Semantics of Definite Logic Programs

C. R. Ramakrishnan

CSE 505
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 $\mathcal{I}$ such that

$$\mathcal{I} \models P \quad \text{and} \quad \mathcal{I} \models F \Rightarrow P \models F$$
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**.
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
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 $\forall(L_1 \lor L_2 \lor \cdots \lor L_n)$ where each $L_i$ is a literal (negative or positive) is called a clause.
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 \( \forall (L_1 \lor L_2 \lor \cdots \lor L_n) \) where each \( L_i \) is a literal (negative or positive) is called a **clause**.
- A clause \( \forall (L_1 \lor L_2 \lor \cdots \lor L_n) \) where *exactly one* literal is positive is called a **definite clause**.
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 $\forall(L_1 \lor L_2 \lor \cdots \lor L_n)$ where each $L_i$ is a literal (negative or positive) is called a clause.

- A clause $\forall(L_1 \lor L_2 \lor \cdots \lor L_n)$ where exactly one literal is positive is called a definite clause.
  A definite clause is usually written as:
  - $\forall(A_0 \lor \neg A_1 \lor \cdots \lor \neg A_n)$
  - $\ldots$ or equivalently as $A_0 \leftarrow A_1, A_2, \ldots A_n$. 
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 \textit{atomic}.

- If \( A \) is an atomic formula then
  - \( A \) is said to be a \textit{positive literal}
  - \( \neg A \) is said to be a \textit{negative literal}

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

- A clause \( \forall (L_1 \lor L_2 \lor \cdots \lor L_n) \) where \textit{exactly one} literal is positive is called a \textit{definite clause}.
  A definite clause is usually written as:
  - \( \forall (A_0 \lor \neg A_1 \lor \cdots \lor \neg A_n) \)
  - \( \ldots \) or equivalently as \( A_0 \leftarrow A_1, A_2, \ldots A_n \).

- A \textbf{definite program} is a set of definite clauses.
Given an alphabet $\mathcal{A}$, the set of all ground terms constructed from the constant and function symbols of $\mathcal{A}$ is called the **Herbrand Universe** of $\mathcal{A}$ (denoted by $U_{\mathcal{A}}$).

Consider the program:

\[
\begin{align*}
p(\text{zero}). \\
p(s(s(X))) & \leftarrow p(X).
\end{align*}
\]

The Herbrand Universe of the program’s alphabet is
Herbrand Universe

- Given an alphabet \( \mathcal{A} \), the set of all \textit{ground terms} constructed from the constant and function symbols of \( \mathcal{A} \) is called the \textbf{Herbrand Universe} of \( \mathcal{A} \) (denoted by \( U_\mathcal{A} \)).

- Consider the program:
  
  \[
  p(\text{zero}). \\
  p(s(s(X))) \leftarrow p(X).
  \]

  The Herbrand Universe of the program’s alphabet is \( \{\text{zero}, s(\text{zero}), s(s(\text{zero})), \ldots\} \).
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 Universe of the program’s alphabet is
Consider the “relations” program:

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

The Herbrand Universe of the program’s alphabet is \{ pam, bob, tom, liz, ann, pat, jim \}. 
Herbrand Base

- Given an alphabet \( \mathcal{A} \),
  the set of all \textit{ground atomic formulas} over \( \mathcal{A} \) is called the \textbf{Herbrand Base} of \( \mathcal{A} \) (denoted by \( B_{\mathcal{A}} \)).

- Consider the program:
  
  \[
  p(\text{zero}). \\
  p(s(s(X))) \leftarrow p(X).
  \]

  The Herbrand Base of the program’s alphabet is
Herbrand Base

- Given an alphabet $\mathcal{A}$, the set of all ground atomic formulas over $\mathcal{A}$ is called the **Herbrand Base** of $\mathcal{A}$ (denoted by $B_{\mathcal{A}}$).

