Definite Logic Programs: Models
Logical Consequences of Formulae

- Recall: F is a logical consequence of P (i.e. \(P \models F\)) iff

  Every model of P is also a model of F.

- Since there are (in general) infinitely many possible interpretations, how can we check if F is a logical consequence of P?

- Solution: choose one "canonical" model I such that

  \(I \models P \quad \text{and} \quad I \models F \Rightarrow P \models F\)
Definite Clauses

- A formula of the form $p(t_1, t_2, \ldots, t_n)$, where $p/n$ is an $n$-ary predicate symbol and $t_i$ are all terms is said to be atomic.

- If $A$ is an atomic formula then
  - $A$ is said to be a positive literal
  - $\neg A$ is said to be a negative literal

- A formula of the form $\forall (L_1 \lor L_2 \lor \ldots \lor L_n)$ where each $L_i$ is a literal (negative or positive) is called a clause.

- A clause $\forall (L_1 \lor L_2 \lor \ldots \lor L_n)$ where exactly one literal is positive is called a definite clause (also called Horn clause).

- A definite clause is usually written as:
  - $\forall (A_0 \lor \neg A_1 \lor \ldots \lor \neg A_n)$
  - or equivalently as $A_0 \leftarrow A_1, A_2, \ldots, A_n$.

- A definite program is a set of definite clauses.
Herbrand Universe

• Given an alphabet $A$, the set of all ground terms constructed from the constant and function symbols of $A$ is called the Herbrand Universe of $A$ (denoted by $U_A$).

• Consider the program:

\[
p(zero).
\]
\[
p(s(s(X))) \leftarrow p(X).
\]

• The Herbrand Universe of the program's alphabet is: $U_A = \{zero, s(zero), s(s(zero)), \ldots\}$
Herbrand Universe: Example

- Consider the "relations" program:

  \[
  \begin{align*}
  &\text{parent}(\text{pam}, \text{bob}). & \text{parent}(\text{bob}, \text{ann}). \\
  &\text{parent}(\text{tom}, \text{bob}). & \text{parent}(\text{bob}, \text{pat}). \\
  &\text{parent}(\text{tom}, \text{liz}). & \text{parent}(\text{pat}, \text{jim}). \\
  &\text{grandparent}(X,Y) : - \\
   &\hspace{1cm} \text{parent}(X,Z), \text{parent}(Z,Y).
  \end{align*}
  \]

- The Herbrand Universe of the program's alphabet is:

  \[U_A = \{\text{pam, bob, tom, liz, ann, pat, jim}\}\]
Herbrand Base

• Given an alphabet $A$, the set of all **ground atomic formulas** over $A$ is called the **Herbrand Base** of $A$ (denoted by $B_A$).

• Consider the program:

  $p(\text{zero})$.

  $p(s(s(X))) \leftarrow p(X)$.

• The Herbrand Base of the program's alphabet is: $B_A = \{ p(\text{zero}), p(s(\text{zero})), p(s(s(\text{zero}))), \ldots \}$
Consider the "relations" program:

parent(pam, bob).
parent(bob, ann).
parent(tom, bob).
parent(bob, pat).
parent(tom, liz).
parent(pat, jim).

grandparent(X, Y) :-
    parent(X, Z), parent(Z, Y).

The Herbrand Base of the program's alphabet is:

\[ B_A = \{ parent(pam, pam), parent(pam, bob), parent(pam, tom), \ldots, parent(bob, pam), \ldots, grandparent(pam, pam), \ldots, grandparent(bob, pam), \ldots \} \].
Herbrand Interpretations and Models

- A **Herbrand Interpretation** of a program $P$ is an interpretation $I$ such that:
  - The domain of the interpretation: $|I| = U_P$
  - For every constant $c$: $c_I = c$
  - For every function symbol $f/n$: $f_I(x_1, \ldots, x_n) = f(x_1, \ldots, x_n)$
  - For every predicate symbol $p/n$: $p_I \subseteq (U_P)^n$ (i.e. some subset of $n$-tuples of ground terms)

- A **Herbrand Model** of a program $P$ is a Herbrand interpretation that is a model of $P$. 
Herbrand Models

• All Herbrand interpretations of a program give the same “meaning” to the constant and function symbols.
• Different Herbrand interpretations differ only in the “meaning” they give to the predicate symbols.
• We often write a Herbrand model simply by listing the subset of the Herbrand base that is true in the model.
• Example: Consider our numbers program, where 
\{p(zero), \ p(s(s(zero))), \ p(s(s(s(s(zero))))), ...\}
represents the Herbrand model that treats 
\p_I=\{zero, s(s(zero)), s(s(s(s(zero)))), ...\}
as the meaning of \(p\).
Sufficiency of Herbrand Models

• Let $P$ be a definite program. If $I'$ is a model of $P$ then $I = \{A \in Bp \mid I' \models A\}$ is a Herbrand model of $P$.

