Practical SAT Solving
Lecture 2
Carsten Sinz, Tomáš Balyo | April 30, 2019
The Resolution Rule

\[
(l \lor x_1 \lor x_2 \lor \cdots \lor x_n) \land (\bar{l} \lor y_1 \lor y_2 \lor \cdots \lor y_m) \\
(x_1 \lor x_2 \lor \cdots \lor x_n \lor y_1 \lor y_2 \lor \cdots \lor y_m)
\]

The upper two clauses are called *Input Clauses* the bottom clause is called the *Resolvent*
### The Resolution Rule

\[
(l \lor x_1 \lor x_2 \lor \cdots \lor x_n) \land (\overline{l} \lor y_1 \lor y_2 \lor \cdots \lor y_m) \\
\overline{x_1} \lor x_2 \lor \cdots \lor x_n \lor y_1 \lor y_2 \lor \cdots \lor y_m
\]

The upper two clauses are called **Input Clauses** the bottom clause is called the **Resolvent**

### Examples

- \((x_1 \lor x_3 \lor \overline{x_7}) \land (\overline{x_1} \lor x_2) \models (x_3 \lor \overline{x_7} \lor x_2)\)
The Resolution Rule

\[(l \lor x_1 \lor x_2 \lor \cdots \lor x_n) \land (\overline{l} \lor y_1 \lor y_2 \lor \cdots \lor y_m) \]

\[
\frac{(x_1 \lor x_2 \lor \cdots \lor x_n \lor y_1 \lor y_2 \lor \cdots \lor y_m)}{}
\]

The upper two clauses are called \textit{Input Clauses} the bottom clause is called the \textit{Resolvent}

Examples

- \((x_1 \lor x_3 \lor \overline{x_7}) \land (\overline{x_1} \lor x_2) \vdash (x_3 \lor \overline{x_7} \lor x_2)\)
- \((x_4 \lor x_5) \land (\overline{x_5}) \vdash (x_4)\)
The Resolution Rule

\[(l \lor x_1 \lor x_2 \lor \cdots \lor x_n) \land (\bar{l} \lor y_1 \lor y_2 \lor \cdots \lor y_m)\]

\[
\frac{(x_1 \lor x_2 \lor \cdots \lor x_n \lor y_1 \lor y_2 \lor \cdots \lor y_m)}{(x_1 \lor x_2 \lor \cdots \lor x_n \lor y_1 \lor y_2 \lor \cdots \lor y_m)}
\]

The upper two clauses are called **Input Clauses** the bottom clause is called the **Resolvent**

**Examples**

- \[(x_1 \lor x_3 \lor \bar{x_7}) \land (\bar{x_1} \lor x_2) \vdash (x_3 \lor \bar{x_7} \lor x_2)\]
- \[(x_4 \lor x_5) \land (\bar{x_5}) \vdash (x_4)\]
- \[(x_1 \lor x_2) \land (\bar{x_1} \lor \bar{x_2}) \vdash\]
The Resolution Rule

\[(l \lor x_1 \lor x_2 \lor \cdots \lor x_n) \land (\neg l \lor y_1 \lor y_2 \lor \cdots \lor y_m)\]

\[(x_1 \lor x_2 \lor \cdots \lor x_n \lor y_1 \lor y_2 \lor \cdots \lor y_m)\]

The upper two clauses are called Input Clauses the bottom clause is called the Resolvent

Examples

- \((x_1 \lor x_3 \lor \neg x_7) \land (\neg x_1 \lor x_2) \vdash (x_3 \lor \neg x_7 \lor x_2)\)
- \((x_4 \lor x_5) \land (\neg x_5) \vdash (x_4)\)
- \((x_1 \lor x_2) \land (\neg x_1 \lor \neg x_2) \vdash\)
- \((x_1) \land (\neg x_1) \vdash\)
The Resolution Rule

\[
\frac{(l \lor x_1 \lor x_2 \lor \cdots \lor x_n) \land (\bar{l} \lor y_1 \lor y_2 \lor \cdots \lor y_m)}{(x_1 \lor x_2 \lor \cdots \lor x_n \lor y_1 \lor y_2 \lor \cdots \lor y_m)}
\]

Special Cases

- **Tautological Resolvent** \((x_1 \lor x_2) \land (\overline{x_1} \lor \overline{x_2}) \vdash (x_1 \lor \overline{x_1})\)
  - Usually forbidden, does no harm, will be useful later

