

SAT-Solving in a Nutshell

Prof. Dr. Erika Ábrahám

Theory of Hybrid Systems
Informatik 2

February 27, 2020

- 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:

$$\varphi := p \mid (\neg\varphi) \mid (\varphi \wedge \varphi)$$

with $p \in AP$.

- Syntactic sugar:

$$\begin{aligned} \perp &:= (a \wedge \neg a) \\ \top &:= (a \vee \neg a) \\ (\varphi_1 \vee \varphi_2) &:= \neg((\neg\varphi_1) \wedge (\neg\varphi_2)) \\ (\varphi_1 \rightarrow \varphi_2) &:= ((\neg\varphi_1) \vee \varphi_2) \\ (\varphi_1 \leftrightarrow \varphi_2) &:= ((\varphi_1 \rightarrow \varphi_2) \wedge (\varphi_2 \rightarrow \varphi_1)) \\ (\varphi_1 \oplus \varphi_2) &:= (\varphi_1 \leftrightarrow (\neg\varphi_2)) \end{aligned}$$

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

$$\alpha \models p \quad \text{iff } \alpha(p) = \text{true}$$

$$\alpha \models \neg\varphi \quad \text{iff } \alpha \not\models \varphi$$

$$\alpha \models \varphi_1 \wedge \varphi_2 \quad \text{iff } \alpha \models \varphi_1 \text{ and } \alpha \models \varphi_2$$

$$\alpha \models \varphi_1 \vee \varphi_2 \quad \text{iff } \alpha \models \varphi_1 \text{ or } \alpha \models \varphi_2$$

$$\alpha \models \varphi_1 \rightarrow \varphi_2 \quad \text{iff } \alpha \models \varphi_1 \text{ implies } \alpha \models \varphi_2$$

$$\alpha \models \varphi_1 \leftrightarrow \varphi_2 \quad \text{iff } \alpha \models \varphi_1 \text{ iff } \alpha \models \varphi_2$$

$$\alpha \models \varphi_1 \oplus \varphi_2 \quad \text{iff } \alpha \models \varphi_1 \text{ iff } \alpha \not\models \varphi_2$$

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	$((a \gg b) \& c) < c$
Arrays	$(i = j \wedge a[j] = 1) \rightarrow a[i] = 1$
Pointer	$p = q \wedge *p = 5 \rightarrow *q = 5$
Combined theories	$(i \leq j \wedge a[j] = 1) \rightarrow a[i] < 2$

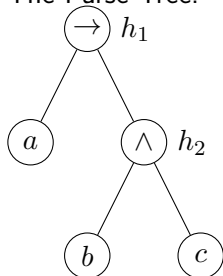
Input for solvers:

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

- Consider the formula

$$\phi = (a \rightarrow (b \wedge c))$$

The Parse Tree:



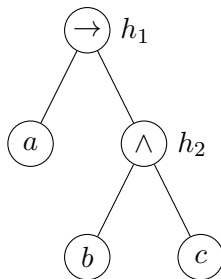
- Associate a new auxiliary variable with each gate.
- Add constraints that define these new variables.
- Finally, enforce the root node.

- 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

A Basic SAT algorithm

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

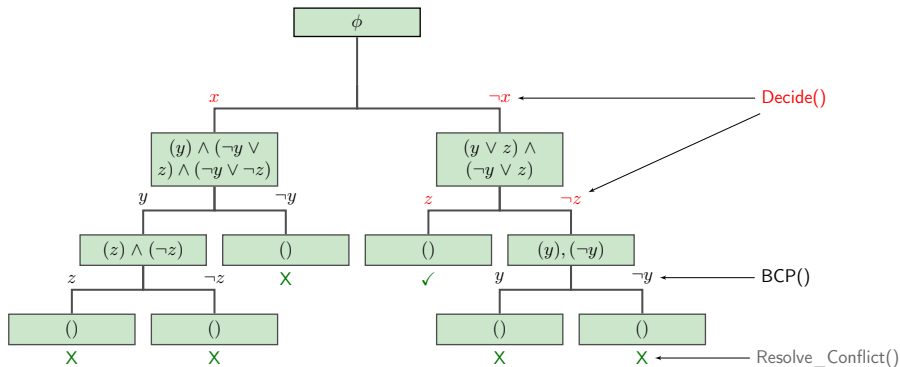
Choose the next variable and value. Return false if all variables are assigned.

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

→ **Conflict**

Unit: one literal is unassigned, the remaining literals are false

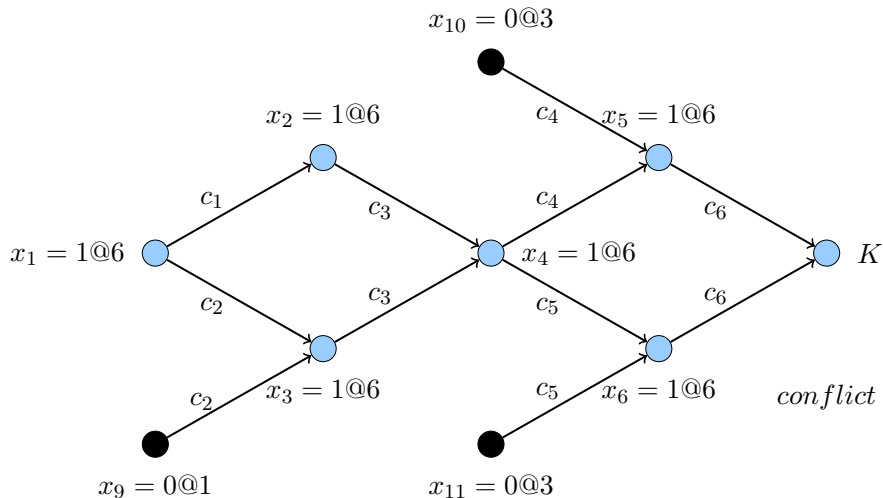
→ **Propagation**

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



The **resolution** inference rule for CNF:

