3.5 Restrictions of Resolution

Plan:

3.5.1. Linear Resolution

3.5.2. Input Resolution

3.5.1. Linear Resolution

**Def 3.5.1 (Linear Resolution)**

Let \( \mathcal{K} \) be a clause set.

\( \beta \) is derivable from a clause \( \kappa \) in \( \mathcal{K} \) by linear resolution iff there exists a sequence \( \kappa_1, \ldots, \kappa_m \) with

\[ \kappa_1 = \kappa \in \mathcal{K} \text{ and } \kappa_m = \beta \]

and for all \( 2 \leq i \leq m \) we have:

\( \kappa_i \) is resolvent of \( \kappa_{i-1} \) and a clause from \( \{ \kappa_1, \ldots, \kappa_{i-1} \} \cup \mathcal{K} \).

**Ex. 3.5.2**

\( \{ p, q \} \)

\( \{ \neg p, q \} \)

\( \{ p, \neg q \} \)

\( \{ \neg p, \neg q \} \)

is not

\( \{ q \} \)

\( \{ \neg q \} \)
is not a linear resolution proof

\[ \{ q, \neg q \} \quad \neg \{ q, \neg q \} \]

**Wrong!**

\[ \{ p, q \} \quad \neg \{ p, q \} \]

**Correct**

\[ \{ q, \neg q \} \]

Can we also obtain \( \square \) by linear resolution?

\[ \{ p, q \} \quad \neg \{ p, q \} \]

\[ \{ q \} \]

\[ \{ p \} \quad \neg \{ p \} \]

\[ \{ q \} \]

This is a linear resolution proof.
Thm 3.5.3 (Soundness + Completeness of Linear Resolution)

Let \( \mathcal{K} \) be a clause set. Then \( \mathcal{K} \) is unsatisfiable iff \( \Box \) can be derived by linear resolution from some \( \mathcal{K} \in \mathcal{K} \).

If \( \mathcal{K} \) is a minimal unsatisfiable clause set, i.e., every \( \mathcal{K}' \subseteq \mathcal{K} \) is satisfiable, then \( \Box \) can be derived by linear resolution from every \( \mathcal{K} \in \mathcal{K} \).

Proof: Soundness follows from Thm 3.4.10 (soundness of full resolution), because every linear resolution step is a resolution step.

For completeness, one first shows completeness of linear ground resolution. Then the lifting lemma is used to prove completeness of linear res. in predicate logic. (Course Notes)

3.5.2. Input Resolution and SLD Resolution

Idea: Restrict linear resolution further. The second parent clause should come from the original input clause set \( \mathcal{K} \).
Def 3.54 (Input Resolution)
Let $K$ be a clause set. Then $\Box$ can be derived from $K \in K$ by input resolution iff there is a sequence $K_1, \ldots, K_m$ where $K_1 = K \in K$, $K_m = \Box$, and for all $2 \leq i \leq m$ we have:

$K_i$ is a resolvent of $K_{i-1}$ and a clause from $K$.

Ex. 3.55 Is input resolution still complete?

\[
\begin{align*}
\{p, q\} & \quad \{\neg p, q\} & \quad \{p, \neg q\} & \quad \{\neg p, \neg q\}
\end{align*}
\]
\[
\begin{align*}
\{q\} & \quad \{\neg q\}
\end{align*}
\]

We cannot derive $\Box$ by input resolution.

Thus: Input resolution is sound, but not complete.

Solution: Do not regard arbitrary clauses anymore, just regard Horn clauses.
It will turn out that on Horn clauses, input resolution is complete.

**Def 356 (Horn Clause)**

A clause $K$ is a **Horn clause** iff it contains at most one positive literal (i.e., at most one of its literals is an atomic formula and the other literals are negated atomic formulas).

A Horn clause is **negative** iff it only contains negative literals, i.e., it has the form $\{\neg A_1, \ldots, \neg A_n\}$ for atomic formulas $A_1, \ldots, A_n$.

A Horn clause is **definite** iff it contains a positive literal, i.e., it has the form $\{B, \neg C_1, \ldots, \neg C_n\}$ for atomic formulas $B, C_1, \ldots, C_n$.

A set of definite Horn clauses corresponds to a conjunction of implications:

$\{\{p, \neg q\}, \{\neg x, \neg p, s\}, \{s\}\}$

is equivalent to

$$(p \lor \neg q) \land (\neg x \lor \neg p \lor s) \land s$$

which is equivalent to

$$(q \implies p) \land (x \lor p \implies s) \land s$$
Connection to Logic Programming:

- **Facts**  
  \[ s \]
  \( \equiv \) definite Horn clause \( \{ s \} \) without negative literals

- **Rules**  
  \[ s \leftarrow r, p. \quad (\equiv (r \lor p) \rightarrow s \equiv \neg (r \lor p) \lor s) \equiv \]  
  \( \equiv \) definite Horn clause \( \{ s, \neg r, \neg p \} \) with negative literals.

\[ s \leftarrow \neg r \quad \] not allowed in
\( \{ s, r \} \quad \) pure logic programming.

