1 Completeness and Soundness of Resolution Proofs

1.1 Definition of a Resolution Proof

Recall the resolution rule:

\[
\frac{C \cup \{x\} \quad D \cup \{\bar{x}\}}{C \cup D}.
\]

**Definition** A set of literals \(\{x_1, \ldots, x_n\}\), with \(x_i\) in \(P_k\) or \(\bar{P}_k\), is called a clause.

**Definition** Resolution refutes a set of clauses if and only all the clauses cannot be simultaneously satisfied.

A clause is a disjunction of literals and a set of clauses is a conjunction of clauses, which can be thought of as a conjunctive normal form formula. We can view resolution as proving disjunctive normal form formulas. For right now, resolution can prove tautologies that are in Disjunctive Normal Form.

**Example** The Pigeon hole principle \((PHP^m_n)\) can be written as

\[
\bigwedge_{i=1}^m \bigwedge_{j=1}^n p_{ij} \rightarrow \bigwedge_{i=1}^{m-1} \bigwedge_{j=i+1}^n \bigwedge_{k=1}^n (p_{ik} \land p_{jk}).
\]

The negation of this \((\neg PHP^m_n)\) is

\[
\bigwedge_{i=1}^m \bigwedge_{j=1}^n \overline{p_{ij}} \land \bigwedge_{i=1}^{m-1} \bigwedge_{j=i+1}^n \bigwedge_{k=1}^n (\overline{p_{ik}} \land \overline{p_{jk}}).
\]

which is in conjunctive normal form.

Written as a set of clauses:

\[
\begin{align*}
\{p_{i1}, \ldots, p_{in}\}, & \quad i = 1, \ldots, m \\
\{\overline{p_{ik}}, \overline{p_{jk}}\}, & \quad i = 1, \ldots, m - 1; \ j = i + 1, \ldots, n; \ k = 1, \ldots n
\end{align*}
\]

\(\leftarrow m\) clauses

\(\leftarrow \approx m^2\) clauses

A resolution “proof” of \(PHP\) means a refutation of this set of clauses.
1.2 Completeness Theorem

Theorem 1 (Completeness Theorem) If \( C \) is an unsatisfiable set of clauses, then \( C \) has a resolution refutation.

Proof Using induction on the number of variables in \( C \), assume \( C \) has zero variables. Then either \( C = \{\theta\} \), in which case it contains the refutation \( \theta \), or \( C = \emptyset \) which is satisfiable. Thus the hypothesis holds for any clause with zero variables.

Now, let \( C \) be an unsatisfiable set of clauses and let \( x \) be a variable in some clause in \( C \). Define

\[
C_x = \{ \text{the set of clauses in } C \text{ that contain } x \}
\]

\[
C_{\bar{x}} = \{ \text{the set of clauses in } C \text{ that contain } \bar{x} \}
\]

\[
C' = C - (C_x \cup C_{\bar{x}}).
\]

Then resolve all \( C_x \) clauses with all \( C_{\bar{x}} \) clauses by

\[
\frac{D \cup \{x\} \quad E \cup \{\bar{x}\}}{D \cup E}
\]

Let \( D = C' \cup \{ \text{all resolvents of the form } D \cup E, \text{ where } D \cup \{x\} \in C_x \text{ and } E \cup \{\bar{x}\} \in C_{\bar{x}} \} \)

Since \( D \) has fewer variables than \( C \), then by the induction hypothesis, if \( D \) is unsatisfiable, then \( D \) has a refutation. Also, from the construction of \( D \), if \( D \) has a refutation, then \( C \) has a refutation. Thus, if we can show that \( D \) is unsatisfiable, then \( C \) has a refutation.

Suppose \( D \) is satisfiable and \( \tau \) is a truth assignment that satisfies \( D \). Define \( \tau^+ \) to be the same as \( \tau \) with the addition that \( \tau(x) = T \), and define \( \tau^- \) to be the same as \( \tau \) with the addition that \( \tau(x) = F \).

