Definite Logic Programs: Models

CSE 505 – Computing with Logic

Stony Brook University

htTp://www.cs.stonybrook.edu/~cse505

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

$$I \models P \quad \text{and} \quad I \models F \quad \rightarrow \quad P \models F$$

Definite Clauses

- A formula of the form $\mathtt{p(t_1, t_2, \ldots, t_n)}$, where $\mathtt{p/n}$ is an $\mathtt{n}$-ary predicate symbol and $\mathtt{t_i}$ are all terms is said to be ***atomic***.
- If $\mathtt{A}$ is an atomic formula then:
  - $\mathtt{A}$ is said to be a ***positive literal***
  - $\neg\mathtt{A}$ is said to be a ***negative literal***
- A formula of the form $\forall(\mathtt{L_1} \vee \mathtt{L_2} \vee \ldots \vee \mathtt{L_n})$ where each $\mathtt{L_i}$ is a literal (negative or positive) is called a ***clause***.
- A clause $\forall(\mathtt{L_1} \vee \mathtt{L_2} \vee \ldots \vee \mathtt{L_n})$ where exactly one literal is positive is called a ***definite clause*** (also called ***Horn clause***).
  - A definite clause is usually written as:
    - $\forall(\mathtt{A_0} \vee \neg\mathtt{A_1} \vee \ldots \vee \neg\mathtt{A_n})$
    - or equivalently as: $\mathtt{A_0} \leftarrow \mathtt{A_1}, \mathtt{A_2}, \ldots, \mathtt{A_n}.$
- A ***definite program*** is a set of definite clauses.

Herbrand Universe

- Given an alphabet A, the set of all **<u>ground terms</u>** 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))) ← p(X).
```

Jacques Herbrand
(1908 –1931)

- The Herbrand Universe of the program's alphabet is: U_A = {**`zero,s(zero),s(s(zero)),...`**}

Herbrand Universe: Example

- 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:

$U_A = \{$**pam, bob, tom, liz, ann, pat, jim**$\}$

Herbrand Base

- Given an alphabet A, the set of all **<u>ground atomic formulas</u>** over A is called the ***Herbrand Base*** of A (denoted by B_A)
- Consider the program:

