Horn Clauses and SLD Resolution

Marco Piastra
Horn Clauses (in $L_p$)

- **Definition**

  A *Horn Clause* is a wff in CF that contains at most one literal in positive form.

- **Three types of Horn Clauses:**
  
  **Rule:** two or more literals, one positive
  
  Examples: $\{B, \neg D, \neg A, \neg C\}, \{A, \neg B\}$ (equivalent to: $(D \land A \land C) \rightarrow B, \ B \rightarrow A$)

  **Facts:** just one positive literal
  
  Examples: $\{B\}, \{A\}$

  **Goal:** one or more literals, all negative
  
  Examples: $\{\neg B\}, \{\neg A, \neg B\}$

  More terminology:

  Rules and facts are also called *definite clauses*
  
  Goals are also called *negative clauses*
Lost in Translation…

Many wffs can be translated into Horn clauses:

\[(A \land B) \rightarrow C\]
\[-(A \land B) \lor C\]
\[-A \lor \neg B \lor C\]  
(rewriting \(\rightarrow\))

\[(\neg A \lor B) \land (\neg A \lor C)\]
\[\neg A \lor B, (\neg A \lor C)\]  
(De Morgan - CF – it is a rule)

\[A \rightarrow (B \land C)\]
\[\neg A \lor (B \land C)\]
\[(\neg A \lor B) \land (\neg A \lor C)\]
\[\neg A \lor B, (\neg A \lor C)\]  
(rewriting \(\rightarrow\))

\[A \rightarrow (B \lor C)\]
\[\neg (A \lor B) \lor C\]
\[\neg A \lor B, \neg C\]
\[\neg (A \lor B) \land (\neg A \lor C)\]
\[\neg A \lor B, (\neg A \lor C)\]  
(distributing \(\lor\))

\[A \rightarrow (B \lor C)\]
\[\neg (A \lor B) \lor C\]
\[\neg A \lor B \lor C\]  
(rewriting \(\rightarrow\))

\[A \rightarrow (B \lor C)\]
\[\neg A \lor B \lor C\]  
(De Morgan)

\[A \land \neg B \rightarrow C\]
\[\neg (A \land \neg B) \lor C\]
\[\neg A \lor B \lor C\]  
(rewriting \(\rightarrow\))

\[A \rightarrow (B \lor C)\]
\[\neg A \lor B \lor C\]  
(De Morgan)

But not all of them:

\[A \rightarrow (B \lor C)\]
\[\neg A \lor B \lor C\]  
(rewriting \(\rightarrow\))
### SLD Resolution (in $L_P$)

*Linear resolution with Selection function for Definite clauses*

#### Algorithm

- Starts from a set of *definite clauses* (also the *program*) + a *goal*
- 1) At each step, the *selection function* identifies a *literal* in the *goal* (i.e. *subgoal*)
- 2) All *definite clause* applicable to the *subgoal* is selected
- 3) The resolution rule is applied generating the *resolvent*

Termination: either the empty clause { } is obtained or step 2) fails.

#### Example:

*Selection function: leftmost subgoal first*

*Definite clauses: { C }, { D }, { B, ¬D }, { A, ¬B, ¬C }*

*Goal: { ¬A }*

---

Resolution and Horn clauses [4]
SLD trees (in $L_p$)

SLD derivations

Example: \{ C \}, \{ D \}, \{ B, \neg D \}, \{ A, \neg B, \neg C \} goal \{ \neg A \}

*In this example each subgoal can be resolved in one mode only*

*This is not true in general*

---

- **SLD trees** (= trace of all SLD derivations from a goal)

  Example: \{ C \}, \{ D \}, \{ B, \neg F \}, \{ B, \neg E \}, \{ B, \neg D \}, \{ A, \neg B, \neg C \} goal \{ \neg A \}

  A few new rules have been added: there are now different possibilities

\[
\begin{align*}
\{ \neg A \} \\
\{ \neg B, \neg C \} \\
\{ \neg F, \neg C \} \\
\{ \neg E, \neg C \} \\
\{ \neg D, \neg C \} \\
\{ \neg C \} \\
\{ \} \\
\end{align*}
\]

*Selection function: leftmost subgoal first*

Each branch correspond to a possible resolution for a subgoal
**SLD Resolution (in \( L_p \))**

