Lecture 5
Resolution
Resolution proof calculus

Print version of the lecture in *Introduction to Logic for Computer Science*

presented by Prof Hongseok Yang

These lecture notes are very minor variants of the ones made by Prof James Worrell and Dr Christoph Haase for their 'Logic and Proof' course at Oxford.

5.1

In this lecture we introduce a proof calculus for propositional logic. In general, a proof calculus consists of rules of inference which can be used to derive a series of conclusions from a series of hypotheses in a mechanical fashion. We study a particularly simple proof calculus, called *resolution*, which has only one rule of inference. This extreme simplicity makes it amenable both to automation and theoretical analysis. The main results in this lecture are the *soundness* and *completeness* of resolution. Roughly speaking, soundness says that everything we prove is valid, while completeness says that anything valid can be proved.

1 Set Representation of CNF Formulas

To apply resolution to a formula F it is required that F be in CNF. Thus, if necessary, one must first convert F to CNF using the equivalences introduced in Lecture 2. It is moreover convenient to regard each clause in a CNF formula as a set of literals and a formula as a set of clauses. For example, a CNF formula

$$F = (p_1 \vee \neg p_2) \wedge (p_3 \vee \neg p_4 \vee p_5) \wedge (\neg p_2)$$

is represented in the following form:

$$F = \{\{p_1, \neg p_2\}, \{p_3, \neg p_4, p_5\}, \{\neg p_2\}\}\,.$$

Since the elements of a set do not have order or multiplicity, the set representation of CNF formulas is a normal form modulo the associativity, commutativity, and idempotence of conjunction and disjunction. For example, the following CNF formulas F, G, H are all represented by the set $\{\{p_3\}, \{p_1, \neg p_2\}\}$:

$$\begin{aligned} F &= (p_3 \wedge (p_1 \vee p_1 \vee \neg p_2) \wedge p_3) \\ G &= ((\neg p_2 \vee p_1 \vee \neg p_2) \wedge (p_3 \vee p_3)) \\ H &= (p_3 \wedge (\neg p_2 \vee p_1))\,. \end{aligned}$$

In the set representation of CNF formulas the empty clause, denoted $\Box$, is equivalent to false since it is the unit of the disjunction operation[1]. Therefore if a CNF formula contains the empty clause it is equivalent to false, i.e., it is unsatisfiable. By contrast, if a CNF formula *is* the empty set of clauses, then it is equivalent to true (the unit of the conjunction operation).

[1]By unit, we refer to the property of false that $\mathit{false} \vee F = F \vee \mathit{false} = F$ for every formula F. In similar fashion, the sum of the empty set of natural numbers is 0, the unit of the addition operation, and the product of the empty set of natural numbers is 1, the unit of the multiplication operation.

2 Resolution

Recall that given a literal L the complementary literal $\overline{L}$ is defined by

$$\overline{L} := \begin{cases} \neg p & \text{if } L = p \\ p & \text{if } L = \neg p. \end{cases}$$

Definition 1. Let C_1 and C_2 be clauses. A clause R is called a *resolvent* of C_1 and C_2 if there are complementary literals $L \in C_1$ and $\overline{L} \in C_2$ such that

$$R = (C_1 \setminus \{L\}) \cup (C_2 \setminus \{\overline{L}\}) .$$

In the situation of Definition 1 we say that R *is derived from* C_1 *and* C_2 *by resolution*. We denote this graphically by

$$\frac{C_1 \quad C_2}{R\,.}$$

For example, $\{p_1, p_3, \neg p_4\}$ is a resolvent of $\{p_1, p_2, \neg p_4\}$ and $\{\neg p_2, p_3\}$, and the empty clause is a resolvent of $\{p_1\}$ and $\{\neg p_1\}$:

$$\frac{\{p_1, p_2, \neg p_4\} \quad \{\neg p_2, p_3\}}{\{p_1, p_3, \neg p_4\}} \qquad \frac{\{p_1\} \quad \{\neg p_1\}}{\Box}$$

A *derivation* (or proof) of a clause C from a set of clauses F is a sequence $C_1, C_2, \ldots, C_m$ of clauses where $C_m = C$ and for each $i = 1, 2, \ldots, m$ either $C_i \in F$ or C_i is a resolvent of C_j and C_k for some $j, k < i$. A derivation of the empty clause $\Box$ from a formula F is called a *refutation* of F.

