Chapter 7

Negation: Declarative Interpretation
Outline

- First-Order Formulas and Logical Truth
- The Completion semantics
- Soundness and restricted completeness of SLDNF-Resolution
- Extended consequence operator
- An alternative semantics: Standard models


First-Order Formulas

$\Pi$, $F$ ranked alphabets of predicate symbols and function symbols, respectively, $V$ set of variables

The (first-order) formulas (over $\Pi$, $F$, and $V$) are inductively defined as follows:

1. If $A \in TB_{\Pi,F,V}$, then $A$ is a formula

2. If $G_1$ and $G_2$ are formulas, then $\neg G_1$, $G_1 \land G_2$ (written $G_1 \land G_2$), $G_1 \lor G_2$, $G_1 \leftarrow G_2$, and $G_1 \leftrightarrow G_2$ are formulas

3. If $G_1$ is formula and $x \in V$, then $\forall x \ G$ and $\exists x \ G$ are formulas
Extended Notion of Logical Truth (I)

$G$ formula, $I$ interpretation with domain $D$, $\sigma : V \rightarrow D$ state

$G$ true in $I$ under $\sigma$, written $I \models_{\sigma} G$ :

- $I \models_{\sigma} p(t_1, ..., t_n) \iff (\sigma(t_1), ..., \sigma(t_n)) \in p_I$
- $I \models_{\sigma} \neg G \iff I \not\models_{\sigma} G$
- $I \models_{\sigma} G_1 \land G_2 \iff I \models_{\sigma} G_1$ and $I \models_{\sigma} G_2$
- $I \models_{\sigma} G_1 \lor G_2 \iff I \models_{\sigma} G_1$ or $I \models_{\sigma} G_2$
- $I \models_{\sigma} G_1 \leftarrow G_2 \iff$ if $I \models_{\sigma} G_2$ then $I \models_{\sigma} G_1$
- $I \models_{\sigma} G_1 \leftrightarrow G_2 \iff I \models_{\sigma} G_1$ iff $I \models_{\sigma} G_2$
- $I \models_{\sigma} \forall x \ G \iff$ for every $d \in D$: $I \models_{\sigma'} G$
- $I \models_{\sigma} \exists x \ G \iff$ for some $d \in D$: $I \models_{\sigma'} G$

where $\sigma' : V \rightarrow D$ with $\sigma'(x) = d$ and $\sigma'(y) = \sigma(y)$ for every $y \in V - \{x\}$
Extended Notion of Logical Truth (II)

Let \( x_1, \ldots, x_k \) be the variables occurring in \( G \).

- \( \forall x_1, \ldots, \forall x_k \ G \) universal closure of \( G \) (abbreviated \( \forall G \))
- \( I \vDash \forall G :\iff I \vDash \forall_\sigma G \) for every state \( \sigma \)
- \( I \vDash_\sigma p(t_1, \ldots, t_n) :\iff (\sigma(t_1), \ldots, \sigma(t_n)) \in p_I \)
- \( G \) true in \( I \) (or: \( I \) model of \( G \)), written: \( I \vDash G :\iff I \vDash \forall G \)
- \( I \) model of \( S \), written: \( I \vDash S :\iff I \vDash G \) for every \( G \in S \)
- \( T \) semantic (or: logical) consequence of \( S \), written \( S \vDash T :\iff \) every model of \( S \) is a model of \( T \)
Programs Never Have Negative Consequences (I)

\[ P_{mem}: \]
\[
\text{member}(x, [x\mid y]) \leftarrow \\
\text{member}(x, [y\mid z]) \leftarrow \text{member}(x, z)
\]

Then \( P_{mem} \models \text{member}(a, [a,b]) \) and \( P_{mem} \not\models \text{member}(a, [\ ]) \).

But also \( P_{mem} \not\models \neg \text{member}(a, [\ ]) \), since

\[ HB_{\{\text{member},\{[\ ],a\} \}} \models P_{mem} \quad \text{and} \quad HB_{\{\text{member},\{[\ ],a\} \}} \not\models \neg \text{member}(a, [\ ]). \]

Nevertheless the SLDNF-tree of \( P_{mem} \cup \{\neg \text{member}(a, [\ ])\} \) is successful:

\[ \neg \text{member}(a,[\ ]) \quad \neg \quad \neg \text{member}(a,[\ ]) \]

\[ \square \quad \text{success} \]

\[ \text{member}(a,[\ ]) \]

\[ \text{failure} \]
Programs Never Have Negative Consequences (II)

Problem: For every extended program $P$ the “corresponding” Herbrand base is a model.

Hence: No negative ground literal $L$ can ever be a logical consequence of $P$.

But: SLDNF-tree of $P \cup \{L\}$ may be successful!

$\Rightarrow$ Soundness of SLDNF-resolution?