- **A resolution method for Horn clauses in \( L_p \)**
  
  It always terminates

  It is *correct*: \( \Gamma \vdash \varphi \Rightarrow \Gamma \models \varphi \)

  It is *complete*: \( \Gamma \models \varphi \Rightarrow \Gamma \vdash \varphi \)

- **Computationally efficient**

  It has polynomial time complexity (w.r.t the # of propositional symbols occurring in \( \Gamma \) and \( \varphi \))

- **Limitations**

  Not all problems can be translated into Horn clauses

  The “Harry is happy” problem does not translate

  \( \Gamma \) : only a set of *rules* and *facts*

  \( \varphi \) : only a conjunction of *facts*
Horn Clauses in $L_{FO}$

The definition is very similar to the propositional case

- **Horn Clauses** (of the skolemization of a set *sentences*)
  Each clause contains at most one literal in positive form

**Facts, rules and goals**

**Fact:** a clause with just an individual *atom*

{\textit{Human(socrates)}}, {\textit{Pyramid(x)}}, {\textit{Sister(sally, motherOf(paul))}}

**Rule:** a clause with at least two literals, exactly one in positive form

{\textit{Human(x), ¬Philosopher(x)}},
\forall x (\textit{Philosopher(x) → Human(x)})

{\textit{¬Female(x), ¬Parent(k(x),x), ¬Parent(k(y),y)}, \textit{Sister(x,y)}}
\forall x \forall y ((\textit{Female(x)} \land \exists z (\textit{Parent(z,x) ∧ Parent(z,y)})) \rightarrow \textit{Sister(x,y)})

{\textit{¬Above(x,y), On(x,k(x))}}, {\textit{¬Above(x,y), On(j(y),y)}}
\forall x \forall y (\textit{Above(x,y) → (∃z On(x,z) ∧ ∃v On(v,y))})

**Goal:** a clause containing negative literals only

{\textit{¬Human(socrates)}}

{\textit{¬Sister(sally,x), ¬Sister(x,paul)}}

Negation of  ∃x (\textit{Sorella(sally,x) ∧ Sorella(x,paul)})
SLD Resolution in $L_{FO}$

*Linear resolution with Selection function for Definite clauses*

- **Description**

  Program (a set of *definite clauses*: rules + facts):
  - Rule: $\beta \lor \neg \gamma_1 \lor \neg \gamma_2 \lor \ldots \lor \neg \gamma_n$
  - Fact: $\delta$

  Goal (a conjunction of facts in negated form):
  - Goal: $\neg \alpha_1 \lor \neg \alpha_2 \lor \ldots \lor \neg \alpha_k$

  Procedure:
  - Starting point: a program $\Pi$ and a goal $\phi$
  - The subgoals are considered according to the *selection function* of choice
  - For each subgoal $\neg \alpha_i$ the resolution (with unification) is attempted with all rules and facts in $\Pi$ whose positive literal is compatible
Example:

\[ \Pi \equiv \{ \{ \text{Human}(x), \neg \text{Philosopher}(x) \}, \{ \text{Mortal}(y), \neg \text{Human}(y) \}, \{ \text{Philosopher}(\text{socrates}) \}, \{ \text{Philosopher}(\text{plato}) \}, \{ \text{Philosopher}(\text{aristotle}) \} \} \]

\[ \text{goal} \equiv \{ \neg \text{Mortal}(x), \neg \text{Human}(x) \} \]

"Is there anyone who is both human and mortal?"

\[ \text{goal 1: } \neg \text{Mortal}(x) \]

\[ \{ \neg \text{Mortal}(x) \}, \{ \text{Mortal}(y), \neg \text{Human}(y) \} \]

\[ \text{goal 2: } \{ \neg \text{Human}(y) \} \]

\[ \{ \neg \text{Human}(y) \}, \{ \text{Human}(x), \neg \text{Philosopher}(x) \} \]

\[ \{ \neg \text{Philosopher}(x) \} \]

\[ \{ \text{Philosopher}(\text{socrates}) \} \]

\[ \{ \text{Philosopher}(\text{plato}) \} \]

\[ \{ \text{Philosopher}(\text{aristotle}) \} \]
Another example

\[ \Pi \equiv \{ \{ \text{Human}(x), \neg \text{Philosopher}(x) \}, \{ \text{Mortal}(y), \neg \text{Human}(y) \}, \{ \text{Philosopher}(\text{socrates}) \}, \{ \text{Philosopher}(\text{plato}) \}, \{ \text{Mortal}(\text{felix}) \} \} \]

\[ \text{goal} \equiv \{ \neg \text{Mortal}(x), \neg \text{Human}(x) \} \]