- Consider the program:
  
  \[
  p(\text{zero}).
  \]
  
  \[
  p(s(s(X))) \leftarrow p(X).
  \]

  The Herbrand Base of the program’s alphabet is
  \[
  \{p(\text{zero}), p(s(\text{zero})), p(s(s(\text{zero}))), \ldots\}.
  \]
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
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 \{ parent(pam, pam), parent(pam, bob), parent(pam, tom), ..., parent(bob, pam), ..., grandparent(pam, pam), ..., grandparent(bob, pam), ... \}
A *Herbrand Interpretation* of a program $P$ is $\mathcal{I}$ such that

- $|\mathcal{I}| = \mathbb{U}_P$
- For every constant $c$: $c_{\mathcal{I}} = c$
- For every function symbol $f/n$: $f_{\mathcal{I}}(x_1, \ldots x_n) = f(x_1, \ldots , x_n)$
- For every predicate symbol $p/n$: $p_{\mathcal{I}} \subseteq (\mathbb{U}_P)^n$
  (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.

Examples:
- Consider our first example program.
  \[ \{ p(\text{zero}), p(s^2(\text{zero})), p(s^4(\text{zero})), \ldots \} \] represents the Herbrand model that treats \( p_{\exists} = \text{zero}, s^2(\text{zero}), s^4(\text{zero}), \ldots \) as the meaning of \( p \).
Properties of Herbrand Models

If $M$ is a family of Herbrand Models of a definite program $P$, then $\bigcap M$ is also a Herbrand Model of $P$. 
Properties of Herbrand Models

1. If $M$ is a family 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

$M_P$ = the set of all atomic ground logical consequences of $P$. 
1. If $M$ is a family 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,
Properties of Herbrand Models

1. If $M$ is a family 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 $M$, $M_P \subseteq M$. 
Properties of Herbrand Models

1. If $M$ is a family 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 $P \models F$. 
Properties of Herbrand Models

1. If $M$ is a family 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. $M_P = \text{the set of all atomic ground logical consequences of } P$. 
Sufficiency of Herbrand Models

Let $P$ be a definite program. Then if $\mathcal{G}'$ is a model of $P$ then $\mathcal{G} = \{ A \in B_P \mid \mathcal{G}' \models A \}$ is a Herbrand model of $P$. 
Sufficiency of Herbrand Models

Let $P$ be a definite program. Then if $\mathcal{S}'$ is a model of $P$ then $\mathcal{S} = \{A \in B_P \mid \mathcal{S}' \models A\}$ is a Herbrand model of $P$.

Proof (by contradiction):

- $\mathcal{S}$ is a Herbrand interpretation.
Sufficiency of Herbrand Models

Let $P$ be a definite program. Then if $\mathcal{S}'$ is a model of $P$ then 
\[ \mathcal{S} = \{ A \in B_P \mid \mathcal{S}' \models A \} \]

is a Herbrand model of $P$.

Proof (by contradiction):

- $\mathcal{S}$ is a Herbrand interpretation.
- Assume that $\mathcal{S}'$ is a model but $\mathcal{S}$ is not a model.
Sufficiency of Herbrand Models

Let $P$ be a definite program. Then if $\mathcal{S}'$ is a model of $P$ then
$\mathcal{S} = \{ A \in B_P \mid \mathcal{S}' \models A \}$ is a Herbrand model of $P$.

Proof (by contradiction):

- $\mathcal{S}$ is a Herbrand interpretation.
- Assume that $\mathcal{S}'$ is a model but $\mathcal{S}$ is not a model.
- Then there is some ground instance of a clause in $P$: $A_0: A_1, \ldots, A_n$ which is not true in $\mathcal{S}$
Sufficiency of Herbrand Models

Let $P$ be a definite program. Then if $\mathcal{S}'$ is a model of $P$ then
$\mathcal{S} = \{ A \in B_P \mid \mathcal{S}' \models A \}$ is a Herbrand model of $P$.

Proof (by contradiction):

- $\mathcal{S}$ is a Herbrand interpretation.
- Assume that $\mathcal{S}'$ is a model but $\mathcal{S}$ 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 $\mathcal{S}$
- i.e., $\mathcal{S} \models A_1 \ldots \mathcal{S} \models A_n$ but $\mathcal{S} \not\models A_0$. 
Sufficiency of Herbrand Models

Let $P$ be a definite program. Then if $\mathcal{S}'$ is a model of $P$ then $\mathcal{S} = \{ A \in BP | \mathcal{S}' \models A \}$ is a Herbrand model of $P$.