```
p(zero).
p(s(s(X))) ← p(X).
```

- The Herbrand Base of the program's alphabet is: B_A={**`p(zero), p(s(zero)), p(s(s(zero))),...`**}

Herbrand Base: Example

- 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), ..., parent(bob, pam), ..., grandparent(pam,pam),...,grandparent(bob,pam), ...`}.

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**: $\mathbf{c}_I = \mathbf{c}$
  - For every function symbol **f/n**: $\mathbf{f}_I(\mathbf{x}_1,\ldots,\mathbf{x}_n) = \mathbf{f}(\mathbf{x}_1,\ldots,\mathbf{x}_n)$
  - For every predicate symbol **p/n**: $\mathbf{p}_I \subseteq (U_P)^{\mathbf{n}}$ (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

Herbrand Models

- 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`**.

  - If we have several predicates, the Herbrand interpretation would be a single set of all true predicates

Sufficiency of Herbrand Models

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

Proof (by contradiction):

- Assume that I' is a model of P but I (defined above) is not a model.
- Then there is some ground instance of a clause in P:

$$\mathbf{A}_0 \ \text{:-}\ \mathbf{A}_1, \ \ldots, \ \mathbf{A}_n.$$

- which is not true in I i.e., $I \models \mathbf{A}_1, \ldots, I \models \mathbf{A}_n$ but $I \not\models \mathbf{A}_0$
- By definition of I then, $I' \models \mathbf{A}_1, \ldots, I' \models \mathbf{A}_n$ but $I' \not\models \mathbf{A}_0$
- 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 $I = \{\mathbf{A} \in B_P \mid I' \models \mathbf{A}\}$ is a Herbrand model of P.
- **This property holds only for definite programs!**
  - Example: 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 are non-Herbrand models, such as I:
      - $|I| = N$ (the set of natural numbers)
      - $a_I = 0$
      - $p_I =$ "is odd"

Properties of Herbrand Models

1. For any definite program P, if every Herbrand Model of P is also a Herbrand Model of F, then P $\models$ F.
2. If M is a set of Herbrand Models of a definite program P, then $\bigcap$M is also a Herbrand Model of P.
3. For every definite program P there is a **unique** ***least model*** Mp such that:
   a) Mp is a Herbrand Model of P and,
   b) for every Herbrand Model M, Mp $\subseteq$ M.
4. Mp = 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.

<u>Proof:</u>

- Assume $M=M_1 \cap M_2$ is not a model.
- Then there is some clause $\mathbf{A_0} :- \mathbf{A_1}, \ldots, \mathbf{A_n}$ such that $M \models \mathbf{A_1}, \ldots, M \models \mathbf{A_n}$ but $M \not\models \mathbf{A_0}$
- Which means $\mathbf{A_0} \notin$ M1 or $\mathbf{A_0} \notin$ M2 by def. of $\cap$
- But $\mathbf{A_1}, \ldots, \mathbf{A_n} \in M_1$ 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.

<u>Proof:</u>

- Let M_1 and M_2 are two incomparable **<u>minimal</u>** Herbrand models (incomparable means neither one is a subset of the other), but $M=M_1 \cap M_2$ is also a Herbrand model (previous theorem), and $M \subset M_1$ or $M \subset M_2$
- Thus M_1 on M_2 is not minimal.

Least Herbrand Model

- The ***least Herbrand model*** Mp of a definite program P is the **set of all ground logical consequences of the program:**

$$Mp = \{A \in Bp \mid P \models \mathbf{A}\}$$

Proof:

- First, $Mp \supseteq \{A \in Bp \mid P \models \mathbf{A}\}$ (i.e., Mp is a superset of the logical consequences $\{A \in Bp \mid P \models \mathbf{A}\}$):
  - By definition of logical consequence, $P \models \mathbf{A}$ means that $\mathbf{A}$ **must be in every model of P** and hence also in the least Herbrand model.

Least Herbrand Model

- Second, $M_P \subseteq \{A \in B_P \mid P \models A\}$ (i.e., Mp is a subset of the logical consequences $\{A \in B_P \mid P \models \mathbf{A}\}$):
  - Assume that **A** is in Mp. Hence, **A** is in every Herbrand model of P by def. of Mp (i.e., subset of all models)
  - Assume that **A** is not true in some non-Herbrand model of P: $I' \models \neg \mathbf{A}$
  - By sufficiency of Herbrand models (i.e., If I' is a model of P then $I = \{\mathbf{A} \in B_P \mid I' \models \mathbf{A}\}$ is a Herbrand model of P), there is some Herbrand model I such that $I \models \neg \mathbf{A}$
  - Hence **A** cannot be an element of the Herbrand model I
  - This contradicts that **A** is in every Herbrand model of P, and their intersection Mp

Construction of Least Herbrand Models

- Definition: ***Immediate consequence operator:***
  - Given an interpretation $I \subseteq Bp$, construct I' such that

    $I' = \{\mathbf{A_0} \in Bp \mid \mathbf{A_0} \leftarrow \mathbf{A_1},\ldots,\mathbf{A_n}$ is a ground instance of a clause in P and $\mathbf{A_1},\ldots,\mathbf{A_n} \in I\}$
  - I' is said to be the ***immediate consequence of*** I

  written as $I' = Tp(I)$, where 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 (Mp is a **superset** of all $Tp^i(\emptyset)$)
  - Let $Tp \uparrow \omega = \bigcup_{i=0,\infty} Tp^i(\emptyset)$

  Then $Mp = Tp \uparrow \omega$

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).
```

M_1	$\emptyset$
$M_2 = T_P(M_1) =$	{parent(pam,bob), parent(tom,bob), parent(tom,liz), parent(bob,ann), parent(bob,pat), parent(pat,jim) }
$M_3 = T_P(M_2) =$	{anc(pam,bob), anc(tom,bob), anc(tom,liz), anc(bob,ann), anc(bob,pat), anc(pat,jim) } $\cup M_2$
$M_4 = T_P(M_3) =$	{anc(pam,ann), anc(pam,pat), anc(tom,ann), anc(tom,pat), anc(bob,jim) } $\cup M_3$
$M_5 = T_P(M_4) =$	{anc(pam,jim), {anc(tom,jim) } $\cup M_4$
$M_6 = T_P(M_5) =$	M_5

Computing Mp

- Computing the least Herbrand model, Mp, as the **<u>least fixed point</u>** of Tp:
  - terminates for ***Datalog*** programs (i.e., programs **w/o function symbols**)
  - may not terminate in general (because it could be infinite)
    - For programs with function symbols
- Even for Datalog programs, computing least fixed point directly using the Tp operator is <u>wasteful</u> (known as ***Naive*** evaluation)
  - Note that $Tp^i(\emptyset) \subseteq Tp^{i+1}(\emptyset)$ for all i
  - We can calculate $\Delta Tp^{i+1}(\emptyset) = Tp^{i+1}(\emptyset) - Tp^i(\emptyset)$ [The difference between the sets computed in two successive iterations] (this strategy is known as the ***semi-naive*** evaluation)