Example 2. A resolution refutation of the CNF formula

$$\{\{x, \neg y\}, \{y, z\}, \{\neg x, \neg y, z\}, \{\neg z\}\}$$

is as follows:

1.	$\{x, \neg y\}$	Assumption	5.	$\{\neg x, z\}$	2,4 Resolution
2.	$\{y, z\}$	Assumption	6.	$\{\neg z\}$	Assumption
3.	$\{x, z\}$	1,2 Resolution	7.	$\{z\}$	3,5 Resolution
4.	$\{\neg x, \neg y, z\}$	Assumption	8.	$\Box$	6,7 Resolution

The same derivation can also be represented by the following *proof tree*:

$$\dfrac{\dfrac{\dfrac{\{x, \neg y\} \qquad \{y, z\}}{\{x, z\}} \qquad \dfrac{\{y, z\} \qquad \{\neg x, \neg y, z\}}{\{\neg x, z\}}}{\{z\}} \qquad \{\neg z\}}{\Box}$$

A resolution refutation of a formula F can be seen as a proof that F is unsatisfiable. This will be made formal in the next section. Resolution can be used to prove entailments by transforming them to refutations. For example, the refutation in Example 2 can be used to show that

$$(x \vee \neg y) \wedge (y \vee z) \wedge (\neg x \vee \neg y \vee z) \models z\,.$$

Given a set of clauses F, we are interested in the set of all clauses that can be derived from F using resolution. We characterise this set abstractly as follows:

Definition 3. Let F be a set of clauses. Then $Res(F)$ is defined as

$$Res(F) = F \cup \{R : \ R \text{ is a resolvent of two clauses in } F\}.$$

Furthermore we define

$$\begin{aligned} Res^0(F) &= F \\ Res^{n+1}(F) &= Res(Res^n(F)) \quad \text{for } n \geq 0\,, \end{aligned}$$

and write

$$Res^*(F) = \bigcup_{n \geq 0} Res^n(F)\,.$$

Proposition 4. $C \in Res^*(F)$ if and only if there is a derivation of C from F.

3 Soundness and Completeness

Lemma 5 (Resolution Lemma)**.** *Let F be a CNF formula represented as a set of clauses. Suppose R is a resolvent of two clauses C_1 and C_2 of F. Then $F \equiv F \cup \{R\}$.*

Proof. Let $\mathcal{A}$ be an assignment. It is obvious that if $\mathcal{A} \models F \cup \{R\}$, then $\mathcal{A} \models F$. We argue the converse as follows. Suppose $\mathcal{A} \models F$ and write $R = (C_1 \setminus \{L\}) \cup (C_2 \setminus \{\overline{L}\})$ for some literal L, where $L \in C_1$ and $\overline{L} \in C_2$. There are two cases:

Case (i): If $\mathcal{A} \models L$, then since $\mathcal{A} \models C_2$ it follows that $\mathcal{A} \models C_2 \setminus \{\overline{L}\}$, and thus $\mathcal{A} \models R$.

Case (ii): If $\mathcal{A} \models \overline{L}$, then since $\mathcal{A} \models C_1$ it follows that $\mathcal{A} \models C_1 \setminus \{L\}$, and thus $\mathcal{A} \models R$. □

Soundness says that we only derive a contradiction from an unsatisfiable set of clauses.

