Propositional Resolution

The propositional resolution calculus operates on a set of clauses and tests unsatisfiability.

Recall that for clauses I switch between the notation as a disjunction, e.g., $P \vee Q \vee P \vee \neg R$, and the multiset notation, e.g., $\{P, Q, P, \neg R\}$. This makes no difference as we consider $\vee$ in the context of clauses always modulo AC. Note that $\bot$, the empty disjunction, corresponds to $\emptyset$, the empty multiset. Clauses are typically denoted by letters C, D, possibly with subscript.

Resolution Inference Rules

Resolution $(N \uplus \{C_1 \vee P, C_2 \vee \neg P\}) \Rightarrow_{\text{RES}}$
$(N \cup \{C_1 \vee P, C_2 \vee \neg P\} \cup \{C_1 \vee C_2\})$

Factoring $(N \uplus \{C \vee L \vee L\}) \Rightarrow_{\text{RES}}$
$(N \cup \{C \vee L \vee L\} \cup \{C \vee L\})$

2.6.1 Theorem (Soundness & Completeness)

The resolution calculus is sound and complete:

$$N \text{ is unsatisfiable iff } N \Rightarrow^*_{\text{RES}} N' \text{ and } \bot \in N' \text{ for some } N'$$

Resolution Reduction Rules

Subsumption $(N \uplus \{C_1, C_2\}) \Rightarrow_{\text{RES}} (N \cup \{C_1\})$
provided $C_1 \subset C_2$

Tautology Deletion $(N \uplus \{C \vee P \vee \neg P\}) \Rightarrow_{\text{RES}} (N)$

Condensation $(N \uplus \{C_1 \vee L \vee L\}) \Rightarrow_{\text{RES}} (N \cup \{C_1 \vee L\})$

Subsumption Resolution $(N \uplus \{C_1 \vee L, C_2 \vee \text{comp}(L)\}) \Rightarrow_{\text{RES}}$
$(N \cup \{C_1 \vee L, C_2\})$
where $C_1 \subseteq C_2$

2.6.6 Theorem (Resolution Termination)

If reduction rules are preferred over inference rules and no inference rule is applied twice to the same clause(s), then $\Rightarrow^{+}_{\text{RES}}$ is well-founded.

The Overall Picture

Application System + Problem
System Algorithm + Implementation
Algorithm Calculus + Strategy
Calculus Logic + States + Rules
Logic Syntax + Semantics

Conflict Driven Clause Learning (CDCL)

The CDCL calculus tests satisfiability of a finite set N of propositional clauses.

I assume that $\bot \notin N$ and that the clauses in N do not contain duplicate literal occurrences. Furthermore, duplicate literal occurrences are always silently removed during rule applications of the calculus. (Exhaustive Condensation.)

The CDCL calculus explicitely builds a candidate model for a clause set. If such a sequence of literals $L_1, \ldots, L_n$ satisfies the clause set N, it is done. If not, there is a false clause $C \in N$ with respect to $L_1, \ldots, L_n$.

Now instead of just backtracking through the literals $L_1, \ldots, L_n$, CDCL generates in addition a new clause, called *learned clause* via resolution, that actually guarantees that the subsequence of $L_1, \ldots, L_n$ that caused C to be false will not be generated anymore.

This causes CDCL to be exponentially more powerful in proof length than its predecessor DPLL or Tableau.