Solution: Strengthen $P$ by completion (“replace implications by equivalences”) to $\text{comp}(P)$ and compare SLDNF-resolution with $\text{comp}(P)$ instead of $P$!
Completion (Example I)

\[
\begin{array}{l}
P: \quad \textit{happy} \quad \leftarrow \quad \textit{sun, holidays} \\
\textit{happy} \quad \leftarrow \quad \textit{snow, holidays} \\
\textit{snow} \quad \leftarrow \quad \textit{cold, precipitation} \\
\textit{cold} \quad \leftarrow \quad \textit{winter} \\
\textit{precipitation} \quad \leftarrow \quad \textit{holidays} \\
\textit{winter} \quad \leftarrow \quad \textit{holidays} \\
\textit{holidays} \quad \leftarrow \quad \end{array}
\]

\[
\begin{array}{l}
\text{comp}(P): \quad \textit{happy} \quad \leftrightarrow \quad (\textit{sun, holidays}) \lor (\textit{snow, holidays}) \\
\textit{snow} \quad \leftrightarrow \quad \textit{cold, precipitation} \\
\textit{cold} \quad \leftrightarrow \quad \textit{winter} \\
\textit{precipitation} \quad \leftrightarrow \quad \textit{holidays} \\
\textit{winter} \quad \leftrightarrow \quad \textit{true} \\
\textit{holidays} \quad \leftrightarrow \quad \textit{true} \\
\textit{sun} \quad \leftrightarrow \quad \textit{false} \\
\end{array}
\]

Then, \( \text{comp}(P) \models \textit{happy}, \textit{snow}, \textit{cold}, \textit{precipitation}, \textit{winter}, \textit{holidays}, \neg \textit{sun} \).
Completion (Example II)

\[ P: \]
\begin{align*}
  &\text{member}(x, [x|y]) \leftarrow \\
  &\text{member}(x, [y|z]) \leftarrow \text{member}(x, z) \\
  &\text{disjoint}([\ ], x) \leftarrow \\
  &\text{disjoint}([x|y], z) \leftarrow \text{member}(x, z), \text{disjoint}(y, z)
\end{align*}

\[ \text{comp}(P): \]
\begin{align*}
  \forall x_1, x_2 &\text{ member}(x_1, x_2) \iff \exists x, y \ (x_1 = x, x_2 = [x|y]) \lor \\
  &\exists x, y, z \ (x_1 = x, x_2 = [y|z], \text{member}(x, z)) \\
  \forall x_1, x_2 &\text{ disjoint}(x_1, x_2) \iff \exists x \ (x_1 = [\ ], x_2 = x) \lor \\
  &\exists x, y, z \ (x_1 = [x|y], x_2 = z, \\
  &\quad \neg \text{member}(x, z), \text{disjoint}(y, z))
\end{align*}

plus standard equality and inequality axioms

Then, e.g. \( \text{comp}(P) \models \text{member}(a, [a|b]), \neg \text{member}(a, [\ ]), \neg \text{disjoint}([a], [a]). \)
Completion (I)

Completion of extended program $P$ (denoted by $\text{comp}(P)$) is the set of formulas constructed from $P$ by the following 6 steps:

1. Associate with every $n$-ary predicate symbol $p$ a sequence of pairwise distinct variables $x_1, \ldots, x_n$ which do not occur in $P$.

2. Transform each clause $c = p(t_1, \ldots, t_n) \leftarrow B$ into

   $$p(x_1, \ldots, x_n) \leftarrow x_1 = t_1, \ldots, x_n = t_n, B$$

3. Transform each resulting formula $p(x_1, \ldots, x_n) \leftarrow G$ into

   $$p(x_1, \ldots, x_n) \leftarrow \exists z G$$

   where $z$ is a sequence of the elements of $\text{Var}(c)$.
4. For every $n$-ary predicate symbol $p$, let

\[ p(x_1, ..., x_n) \leftarrow \exists z_1 G_1, ..., p(x_1, ..., x_n) \leftarrow \exists z_m G_m \]

be all implications obtained in Step 3 ($m \geq 0$).

- If $m > 0$, then replace these by the formula
  \[ \forall x_1, ..., x_n \ p(x_1, ..., x_n) \leftrightarrow \exists z_1 G_1 \lor ... \lor \exists z_m G_m \]
  (If some $\exists z_i G_i$ is empty, then replace it by true.)

- If $m = 0$, then add the formula
  \[ \forall x_1, ..., x_n \ p(x_1, ..., x_n) \leftrightarrow false \]
5. **Standard axioms of equality**

\[ \forall [ x = x ] \]
\[ \forall [ x = y \rightarrow y = x ] \]
\[ \forall [ x = y \land y = z \rightarrow x = z ] \]
\[ \forall [ x_i = y \rightarrow f(x_1, \ldots, x_i, \ldots, x_n) = f(x_1, \ldots, y, \ldots, x_n) ] \]
\[ \forall [ x_i = y \rightarrow (p(x_1, \ldots, x_i, \ldots, x_n) \leftrightarrow p(x_1, \ldots, y, \ldots, x_n)) ] \]