"Is there anyone who is both human and mortal?"

\[ \text{goal 1: } \neg \text{Mortal}(x) \quad [] \]

\[ \{ \neg \text{Mortal}(x) \}, \{ \text{Mortal}(y_1), \neg \text{Human}(y_1) \} \quad [] \]

\[ \text{goal 2: } \neg \text{Human}(y_1) \quad [x/y_1] \]

\[ \{ \neg \text{Human}(y_1) \}, \{ \text{Human}(x_1), \neg \text{Philosopher}(x_1) \} \quad [x/y_1] \]

\[ \{ \neg \text{Philosopher}(x_1) \} \quad [x/y_1][y_1/x_1] \]

\[ \{ \neg \text{Philosopher}(x_1) \} \quad \{ \text{Philosopher}(\text{socrates}) \} \quad [x/y_1][y_1/x_1][x_1/\text{socrates}] \]

\[ \{ \neg \text{Philosopher}(x_1) \} \quad \{ \text{Philosopher}(\text{plato}) \} \quad [x/y_1][y_1/x_1][x_1/\text{plato}] \]

\[ \{ \neg \text{Mortal}(x) \}, \{ \text{Mortal}(\text{felix}) \} \quad [] \]

\[ \text{goal 2 cannot be resolved} \quad (\text{due to } [x/\text{felix}]) \]

\[ \{ \neg \text{Human}(y_1) \} \quad [x/\text{felix}] \]

\[ \{ \neg \text{Mortal}(x) \}, \{ \text{Mortal}(\text{felix}) \} \quad [] \]

\[ \{ \neg \text{Philosopher}(x_1) \} \quad \{ \text{Philosopher}(\text{socrates}) \} \quad [x/y_1][y_1/x_1][x_1/\text{socrates}] \]

\[ \{ \neg \text{Philosopher}(x_1) \} \quad \{ \text{Philosopher}(\text{plato}) \} \quad [x/y_1][y_1/x_1][x_1/\text{plato}] \]
Infinite SLD Trees

- A first example:

\[ \Pi \equiv \{ \{ P(x), \neg P(x) \} \} \]
\[ \neg \phi \equiv \{ \neg P(x) \} \]

\[
\text{goal: } \neg P(x) \]
\{ \neg P(x) \}, \{ P(x_1), \neg P(x_1) \} \]
\{ \neg P(x_1) \} [x/x_1] \]
\{ \neg P(x_1) \}, \{ P(x_2), \neg P(x_2) \} [x/x_1] \]
\{ \neg P(x_2) \} [x/x_1] [x_1/x_2] \]

Since \( \Pi \not\models \phi \), the method can **divege** (and it does…)

---

*Artificial Intelligence – A.A. 2012–2013*
Infinite SLD Trees

- A second example:
  \[
  \Pi \equiv \{\{P(x), \neg P(x)\}, \{P(a)\}\}
  \]
  \[
  \neg \phi \equiv \{\neg P(x)\}
  \]

\[
\begin{align*}
\text{goal:} & \quad \neg P(x) \,
\mid \\
\{ \neg P(x) \}, \{ P(x_1), \neg P(x_1) \} \,
\mid \\
\{ \neg P(x_1) \} \,[x/x_1] \,
\mid \\
\{ \neg P(x_1) \}, \{ P(x_2), \neg P(x_2) \} \,[x/x_1] \,
\mid \\
\{ \neg P(x_2) \} \,[x/x_1] \,[x_1/x_2] \\
& \quad \ldots
\end{align*}
\]

In this case \( \Pi \models \phi \), so the method should not diverge.
However, when a \textit{depth-first} selection function is used, the infinite branch
in the SLD-tree makes the method diverge anyway.

A \textit{fair} selection function is such that no possible resolution will be postponed
indefinitely: that is, \textit{any} possible resolution will be performed, eventually.