Suppose \( \tau^+ \) does not satisfy \( C \). Then there is a \( E \cup \{\bar{x}\} \in C \) such that \( \tau^+ \) does not satisfy \( E \cup \{\bar{x}\} \). But then \( \tau \) does not satisfy \( E \). Similarly, if \( \tau^- \) does not satisfy \( C \), then there is a \( D \cup \{x\} \) such that \( \tau \) does not satisfy \( D \). However, since \( \tau \) satisfies \( D \), \( \tau \) satisfies \( D \cup E \). So, either \( \tau^+ \) or \( \tau^- \) satisfies \( C \).

1.3 Size of a Resolution Proof

The size of a resolution proof can be measured in two ways:

a) Total number of literals in all clauses.

b) Number of clauses.

Clearly, \((b) \leq (a) \leq (b) \cdot (\text{number of distinct variables})\), so a polynomial size bound on \( b \) implies a polynomial size bound on \( a \).

1.4 Subsumption Rule

Definition The subsumption rule (weakening rule), for any two clauses \( C \) and \( D \) with \( C \subseteq D \), is given by

\[
\frac{C}{D}.
\]
Theorem 2  A resolution and subsumption refutation of a set $C$ of clauses can be converted into a smaller resolution refutation of $C$.

In practice, a theorem prover has $C_1, \ldots, C_k$ as input clauses and generates clauses with resolution. At some point, if it has clauses $D$ and $E$ with $E \subseteq D$, then it is alright to discard $D$ without any negative consequences.

Proof  Let $\phi_1, \ldots, \phi_k = \emptyset$ be a refutation using resolution and subsumption. A new refutation $\psi_1, \ldots, \psi_k = \emptyset$, built recursively in the following way using only resolution, will have the property that $\psi_i \subseteq \phi_i$ for each $i \leq k$.

For each $i \leq k$, define $\psi_i$ as follows:

1) If $\phi_i \in C$, then set $\psi_i = \phi_i$. In this case, clearly $\psi_i \subseteq \phi_i$.

2) If $\phi_i$ is inferred by subsumption $\frac{\phi_l}{\phi_i}$ for some $l \leq i$, with $\phi_l \subseteq \phi_i$, then set $\psi_i = \psi_l$. Here, we have $\psi_i = \psi_l \subseteq \phi_l \subseteq \phi_i$.

3) If $\phi_i$ is inferred by resolution, for some $j, l \leq i$,

$$
\frac{\phi_j \phi_l}{\phi_i}
$$

resolving on $x \in \phi_j$ and $\bar{x} \in \phi_l$, do the following:

a) If $x \not\in \psi_j$, set $\psi_i = \psi_j \subseteq \phi_i$.

b) If $\bar{x} \not\in \psi_l$, set $\psi_i = \psi_l \subseteq \phi_i$.

c) Otherwise, set $\psi_i = \text{res}_x(\psi_j, \psi_l)$, where $\text{res}_x$ is defined to be the resolvent obtained by the resolution using the literal $x$. Since $\psi_j \subseteq \phi_j$ and $\psi_l \subseteq \phi_l$, then $\psi_i \subseteq \phi_i$.

Clearly, $\psi_k = \emptyset$, since $\psi_k \subseteq \phi_k = \emptyset$. Finally, erase any duplicate $\psi_i$’s.

1.5 Refutation Proof of the Pigeon Hole Principle

As a point of notation, throughout this proof, we will use $[k]$ to denote the set $\{1, \ldots, k\}$.