Theorem 6 (Soundness)**.** *If there is a derivation of □ from F, then F is unsatisfiable.*

Proof. Suppose that $C_1, C_2, \ldots, C_m = \Box$ is a proof of □ from F. By repeated applications of the Resolution Lemma, we have that F is equivalent to $F \cup \{C_1, C_2, \ldots, C_m\}$. But the latter set of clauses includes the empty clause and thus is unsatisfiable. □

Completeness is the converse of soundness: it says that if a CNF formula F is unsatisfiable, then we can derive the empty clause from F by resolution. Before giving the proof, we specialise the notion of substitution of propositional formulas, introduced in an earlier lecture.

Given a set of clauses F involving a propositional variable p, let $F[\mathit{false}/p]$ (read "F with false for p") denote that clause set obtained by replacing p everywhere in F with false, replacing $\neg p$ with true, and then simplifying by applying the unit laws. That is, $F[\mathit{false}/p]$ arises by deleting p from all clauses in F that contain p, and removing from F all clauses that contain $\neg p$ (since they become true). Likewise we define $F[\mathit{true}/p]$ by deleting all clauses containing p and removing $\neg p$ from all clauses containing $\neg p$.

Example 7. If $F = \{\{p_1, p_3\}, \{\neg p_1, p_2\}, \{\neg p_2, p_3\}, \{\neg p_3\}\}$, then

$$F[\mathit{false}/p_3] = \{\{p_1\}, \{\neg p_1, p_2\}, \{\neg p_2\}\}$$

and

$$F[\mathit{true}/p_3] = \{\{\neg p_1, p_2\}, \Box\}.$$

Theorem 8 (Completeness)**.** *If F is unsatisfiable, then there is a derivation of $\Box$ from F.*

Proof. The proof is by induction on the number n of different propositional variables in F.

Base case. The base case is that $n = 0$, i.e., F does not mention any propositional variables. Then F either contains no clauses or it contains only the empty clause. In the former case F is equivalent to $true$. But F is assumed to be unsatisfiable, so it must be that $F = \{\Box\}$ and there is a one-line resolution refutation of F.

Induction step. Suppose that F is an unsatisfiable set of clauses that mentions variables $p_1, p_2, \ldots, p_n$ and let $F_0 := F[\mathit{false}/p_n]$ and $F_1 := F[\mathit{true}/p_n]$.

Since F is unsatisfiable, F_0 is also unsatisfiable. Moreover F_0 mentions one fewer variable than F so by the induction hypothesis there is a resolution proof $C_0, C_1, \ldots, C_m = \Box$ that derives the empty clause from F_0. For each clause C_i in the above proof that comes from F_0, either C_i is already in F or $C_i \cup \{p_n\}$ is in F. By replacing the deleted copies of p_n from the latter type of clauses and propagating p_n through the proof we obtain a new proof $C'_0, C'_1, \ldots, C'_m$ from F, where either $C'_m = \Box$ or $C'_m = \{p_n\}$ (see Example 9). In the first case there is nothing more left to prove, so we consider the second case: $C'_m = \{p_n\}$. Applying similar reasoning to F_1 we can assume that there is a proof of $\{\neg p_n\}$ from F. Gluing together these two proofs and applying one more resolution step to $\{p_n\}$ and $\{\neg p_n\}$ gives a proof of $\Box$ from F. $\Box$

Example 9. We illustrate the transformation on proofs underlying the induction step in the proof of the Completeness Theorem.

Consider the formula $F = \{\{p, r\}, \{\neg p, q\}, \{\neg q, r\}\}$. We transform the following derivation of $\Box$ from $F[\mathit{false}/r]$

$$\frac{\dfrac{\{p\} \qquad \{\neg p, q\}}{\{q\}} \qquad\qquad \{\neg q\}}{\Box}$$

to the following derivation of $\{r\}$ from F:

$$\frac{\dfrac{\{p, r\} \qquad \{\neg p, q\}}{\{q, r\}} \qquad\qquad \{\neg q, r\}}{\{r\}}$$