- **Empty Clause** \((x_1) \land (\overline{x_1}) \vdash \bot\)
  - The empty clause a.k.a conflict clause a.k.a “⊥” is unsatisfiable

Notation

- **R**\(((x_1 \lor x_2), (\overline{x_1} \lor x_3)) = (x_2 \lor x_3)\)
Theorem: Resolution maintains satisfiability

Let $F$ be a CNF formula and $C_1$ and $C_2$ two of it’s clauses with a pair complementary literals. Then $F$ is satisfiable if and only if $F \land R(C_1, C_2)$ is satisfiable.
Theorem: Resolution maintains satisfiability

Let $F$ be a CNF formula and $C_1$ and $C_2$ two of its clauses with a pair complementary literals. Then $F$ is satisfiable if and only if $F \land R(C_1, C_2)$ is satisfiable.

Proof:

- If $F$ is not satisfiable then $F \land C$ for any $C$ is also not satisfiable.
- If $F$ is satisfiable and $\phi$ is a satisfying assignment of $F$ then we show that $\phi$ also satisfies $R(C_1, C_2)$.
  - If $C_1 = (l \lor P_1)$ and $C_2 = (\bar{l} \lor P_2)$ then $R(C_1, C_2) = (P_1 \lor P_2)$
  - Since $\phi$ satisfies both $C_1$ and $C_2$ it must satisfy at least one of the literals in $P_1$ or $P_2$.
    - if $\phi$ satisfies $l$ then it satisfies some literal in $P_2$
    - if $\phi$ satisfies $\bar{l}$ then it satisfies some literal in $P_1$
Theorem: Resolution maintains satisfiability

Let $F$ be a CNF formula and $C_1$ and $C_2$ two of its clauses with a pair complementary literals. Then $F$ is satisfiable if and only if $F \land R(C_1, C_2)$ is satisfiable.

Consequences

- If we manage to resolve the empty clause ($\bot$) the original formula is unsatisfiable.
**Theorem: Resolution maintains satisfiability**

Let $F$ be a CNF formula and $C_1$ and $C_2$ two of its clauses with a pair complementary literals. Then $F$ is satisfiable if and only if $F \land R(C_1, C_2)$ is satisfiable.

**Consequences**
- If we manage to resolve the empty clause ($\bot$) the original formula is unsatisfiable

**Usage**
- Proof of unsatisfiability – Resolution Proof
  - A resolution proof is a sequence of clauses such that each clause is either a clause of the original formula or a resolvent of two previous clauses ending with $\bot$. 
SAT Tools – Resolution

Theorem: Resolution maintains satisfiability

Let $F$ be a CNF formula and $C_1$ and $C_2$ two of its clauses with a pair complementary literals. Then $F$ is satisfiable if and only if $F \land R(C_1, C_2)$ is satisfiable.

Consequences

- If we manage to resolve the empty clause ($\bot$) the original formula is unsatisfiable

Usage

- Proof of unsatisfiability – Resolution Proof
  - A resolution proof is a sequence of clauses such that each clause is either a clause of the original formula or a resolvent of two previous clauses ending with $\bot$.

Example: $(x_1 \lor x_2), (\overline{x_1} \lor x_2), (x_1 \lor \overline{x_2}), (\overline{x_1} \lor \overline{x_2})$
Theorem: Resolution maintains satisfiability

Let $F$ be a CNF formula and $C_1$ and $C_2$ two of its clauses with a pair complementary literals. Then $F$ is satisfiable if and only if $F \land R(C_1, C_2)$ is satisfiable.

Consequences

- If we manage to resolve the empty clause ($\bot$) the original formula is unsatisfiable

Usage

- Proof of unsatisfiability – Resolution Proof
  - A resolution proof is a sequence of clauses such that each clause is either a clause of the original formula or a resolvent of two previous clauses ending with $\bot$.