Proof (by contradiction):

- $\mathcal{S}$ is a Herbrand interpretation.
- Assume that $\mathcal{S}'$ is a model but $\mathcal{S}$ is not a model.
- Then there is some ground instance of a clause in $P$: $A_0 \leftarrow A_1, \ldots, A_n$ which is not true in $\mathcal{S}$
- i.e., $\mathcal{S} \models A_1 \ldots \mathcal{S} \models A_n$ but $\mathcal{S} \nvdash A_0$.
- By definition of $\mathcal{S}$ then, $\mathcal{S}' \models A_1 \ldots \mathcal{S}' \models A_n$ but $\mathcal{S}' \nvdash A_0$
Sufficiency of Herbrand Models

Let $P$ be a definite program. Then if $\mathcal{S}'$ is a model of $P$ then $\mathcal{S} = \{A \in B_P \mid \mathcal{S}' \models A\}$ is a Herbrand model of $P$.

Proof (by contradiction):

- $\mathcal{S}$ is a Herbrand interpretation.
- Assume that $\mathcal{S}'$ is a model but $\mathcal{S}$ 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 $\mathcal{S}$
  - i.e., $\mathcal{S} \models A_1 \ldots \mathcal{S} \models A_n$ but $\mathcal{S} \not\models A_0$.
- By definition of $\mathcal{S}$ then, $\mathcal{S}' \models A_1 \ldots \mathcal{S}' \models A_n$ but $\mathcal{S}' \not\models A_0$
- Thus $\mathcal{S}'$ is not a model, which contradicts our earlier assumption.
Sufficiency of Herbrand Models (contd.)

Let $P$ be a definite program. Then if $\mathcal{S}'$ is a model of $P$ then $\mathcal{S} = \{ A \in BP \mid \mathcal{S}' \models A \}$ is a Herbrand model of $P$.

- This holds only for definite programs.
Let $P$ be a definite program. Then if $\mathcal{S}'$ is a model of $P$ then $\mathcal{S} = \{ A \in B_P \mid \mathcal{S}' \models A \}$ is a Herbrand model of $P$.

- This holds only for definite programs.
- Consider $P = \{ \neg p(a), \exists X. p(X) \}$. 
Let $P$ be a definite program. Then if $\mathcal{S}'$ is a model of $P$ then $\mathcal{S} = \{A \in B_P \mid \mathcal{S}' \models A\}$ is a Herbrand model of $P$.

- This holds only for definite programs.
- Consider $P = \{\neg p(a), \exists X.p(X)\}$.
  - There are two Herbrand interpretations: $\mathcal{S}_1 = \{p(a)\}$ and $\mathcal{S}_2 = \{\}$. 
Sufficiency of Herbrand Models (contd.)

Let $P$ be a definite program. Then if $\mathcal{S}'$ is a model of $P$ then $\mathcal{S} = \{ A \in B_P | \mathcal{S}' \models A \}$ is a Herbrand model of $P$.

- This holds only for definite programs.
- Consider $P = \{ \neg p(a), \exists X. p(X) \}$.
  - There are two Herbrand interpretations: $\mathcal{S}_1 = \{ p(a) \}$ and $\mathcal{S}_2 = \{ \}$.
  - The first is not a model of $P$ since $\mathcal{S}_1 \not\models \neg p(a)$.
Sufficiency of Herbrand Models (contd.)

Let \( P \) be a definite program. Then if \( \mathcal{S}' \) is a model of \( P \) then
\[
\mathcal{S} = \{ A \in B_P | \mathcal{S}' \models A \}
\]
is a Herbrand model of \( P \).

- This holds only for definite programs.
- Consider \( P = \{ \neg p(a), \exists X.p(X) \} \).
  - There are two Herbrand interpretations: \( \mathcal{S}_1 = \{ p(a) \} \) and \( \mathcal{S}_2 = \{ \} \).
  - The first is not a model of \( P \) since \( \mathcal{S}_1 \not\models \neg p(a) \).
  - The second is not a model of \( P \) since \( \mathcal{S}_2 \not\models \exists X.p(X) \).
Sufficiency of Herbrand Models (contd.)

Let $P$ be a definite program. Then if $\mathcal{S}'$ is a model of $P$ then $\mathcal{S} = \{ A \in B_P \mid \mathcal{S}' \models A \}$ is a Herbrand model of $P$.