Proof (by contradiction):

Let $I$ be a Herbrand interpretation.

Assume that $I'$ is a model of $P$ but $I$ is not a model.

Then there is some ground instance of a clause in $P$:

$$A_0 :\neg A_1, \ldots, A_n.$$ 

which is not true in $I$ i.e., $I \models A_1, \ldots, I \models A_n$ but $I \not\models A_0$

By definition of $I$ then, $I' \models A_1, \ldots, I' \models A_n$ but $I' \not\models A_0$

Thus, $I'$ is not a model of $P$, which contradicts our earlier assumption.
Definite programs only

- Let P be a definite program. If I' is a model of P then 
  \[ I = \{ A \in B_p \mid I' \models A \} \]
  is a Herbrand model of P.
  
  **This property holds only for definite programs!**

- Consider \( P = \{ \neg p(a), \exists X . p(X) \} \)
  
  - There are two Herbrand interpretations:
    \( I_1 = \{ p(a) \} \) and \( I_2 = \{ \} \)
      - The first is not a model of P since \( I_1 \not\models \neg p(a) \).
      - The second is not a model of P since \( I_2 \not\models \exists X . p(X) \).

- But there is a non-Herbrand model I:
  - \( | I | = \mathbb{N} \), the set of natural numbers
  - \( a_1 = 0 \)
  - \( p_1 = \text{“is odd”} \)
Properties of Herbrand Models

1) If $M$ is a set of Herbrand Models of a definite program $P$, then $\bigcap M$ is also a Herbrand Model of $P$.

2) For every definite program $P$ there is a unique least model $M_p$ such that:
   - $M_p$ is a Herbrand Model of $P$ and,
   - for every Herbrand Model $M$, $M_p \subseteq M$.

3) For any definite program, if every Herbrand Model of $P$ is also a Herbrand Model of $F$, then $P \models F$.

4) $M_p = \text{the set of all ground logical consequences of } P$. 

Properties of Herbrand Models

- If \( M_1 \) and \( M_2 \) are Herbrand models of \( P \), then \( M = M_1 \cap M_2 \) is a model of \( P \).
- Assume \( M \) is not a model.
- Then there is some clause \( A_0 : \neg A_1, \ldots, A_n \) such that \( M \models A_1, \ldots, M \models A_n \) but \( M \not\models A_0 \).
- Which means \( A_0 \notin M_1 \) or \( A_0 \notin M_2 \).
- But \( A_1, \ldots, A_n \in M_1 \) as well as \( M_2 \).
- Hence one of \( M_1 \) or \( M_2 \) is not a model.
Properties of Herbrand Models

- There is a unique least Herbrand model
- Let $M_1$ and $M_2$ are two incomparable minimal Herbrand models, i.e., $M = M_1 \cap M_2$ is also a Herbrand model (previous theorem), and $M \subseteq M_1$ and $M \subseteq M_2$
- Thus $M_1$ and $M_2$ are not minimal.
Least Herbrand Model

- The **least Herbrand model** $M_p$ of a definite program $P$ is the set of all ground logical consequences of the program.

$$M_p = \{ A \in B_p \mid P \models A \}$$

- First, $M_p \supseteq \{ A \in B_p \mid P \models A \}$:
  - By definition of logical consequence, $P \models A$ means that $A$ has to be in every model of $P$ and hence also in the least Herbrand model.
Least Herbrand Model

- Second, $M_p \subseteq \{ A \in B_p \mid P \models A \}$:
  - If $M_p \models A$ then $A$ is in every Herbrand model of $P$.
  - But assume there is some model $I' \models \neg A$.
  - By sufficiency of Herbrand models, there is some Herbrand model $I$ such that $I \models \neg A$.
  - Hence $A$ is not in some Herbrand model, and hence is not in $M_p$. 
Finding the Least Herbrand Model

• **Immediate consequence operator:**
  - Given $I \subseteq Bp$, construct $I'$ such that
    
    $I' = \{ A_0 \in Bp \mid A_0 \leftarrow A_1, \ldots, A_n \text{ is a ground instance of a clause in } P \text{ and } A_1, \ldots, A_n \in I \}$
  - $I'$ is said to be the immediate consequence of $I$.
  - Written as $I' = Tp(I)$, $Tp$ is called the *immediate consequence operator*.