Example: $(x_1 \lor x_2), (\overline{x_1} \lor x_2), (x_1 \lor \overline{x_2}), (\overline{x_1} \lor \overline{x_2}) , (x_2), (\overline{x_2}), \bot$
Saturation Algorithm

- **INPUT:** CNF formula $F$
- **OUTPUT:** \{SAT, UNSAT\}

```plaintext
while (true) do
    \( R = \text{resolveAll}(F) \)
    if \( R \cap F \neq R \) then \( F = F \cup R \)
    else break
if (\( \bot \in F \)) then return UNSAT else return SAT
```
Saturation Algorithm

- INPUT: CNF formula \( F \)
- OUTPUT: \{ SAT, UNSAT \}

```
while (true) do
    \( R = \text{resolveAll}(F) \)
    if \( R \cap F \neq R \) then \( F = F \cup R \)
    else break

if (\( \bot \in F \)) then return UNSAT else return SAT
```

Properties of the saturation algorithm:
- it is sound and complete – always terminates and answers correctly
- has exponential time and space complexity (always for Pigeons)
Unit Resolution

= at least one of the resolved clauses is unit (has one literal).

Example:

\[ R((x_1 \lor x_7 \lor \overline{x_2} \lor x_4), (x_2)) = (x_1 \lor x_7 \lor x_4) \]
SAT Tools – Unit Propagation

Unit Resolution

= at least one of the resolved clauses is unit (has one literal).

Example:

\[ R((x_1 \lor x_7 \lor \overline{x_2} \lor x_4), (x_2)) = (x_1 \lor x_7 \lor x_4) \]

Unit Propagation

= a process of applying unit resolution as long as we get new clauses.

Example:

\[ (x_1) \land (x_7 \lor x_2 \lor \overline{x_3}) \land (\overline{x_1} \lor x_3) \]
\[ (x_1) \land (x_7 \lor x_2 \lor \overline{x_3}) \land (\overline{x_1} \lor x_3) \land (x_3) \]
\[ (x_1) \land (x_7 \lor x_2 \lor \overline{x_3}) \land (\overline{x_1} \lor x_3) \land (x_3) \land (x_7 \lor x_2) \]
Easy Cases

SAT is not always hard, in the following cases it is polynomially solvable

- 2-SAT
- Horn-SAT
- Hidden Horn-SAT
- SLUR
2-SAT

2-SAT Formula

= each clause has exactly 2 literals.

Example:

- \((x_1 \lor x_3) \land (x_7 \lor \overline{x}_3) \land (\overline{x}_1 \lor x_3)\)
- \((x_1 \lor x_2) \land (\overline{x}_1 \lor x_2) \land (x_1 \lor \overline{x}_2) \land (\overline{x}_1 \lor \overline{x}_2)\)

Also called Binary SAT or Quadratic SAT

How to solve 2-SAT?
How to solve 2-SAT?

Saturation Algorithm

The resolution saturation algorithm is polynomial for 2-SAT

Proof:
- Only 2-literal resolvents are possible
- There are only $O(n^2)$ 2-literal clauses on $n$ variables

Complexity:
- Both time and space $O(n^2)$
- There exists a linear algorithm! [?]
Implication graph of a formula $F$ is an oriented graph that has:

- a vertex for each literal of $F$
- 2 edges for each clause $(l_1 \lor l_2)$
  - $\overline{l_1} \rightarrow l_2$
  - $\overline{l_2} \rightarrow l_1$

Example:

$$(\overline{x_1} \lor x_2) \land (\overline{x_2} \lor x_3) \land (\overline{x_3} \lor x_1) \land (x_2 \lor x_4) \land (x_3 \lor x_4) \land (x_1 \lor x_3)$$
The next step is to analyze the *Strongly Connected Components* of the implication graphs

- SCC = there is a path in each direction between each pair
- Tarjan’s algorithm finds SCCs in $\mathcal{O}(|V| + |E|)$
- If any $x$ and $\overline{x}$ literal pair is in the same SCC then the formula is UNSAT
  - All the literals in an SCC must be all True or all False
How to find the solution?

- Construct the *Condensation* of the implication graph
  - contract each SCC into one vertex
- Topologically order the vertices of the condensation
- In reverse topological order, if the variables do not already have truth assignments, set all the terms to true.

Example: \( x_1 = x_2 = x_3 = True, x_4 = True \), the rest is already assigned.
How to solve 2-SAT?

**Linear Algorithm**

- Construct the Implication Graph
- Find all the SCCs
- Check if any SCC contains a complementary pair
- Construct a condensation of the implication graph
- Run topological sort on the condensation
- Construct the solution

**Complexity:**

- All the steps can be done in linear time