- This holds only for definite programs.
- Consider $P = \{ \neg p(a), \exists X. p(X) \}$.
  - There are two Herbrand interpretations: $\mathcal{S}_1 = \{ p(a) \}$ and $\mathcal{S}_2 = \{ \}$.
  - The first is not a model of $P$ since $\mathcal{S}_1 \not\models \neg p(a)$.
  - The second is not a model of $P$ since $\mathcal{S}_2 \not\models \exists X. p(X)$.
  - But there is a non-Herbrand model $\mathcal{S}$:
Let $P$ be a definite program. Then if $\mathcal{S}'$ is a model of $P$ then $\mathcal{S} = \{ A \in B_P \mid \mathcal{S}' \models A \}$ is a Herbrand model of $P$.

- This holds only for definite programs.
- Consider $P = \{ \neg p(a), \exists X. p(X) \}$.
  - There are two Herbrand interpretations: $\mathcal{S}_1 = \{ p(a) \}$ and $\mathcal{S}_2 = \{ \}$.
  - The first is not a model of $P$ since $\mathcal{S}_1 \not\models \neg p(a)$.
  - The second is not a model of $P$ since $\mathcal{S}_2 \not\models \exists X. p(X)$
  - But there is a non-Herbrand model $\mathcal{S}$:
    - $| \mathcal{S} | = N$, the set of natural numbers
Let $P$ be a definite program. Then if $\mathcal{I}'$ is a model of $P$ then $\mathcal{I} = \{A \in BP \mid \mathcal{I}' \models A\}$ is a Herbrand model of $P$.

- This holds only for definite programs.
- Consider $P = \{\neg p(a), \exists X.p(X)\}$.
  - There are two Herbrand interpretations: $\mathcal{I}_1 = \{p(a)\}$ and $\mathcal{I}_2 = \{\}$.  
  - The first is not a model of $P$ since $\mathcal{I}_1 \not\models \neg p(a)$.
  - The second is not a model of $P$ since $\mathcal{I}_2 \not\models \exists X.p(X)$
  - But there is a non-Herbrand model $\mathcal{I}$:
    - $|\mathcal{I}| = N$, the set of natural numbers
    - $a_{\mathcal{I}} = 0$
Sufficiency of Herbrand Models (contd.)

Let \( P \) be a definite program. Then if \( \mathcal{S}' \) is a model of \( P \) then \( \mathcal{S} = \{ A \in B_P \mid \mathcal{S}' \models A \} \) is a Herbrand model of \( P \).

- This holds only for definite programs.
- Consider \( P = \{ \neg p(a), \exists X. p(X) \} \).
  - There are two Herbrand interpretations: \( \mathcal{S}_1 = \{ p(a) \} \) and \( \mathcal{S}_2 = \{ \} \).
  - The first is not a model of \( P \) since \( \mathcal{S}_1 \not\models \neg p(a) \).
  - The second is not a model of \( P \) since \( \mathcal{S}_2 \not\models \exists X. p(X) \).
  - But there is a non-Herbrand model \( \mathcal{S} \):
    - \( | \mathcal{S} | = N \), the set of natural numbers
    - \( a_\mathcal{S} = 0 \)
    - \( p_\mathcal{S} = \text{“is odd”} \)
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 $P$. 
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 $M \models A_1 \cdots M \models A_n$ but $M \not\models A_0$. 
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 $A_0 \Leftarrow A_1, \ldots, A_n$ such that $M \models A_1 \ldots M \models A_n$ but $M \not\models A_0$.
  - Which means $A_0 \not\in M_1$ or $A_0 \not\in M_2$. 
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 $M \models A_1 \cdots M \models A_n$ but $M \not\models A_0$.
  
  - Which means $A_0 \not\in M_1$ or $A_0 \not\in M_2$.
  
  - But $A_1, \ldots A_n \in M_1$ as well as $M_2$. 


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 $A_0 \not\in M_1$ or $A_0 \not\in M_2$.
  - 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

- 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 $M \models A_1 \cdots M \models A_n$ but $M \not\models A_0$.
  - Which means $A_0 \not\in M_1$ or $A_0 \not\in M_2$.
  - 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.

- There is a unique least Herbrand model.
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 $A_0 \not\in M_1$ or $A_0 \not\in M_2$.
  - 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.

- There is a unique least Herbrand model.
  - Let $M_1$ and $M_2$ are two incomparable minimal Herbrand models,
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 $M \models A_1 \cdots M \models A_n$ but $M \not\models A_0$.
  
  Which means $A_0 \not\in M_1$ or $A_0 \not\in M_2$.
  
  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.

- There is a unique least Herbrand model.
  
