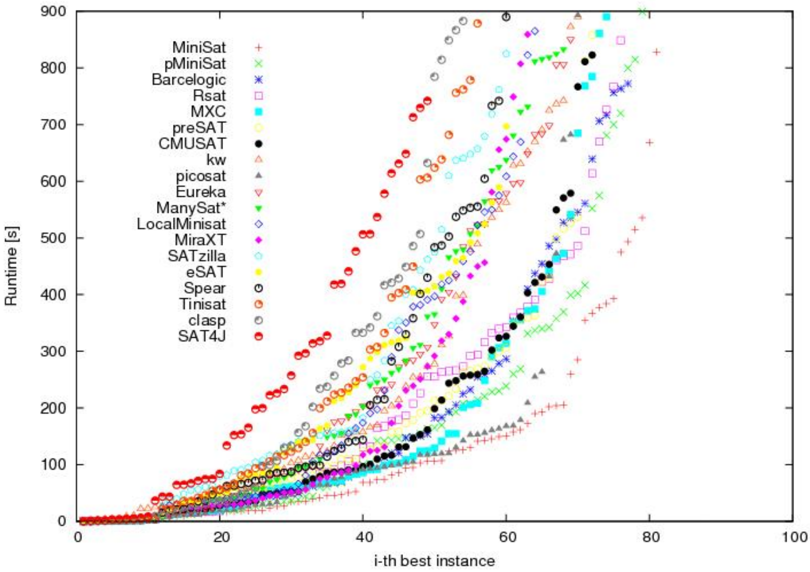
- Backtrack to the largest decision level in the conflict clause.
- This resolves the conflict and triggers an implication by the new conflict clause.

Progress of a SAT solver



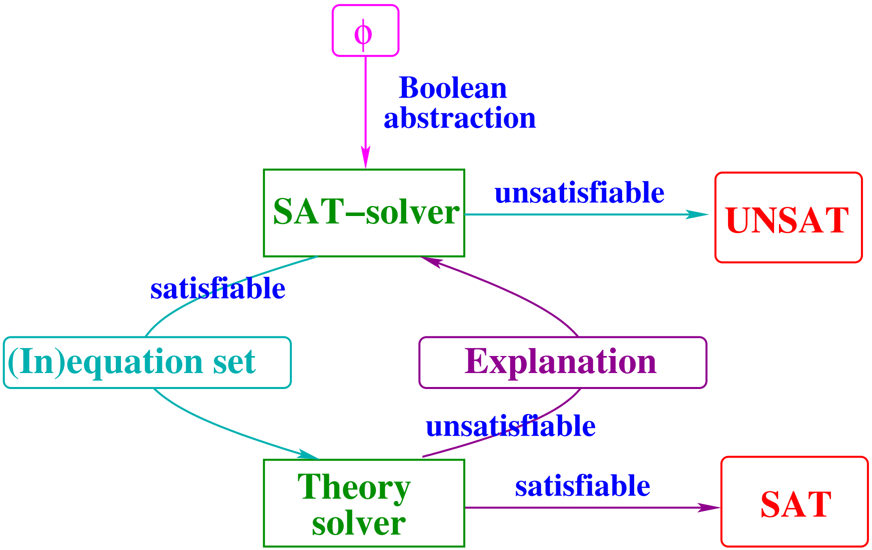
VSIDS(Variable State Independent Decaying Sum)

1. Each variable (in each polarity) has an activity initialized to 0.
2. When resolution gets applied to a clause, the activities of its literals are increased.
3. Decision: The unassigned variable with the highest activity is chosen.
4. Periodically, all the activities are divided by a constant.



Taken from http://baldur.iti.uka.de/sat-race-2008/analysis.html

1. Propositional logic, theories, normal forms
2. Propositional SAT-solving
3. SMT-solving


φ
Boolean abstraction
SAT–solver
unsatisfiable
UNSAT
satisfiable
(In)equation set
Explanation
unsatisfiable
Theory solver
satisfiable
SAT