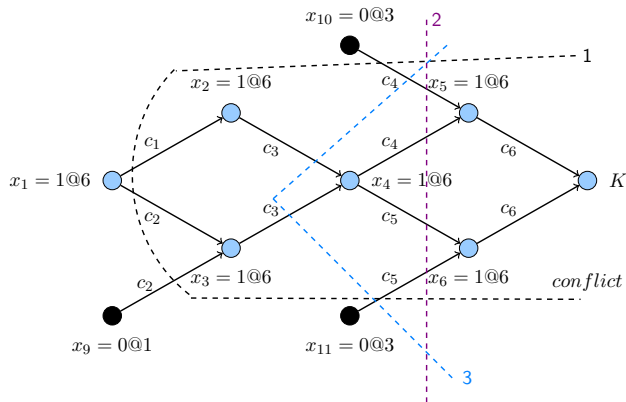
$$\frac{(l \vee l_1 \vee l_2 \vee \dots \vee l_n) \quad (\neg l \vee l'_1 \vee \dots \vee l'_m)}{(l_1 \vee \dots \vee l_n \vee l'_1 \vee \dots \vee l'_m)} \text{ Resolution}$$

Example:

$$\frac{(a \vee b) \quad (\neg a \vee c)}{(b \vee c)}$$

- 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:



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

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

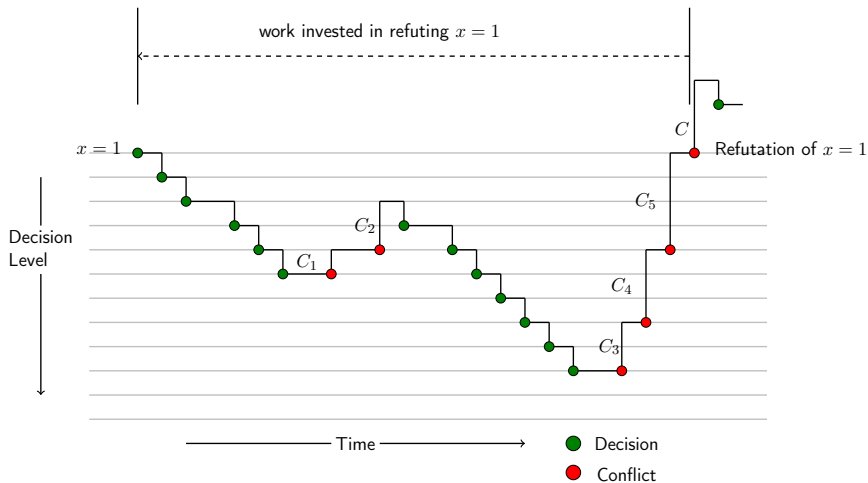
$$3. (x_{10} \vee \neg x_2 \vee \neg x_3 \vee x_{11})$$

⋮

⋮

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

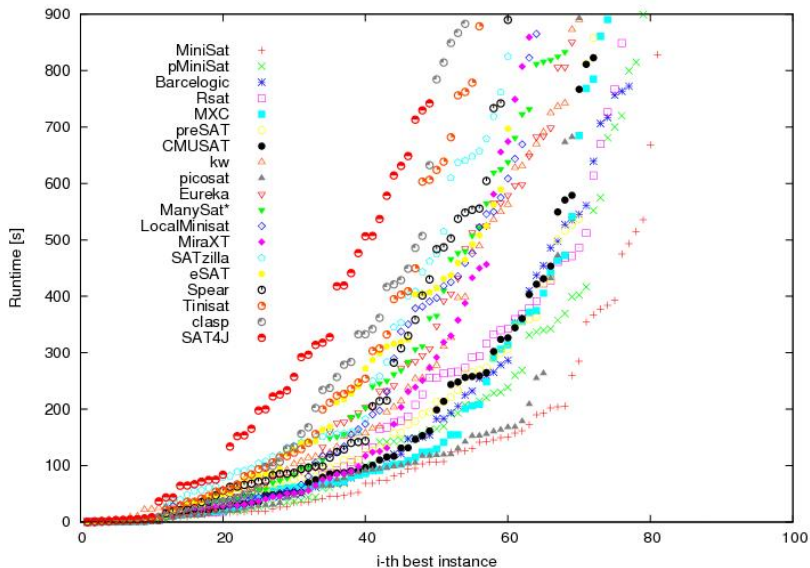
Progress of a SAT solver



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.

The SAT competitions



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