• Consider the sequence:
  
  $\emptyset, Tp(\emptyset), Tp(Tp(\emptyset)), \ldots, Tp^i(\emptyset), \ldots$
  
  $Mp \supseteq Tp^i(\emptyset)$ for all $i$.

• Let $Tp \uparrow \omega = \bigcup_{i=0,\infty} Tp^i(\emptyset)$

• Then $Mp \subseteq Tp \uparrow \omega$
Computing Least Herbrand Models: An Example

\begin{tabular}{|l|l|}
\hline
\textbf{parent}(pam, bob). & \textbf{M}_1 = \emptyset \\
\textbf{parent}(tom, bob). & \textbf{M}_2 = T_P(\textbf{M}_1) = \{\text{parent}(pam, bob), \\
\textbf{parent}(tom, liz). & \text{parent}(tom, bob), \\
\textbf{parent}(bob, ann). & \text{parent}(tom, liz), \\
\textbf{parent}(bob, pat). & \text{parent}(bob, ann), \\
\textbf{parent}(pat, jim). & \text{parent}(bob, pat), \\
\textbf{anc}(X,Y) :- & \text{parent}(pat, jim) \} \\
\text{parent}(X,Y). & \textbf{M}_3 = T_P(\textbf{M}_2) = \{\text{anc}(pam, bob), \\
\textbf{anc}(X,Y) :- & \text{anc}(tom, bob), \\
\text{parent}(X,Z), & \text{anc}(tom, liz), \\
\text{anc}(Z,Y). & \text{anc}(bob, ann), \\
\hline
\end{tabular}

\begin{tabular}{|l|l|}
\hline
\textbf{anc}(X,Y) :- & \text{anc}(bob, pat), \\
\text{parent}(X,Z), & \text{anc}(pat, jim) \} \\
\text{anc}(Z,Y). & \textbf{M}_4 = T_P(\textbf{M}_3) = \{\text{anc}(pam, ann), \\
\hline
\textbf{anc}(X,Y) :- & \text{anc}(pam, pat), \\
\text{parent}(X,Z), & \text{anc}(tom, ann), \\
\text{anc}(Z,Y). & \text{anc}(tom, pat), \\
\hline
\end{tabular}

\begin{tabular}{|l|l|}
\hline
\textbf{anc}(X,Y) :- & \text{anc}(bob, jim) \} \cup \textbf{M}_2 \\
\text{parent}(X,Z), & \textbf{M}_5 = T_P(\textbf{M}_4) = \{\text{anc}(pam, jim), \\
\text{anc}(Z,Y). & \{\text{anc}(tom, jim) \} \} \cup \textbf{M}_4 \\
\hline
\textbf{anc}(X,Y) :- & \textbf{M}_6 = T_P(\textbf{M}_5) = \textbf{M}_5 \\
\text{parent}(X,Z), & \\
\text{anc}(Z,Y). & \\
\hline
\end{tabular}
Computing $M_p$: Practical Considerations

- Computing the least Herbrand model, $M_p$, as the least fixed point of $T_p$:
  - terminates for Datalog programs (programs w/o function symbols)
  - may not terminate in general.
- For programs with function symbols, computing logical consequence by first computing $M_p$ is impractical.
- Even for Datalog programs, computing least fixed point directly using the $T_p$ operator is wasteful (known as *Naive* evaluation).
- Note that $T_p^i(\emptyset) \subseteq T_p^{i+1}(\emptyset)$.
- We can calculate $\Delta T_p^{i+1}(\emptyset) = T_p^{i+1}(\emptyset) - T_p^i(\emptyset)$ [The difference between the sets computed in two successive iterations] This strategy is known as *semi-naive* evaluation.