  Let $M_1$ and $M_2$ are two incomparable minimal Herbrand models, $M = M_1 \cap M_2$ is also a Herbrand model, and
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 $M \models A_1 \cdots M \models A_n$ 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.

- There is a unique least Herbrand model.
  - Let $M_1$ and $M_2$ are two incomparable minimal Herbrand models,
  - $M = M_1 \cap M_2$ is also a Herbrand model, and
  - $M \subseteq M_1$ and $M \subseteq M_2$. 


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 $M \models A_1 \cdots M \models A_n$ but $M \not\models A_0$.
  - Which means $A_0 \not\in M_1$ or $A_0 \not\in M_2$.
  - 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.

- There is a unique least Herbrand model.
  - Let $M_1$ and $M_2$ are two incomparable minimal Herbrand models,
  - $M = M_1 \cap M_2$ is also a Herbrand model, 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.

- $M_P = \{ A \in B_P \mid P \models A \}$
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.
  - 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 $\mathcal{I}' \models \neg A$.
    - By sufficiency of Herbrand models, there is some Herbrand model $\mathcal{I}$ such that $\mathcal{I} \models \neg A$.
    - Hence $A$ is not in some Herbrand model, and hence is not in $M_P$. 

---

Motivation
Herbrand Models
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.
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.
- Second, $M_P \subseteq \{ A \in B_P \mid P \models A \}$:
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.
- 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 $P$. 

**Motivation**

Herbrand Models

*Least Herbrand Model*
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.
- 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 $\mathcal{G}' \models \neg A$. 
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.
- 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 $\mathcal{G}' \models \neg A$.
  - By sufficiency of Herbrand models, there is some Herbrand model $\mathcal{G}$ such that $\mathcal{G} \models \neg A$. 
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.
- 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 $\mathcal{S}' \models \neg A$.
  - By sufficiency of Herbrand models, there is some Herbrand model $\mathcal{S}$ such that $\mathcal{S} \models \neg A$.
  - Hence $A$ is not in some Herbrand model, and hence is not in $M_P$. 

Motivation

Herbrand Models

Least Herbrand Model
Finding the Least Herbrand Model

Immediate consequence operator:

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

- Consider the sequence: $\emptyset, T_P(\emptyset), T_P^2(\emptyset), \ldots, T_P^i(\emptyset), \ldots$
- $M_P \supseteq T_P^i(\emptyset)$ for all $i$.
- Let $T_P \uparrow \omega = \bigcup_{i=0}^{\infty} T_P^i(\emptyset)$.
  
  Then $M_P \subseteq T_P \uparrow \omega$.
Computing Least Herbrand Models: An Example

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

anc(X, Y) :-
    parent(X, Y).
anc(X, Y) :-
    parent(X, Z),
    anc(Z, Y).
Computing Least Herbrand Models: An Example

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

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

\[ M_1 \]
\[ \emptyset \]
Computing Least Herbrand Models: An Example

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

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

\[ M_1 = \emptyset \]
\[ M_2 = TP(M_1) = \]
\[ M_3 = TP(M_2) = \]
\[ M_4 = TP(M_3) = \]
\[ M_5 = TP(M_4) = \]
\[ M_6 = TP(M_5) = \]
Computing Least Herbrand Models: An Example

\[
\begin{align*}
M_1 & = \emptyset \\
M_2 = TP(M_1) & = \{\text{parent(pam, bob), parent(tom, bob), parent(tom, liz), parent(bob, ann), parent(bob, pat), parent(pat, jim)}\}
\end{align*}
\]
Computing Least Herbrand Models: An Example

\[
\begin{align*}
M_1 &= \emptyset \\
M_2 &= TP(M_1) = \{ \text{parent(pam, bob)}, \\
& \quad \text{parent(tom, bob)}, \\
& \quad \text{parent(tom, liz)}, \\
& \quad \text{parent(bob, ann)}, \\
& \quad \text{parent(bob, pat)}, \\
& \quad \text{parent(pat, jim)} \} \\
M_3 &= TP(M_2) = \\
M_4 &= TP(M_3) = \\
M_5 &= TP(M_4) = \\
M_6 &= TP(M_5) = \\
\end{align*}
\]

\text{parent(pam, bob).} \\
\text{parent(tom, bob).} \\
\text{parent(tom, liz).} \\
\text{parent(bob, ann).} \\
\text{parent(bob, pat).} \\
\text{parent(pat, jim).}