Prolog has a form of negation \( \Rightarrow \) Chapter 5

Even with pure logic programming, one can have infinitely many answers:

\[
\begin{align*}
\text{nat}(0), \\
\text{nat}(s(X)) & \leftarrow \neg \text{nat}(X).
\end{align*}
\]

\[
\begin{align*}
? - \neg \text{nat}(Y). & \quad \uparrow \\
\{ \text{nat}(s(X)), \neg \text{nat}(X) \}
\end{align*}
\]

\[
\begin{align*}
Y & \equiv 0; \\
Y & \equiv s(0); \\
Y & \equiv s(s(0));
\end{align*}
\]

- **Queries**  
  \[ ? - p, q. \quad " \text{Does } p \land q \text{ hold?}" \]
Negation Yields \( T \lor \neg T \)

Corresponds to negative Horn clause \( \{ T, \neg T \} \)

Restriction to Horn clauses affects the expressivity: \( \{ p, \neg p \} \) has no equivalent Horn clause.

But this restriction improves efficiency:

- Propositional logic:
  - Satisfiability in general: decidable, NP-complete
  - Satisfiability for Horn clauses: polynomial time

- Pre. Logic
  - Satisfiability in general: undecidable, input res. incomplete
  - Satisfiability for Horn clauses: undecidability, input res. complete

If we restrict ourselves to Horn clauses, then input resolution can be improved further to SLD resolution. We will prove that SLD resolution is complete on Horn clauses (which implies that input resol. is complete on Horn clauses).

Idea: Input resolution proofs should start
with a negative Horn clause.

**Def 3.5.7** (SLD Resolution)

Let $X$ be a set of Horn clauses with $X = X^d \cup X^n$ where $X^d$ contains the definite Horn clauses from $X$ and $X^n$ contains the negative Horn clauses from $X$. Then $\bar{O}$ can be derived from $X$ if $X^n$ by SLD resolution if and there is a seq. $K_1, \ldots, K_m$ with $K_n = \emptyset \in X^n$, $K_m = \bar{O}$, and for all $2 \leq i \leq m$ we have:

$K_i$ is a resolvent of $K_{i-1}$ and a clause from $X^d$.

So SLD resolution proofs have the following form:

$$
\begin{array}{c}
\text{N} \\
\text{K}_1 \quad \text{K}_2 \quad \ldots \quad \text{K}_n \\
\text{R}_1 \\
\text{R}_2 \\
\ldots \\
\text{R}_n
\end{array}
$$

These are all negative Horn clauses.

A negative Horn clause can only be resolved with a definite Horn clause.

"SLD": Linear resolution with selection function.
for definite clauses

Thm 3.5.8 (Soundness and Completeness of SLD Resolution)

Let \( K \) be a set of Horn clauses.
Then \( K \) is unsatisfiable iff \( \bot \) can be derived from some negative clause \( N \in K \) by SLD resolution.

Proof: Soundness (\( \leq \)) follows from Thm 3.4.10 (soundness of resolution).
Completeness (\( \Rightarrow \)): Let \( K_{\text{min}} \) be a minimal unsatisfiable subset of \( K \). Every set of definite Horn clauses is satisfiable (by the interpretation that satisfies all atomic formulas). Thus, \( K_{\text{min}} \) contains a negative Horn clause \( N \).

By Thm 3.5.3 (Completeness of linear resolution), \( \bot \) can be derived by linear resolution from every clause of \( K_{\text{min}} \). Hence, there is a linear resolution proof of \( \bot \) that starts with \( N \).

- There cannot be a resolution step with 2 negative parent clauses
- If one performs resolution with a negative and a definite clause,
then the resolvent is a negative clause.

\[ \Rightarrow \text{This linear resolution proof of } \square \text{ is an SLD proof.} \]

Logic programming uses one more restriction.

Instead of resolving \( L_1, \ldots, L_m \) from \( \forall (L_1) \)
and \( L_1', \ldots, L_n' \) from \( \forall (L_2) \),

one fixes \( m = n = 1 \).

This is called **binary resolution**.

Binary resolution is not complete on arbitrary clauses!

**Ex. 3.5.9**

\[ \mathcal{X} = \{ \{ p(X), p(Y) \}, \{ \neg p(U), \neg p(V) \} \} \]

because

\[ \{ p(X), p(Y), p(U), p(V) \} \]

are unifiable by

unify \{ X/V, Y/V, U/V \}.

Binary resolution:

\[ \_ \_ \_ (V) (V) \_ \_ \_ \_ \_ \_ \_ \_ \]
One will never read □.

However, binary resolution is complete on Horn clauses.

**Theorem 3.5.10. (Soundness and Completeness of binary SLD Resolution)**

Let \( \mathcal{H} \) be a set of Horn clauses.

Then \( \mathcal{H} \) is unsatisfiable iff □ can be derived from a negative clause \( N \in \mathcal{H} \) by binary SLD resolution.

**Proof:** Course Notes.