Recall that the negation of the Pigeon Hole Principle can be written as:

$$
\bigcap_{i=1}^{m} \bigcup_{j=1}^{n} p_{ij} \land \bigcap_{i=1}^{m-1} \bigcap_{j=i+1}^{n} (p_{ik} \land p_{jk}).
$$

For this proof, we will prove the special case $PHP_{n+1}^n$ (i.e. $m = n + 1$). Writing this as a set of clauses, we get

$$
C = \{\{P_{i,1}, \ldots, P_{i,n}\}, 1 \leq i \leq n\} \cup \{\{\bar{P}_{i,k}, \bar{P}_{j,k}\}, 1 \leq i \leq j \leq m; 1 \leq k \leq n\}
$$

Proof  The refutation will proceed in a series of stages, $s = n, n-1, \ldots, 0$. At stage $s$, we have the following clauses: For each injective map $\pi : \{1, \ldots, s\} \to \{1, \ldots, n\}$ we have the clause $\{\bar{P}_{1,\pi(1)}, \bar{P}_{2,\pi(2)}, \ldots, \bar{P}_{s,\pi(s)}\}$.

At stage $s = 0$, the only map is $\pi : \emptyset \to [n]$ and the clause is $\emptyset$. 

3
At stage $s = n$, for any injective map $\pi : [n] \to [n]$, start with the initial clause $\{P_{n+1,1}, \ldots, P_{n+1,n}\}$ and resolve with the initial clauses $\{\bar{P}_{\pi(i)}, \bar{P}_{n+1,\pi(i)}\}$ for each $1 \leq i \leq n$.

For the induction step, assume we have the stage $s + 1$ clauses. Given any injective map $\pi : [s] \to [n]$ we need to derive $\{\bar{P}_{\pi(1)}, \bar{P}_{\pi(2)}, \ldots, \bar{P}_{\pi(s)}\}$. For $j \not\in \text{Range}(\pi)$, define $\pi_j$ to be $\pi \cup \{(s + 1) \mapsto j\}$. Since $\pi_j : [s + 1] \to [n]$, then from stage $s + 1$ we already have

$$(**_j) \quad \{\bar{P}_{\pi(1)}, \bar{P}_{\pi(2)}, \ldots, \bar{P}_{\pi(s)}, \bar{P}_{s+1,j}\}.$$ 

To derive the stage $s$ clauses, start with the initial clause $\{P_{s+1,1}, \ldots, P_{s+1,n}\}$ and resolve with the initial clauses $\{P_{\pi(i)}, \bar{P}_{s+1,\pi(i)}\}$ for each $1 \leq i \leq s$. After resolving with each of the $s$ clauses, we get

$$\{\bar{P}_{\pi(1)}, \bar{P}_{\pi(2)}, \ldots, \bar{P}_{\pi(s)}, P_{s+1,j_1}, \ldots, P_{s+1,j_{n-s}}\}$$

where $[n] - \text{Range}(\pi) = \{j_1, \ldots, j_{n-s}\}$. Finally, resolve with the $(**_j)$ clauses for $j = j_1, \ldots, j_{n-s}$ and we get $\{\bar{P}_{\pi(1)}, \bar{P}_{\pi(2)}, \ldots, \bar{P}_{s+1,\pi(s)}\}$ as desired.

### 1.6 Size of Proof of Pigeon Hole Principle

There are $n$ stages for this proof of $\text{PHP}_n^{n+1}$. At each stage, there are on the order of $O(n^s)$ injective maps $\pi : [s] \to [n]$. Also, there are $n$ steps required to derive each clause. Thus, the size of this proof is on the order of $O(n \cdot n^s) = 2^{O(n \log n)}$ total number of clauses.

However, a more honest measure of the size of the proof is in terms of the number of variables $v = \Omega(n^2)$. In terms of $v$, the size of the proof is on the order $2^{O(\sqrt{\log V})} = 2^{O(\sqrt{\log V})}$.

### 1.7 Soundness Theorem

**Theorem 3** *(Soundness Theorem)* If $\mathcal{C}$ is a set of clauses with a refutation, then $\mathcal{C}$ is unsatisfiable.

**Proof** Proof of the soundness theorem is deferred until the next lecture.