\text{anc(X,Y) :-} \\
\quad \text{parent(X,Y).} \\
\text{anc(X,Y) :-} \\
\quad \text{parent(X,Z),} \\
\quad \text{anc(Z,Y).}
Computing Least Herbrand Models: An Example

\[
\begin{align*}
M_1 & = \emptyset \\
M_2 = T_P(M_1) & = \{ \text{parent}(\text{pam, bob}), \text{parent}(\text{tom, bob}), \text{parent}(\text{tom, liz}), \text{parent}(\text{bob, ann}), \text{parent}(\text{bob, pat}), \text{parent}(\text{pat, jim}) \} \\
M_3 = T_P(M_2) & = \{ \text{anc}(\text{pam, bob}), \text{anc}(\text{tom, bob}), \text{anc}(\text{tom, liz}), \text{anc}(\text{bob, ann}), \text{anc}(\text{bob, pat}), \text{anc}(\text{pat, jim}) \} \\
& \cup M_2
\end{align*}
\]
Computing Least Herbrand Models: An Example

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

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

\[ M_1 = \emptyset \]
\[ M_2 = T_P(M_1) = \{ \text{parent}(\text{pam}, \text{bob}), \text{parent}(\text{tom}, \text{bob}), \text{parent}(\text{tom}, \text{liz}), \text{parent}(\text{bob}, \text{ann}), \text{parent}(\text{bob}, \text{pat}), \text{parent}(\text{pat}, \text{jim}) \} \]
\[ M_3 = T_P(M_2) = \{ \text{anc}(\text{pam}, \text{bob}), \text{anc}(\text{tom}, \text{bob}), \text{anc}(\text{tom}, \text{liz}), \text{anc}(\text{bob}, \text{ann}), \text{anc}(\text{bob}, \text{pat}), \text{anc}(\text{pat}, \text{jim}) \} \cup M_2 \]
\[ M_4 = T_P(M_3) = \]
Computing Least Herbrand Models: An Example

<table>
<thead>
<tr>
<th>$M_1$</th>
<th>$\emptyset$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_2 = T_P(M_1)$ =</td>
<td>${ \text{parent(pam,bob)},$</td>
</tr>
<tr>
<td></td>
<td>$\text{parent(tom,bob)},$</td>
</tr>
<tr>
<td></td>
<td>$\text{parent(tom,liz)},$</td>
</tr>
<tr>
<td></td>
<td>$\text{parent(bob,ann)},$</td>
</tr>
<tr>
<td></td>
<td>$\text{parent(bob,pat),}$</td>
</tr>
<tr>
<td></td>
<td>$\text{parent(pat,jim)} }$</td>
</tr>
<tr>
<td>$M_3 = T_P(M_2)$ =</td>
<td>${ \text{anc(pam,bob)},$</td>
</tr>
<tr>
<td></td>
<td>$\text{anc(tom,bob)},$</td>
</tr>
<tr>
<td></td>
<td>$\text{anc(tom,liz)},$</td>
</tr>
<tr>
<td></td>
<td>$\text{anc(bob,ann)},$</td>
</tr>
<tr>
<td></td>
<td>$\text{anc(bob,pat),}$</td>
</tr>
<tr>
<td></td>
<td>$\text{anc(pat,jim)} }$</td>
</tr>
<tr>
<td></td>
<td>$\cup M_2$</td>
</tr>
<tr>
<td>$M_4 = T_P(M_3)$ =</td>
<td>${ \text{anc(pam,ann)},$</td>
</tr>
<tr>
<td></td>
<td>$\text{anc(pam,pat)},$</td>
</tr>
<tr>
<td></td>
<td>$\text{anc(tom,ann)},$</td>
</tr>
<tr>
<td></td>
<td>$\text{anc(tom,pat),}$</td>
</tr>
<tr>
<td></td>
<td>$\text{anc(bob,jim)} }$</td>
</tr>
<tr>
<td></td>
<td>$\cup M_3$</td>
</tr>
</tbody>
</table>
### Computing Least Herbrand Models: An Example

<table>
<thead>
<tr>
<th></th>
<th>$M_1$</th>
<th>$\emptyset$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_2 = T_P(M_1)$</td>
<td>${\text{parent(pam,bob), parent(tom,bob), parent(tom,liz), parent(bob,ann), parent(bob,pat), parent(pat,jim)}}$</td>
<td></td>
</tr>
<tr>
<td>$M_3 = T_P(M_2)$</td>
<td>${\text{anc(pam,bob), anc(tom,bob), anc(tom,liz), anc(bob,ann), anc(bob,pat), anc(pat,jim)}}$ $\cup M_2$</td>
<td></td>
</tr>
<tr>
<td>$M_4 = T_P(M_3)$</td>
<td>${\text{anc(pam,ann), anc(pam,pat), anc(tom,ann), anc(tom,pat), anc(bob,jim)}}$ $\cup M_3$</td>
<td></td>
</tr>
<tr>
<td>$M_5 = T_P(M_4)$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Computing Least Herbrand Models: An Example

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

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

\[ M_1 = T_P(M_1) = \emptyset \]

\[ M_2 = T_P(M_1) = \{\text{parent(pam,bob), parent(tom,bob), parent(tom,liz), parent(bob,ann), parent(bob,pat), parent(pat,jim)}\} \]

\[ M_3 = T_P(M_2) = \{\text{anc(pam,bob), anc(tom,bob), anc(tom,liz), anc(bob,ann), anc(bob,pat), anc(pat,jim)}\} \cup M_2 \]

\[ M_4 = T_P(M_3) = \{\text{anc(pam,ann), anc(pam,pat), anc(tom,ann), anc(tom,pat), anc(bob,jim)}\} \cup M_3 \]

\[ M_5 = T_P(M_4) = \{\text{anc(pam,jim), \{anc(tom,jim)\}}\} \cup M_4 \]
Computing Least Herbrand Models: An Example

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

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

<table>
<thead>
<tr>
<th>( M_1 )</th>
<th>( \emptyset )</th>
</tr>
</thead>
</table>
| \( M_2 = TP(M_1) = \) | \{parent(pam,bob),
        parent(tom,bob),
        parent(tom,liz),
        parent(bob,ann),
        parent(bob,pat),
        parent(pat,jim)\} |

| \( M_3 = TP(M_2) = \) | \{anc(pam,bob),
        anc(tom,bob),
        anc(tom,liz),
        anc(bob,ann),
        anc(bob,pat),
        anc(pat,jim)\} \cup M_2 |

| \( M_4 = TP(M_3) = \) | \{anc(pam,ann),
        anc(pam,pat),
        anc(tom,ann),
        anc(tom,pat),
        anc(bob,jim)\} \cup M_3 |

| \( M_5 = TP(M_4) = \) | \{anc(pam,jim), \{anc(tom,jim)\}\} \cup M_4 |

\( M_6 = TP(M_5) = \)
Computing Least Herbrand Models: An Example

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

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

M₁ = \emptyset

M₂ = \text{T}_{P}(M₁) = \{\text{parent}(pam, bob), \text{parent}(tom, bob), \text{parent}(tom, liz), \text{parent}(bob, ann), \text{parent}(bob, pat), \text{parent}(pat, jim)\}

M₃ = \text{T}_{P}(M₂) = \{\text{anc}(pam, bob), \text{anc}(tom, bob), \text{anc}(tom, liz), \text{anc}(bob, ann), \text{anc}(bob, pat), \text{anc}(pat, jim)\} \cup M₂

M₄ = \text{T}_{P}(M₃) = \{\text{anc}(pam, ann), \text{anc}(pam, pat), \text{anc}(tom, ann), \text{anc}(tom, pat), \text{anc}(bob, jim)\} \cup M₃

M₅ = \text{T}_{P}(M₄) = \{\text{anc}(pam, jim), \{\text{anc}(tom, jim)\}\} \cup M₄

M₆ = \text{T}_{P}(M₅) = M₅
```
Computing $M_P$: Practical Considerations

- Computing the least Herbrand model, $M_P$, as the *least fixed point of* $T_P$:
Computing $M_P$: Practical Considerations

- Computing the least Herbrand model, $M_P$, as the \textit{least fixed point} of $T_P$:
  - terminates for Datalog programs (programs w/o function symbols)
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
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.
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).
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 $T_P^i(\emptyset) \subseteq T_P^{i+1}(\emptyset)$. 

Note that $T_P^i(\emptyset) \subseteq T_P^{i+1}(\emptyset)$. 

This strategy is known as semi-naive evaluation.
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]
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.