6. **Standard axioms of inequality**

\[ \forall [ x_1 \neq y_1 \lor \ldots \lor x_n \neq y_n \rightarrow f(x_1, \ldots, x_n) \neq f(y_1, \ldots, y_n) ] \]
\[ \forall [ f(x_1, \ldots, x_m) \neq g(y_1, \ldots, y_n) ] \quad \text{(whenever } f \neq g) \]
\[ \forall [ x \neq t ] \quad \text{(whenever } x \text{ is proper subterm of } t) \]

5. and 6. ensure that = must be interpreted as equality!
Soundness of SLDNF-Resolution

$P$ extended program, $Q$ extended query, $\theta$ substitution:

- $\theta \mid_{\text{var}(Q)}$ correct answer substitution of $Q : \iff \text{comp}(P) \models Q\theta$
- $Q\theta$ correct instance of $Q : \iff \text{comp}(P) \models Q\theta$

**Theorem (cf. e.g. [Lloyd, 1987])**

If there exists a successful SLDNF-derivation of $P \cup \{Q\}$ with $\text{CAS } \theta$, then $\text{comp}(P) \models Q\theta$.

**Corollary**

If there exists a successful SLDNF-derivation of $P \cup \{Q\}$, then $\text{comp}(P) \models \exists Q$. 
SLDNF-Resolution is Not Complete (I): Inconsistency

\[ P: \quad p \leftarrow \neg p \]

\[ \text{comp}(P) \supseteq \{p \leftrightarrow \neg p\} \quad \text{"="} \quad \{\text{false}\}. \]

Hence, \( \text{comp}(P) \models p \) and \( \text{comp}(P) \models \neg p. \)

(because \( I \not\models \text{comp}(P) \) for every interpretation \( I \), i.e. \( \text{comp}(P) \) is inconsistent)

But there is neither a successful SLDNF-derivation of \( P \cup \{p\} \) nor of \( P \cup \{\neg p\}. \)
SLDNF-Resolution is Not Complete (II): Non-Strictness

\[
P: \\
p \leftarrow q \\
p \leftarrow \neg q \\
q \leftarrow q
\]

\[\text{comp}(P) \supseteq \{p \leftrightarrow q \lor \neg q, q \leftrightarrow q\} \equiv \{p \leftrightarrow \text{true}\}.\]

Hence, \(\text{comp}(P) \models p.\)

But there is no successful SLDNF-derivation of \(P \cup \{p\}.\)
SLDNF-Resolution is Not Complete (III): Floundering

$$P: \quad p(x) \leftarrow \neg q(x)$$

\[\text{comp}(P) \supseteq \{ \forall x_1 \, p(x_1) \leftarrow \exists x \, x_1 = x, \neg q(x), \ \forall x_1 \, q(x_1) \leftarrow \text{false} \} \]

\[\text{"=\} } \{ \forall x_1 \, p(x_1) \leftarrow \text{true, } \forall x_1 \, q(x_1) \leftarrow \text{false} \}.\]

Hence, \(\text{comp}(P) \models \forall x_1 \, p(x_1).\)

But there is no successful SLDNF-derivation of \(P \cup \{p(x_1)\}.\)
SLDNF-Resolution is Not Complete (IV): Unfairness

\[ P : \begin{align*}
  r & \leftarrow p, q \\
  p & \leftarrow p
\end{align*} \]

\[ \text{comp}(P) \supseteq \{ r \leftrightarrow p, q, \ p \leftrightarrow p, \ q \leftrightarrow \text{false} \} = \{ r \leftrightarrow \text{false}, \ q \leftrightarrow \text{false} \}. \]

Hence, \( \text{comp}(P) \models \neg r. \)

But there is no successful SLDNF-derivation of \( P \cup \{ \neg r \} \) w.r.t. leftmost selection rule.
Dependency Graphs

dependency graph $D_P$ of an extended program $P$

$\iff$

directed graph with labeled edges, where

- the nodes are the predicate symbols of $P$;
- the edges are either labeled by $+$ (positive edge) or by $-$ (negative edge);
- $p \rightarrow^+ q$ edge in $D_P$: $\iff$

  $P$ contains a clause $p(s_1, \ldots, s_m) \leftarrow L, q(t_1, \ldots, t_n), N$

- $p \rightarrow^- q$ edge in $D_P$: $\iff$

  $P$ contains a clause $p(s_1, \ldots, s_m) \leftarrow L, \neg q(t_1, \ldots, t_n), N$
Strict, Hierarchical, Stratified Programs

$P$ extended program, $D_P$ dependency graph of $P$, $p$, $q$ predicate symbols, $Q$ extended query:

- *$p$ depends evenly* (resp. *oddly*) on $q$ :⇔ there is a path in $D_P$ from $p$ to $q$ with an even–including 0–(resp. odd) number of negative edges

- $P$ is *strict* w.r.t. $Q$ :⇔ no predicate symbol occurring in $Q$ depends both evenly and oddly on a predicate symbol in the head of a clause in $P$

- $P$ is *hierarchical* :⇔ no cycle exists in $D_P$

- $P$ is *stratified* :⇔ no cycle with a negative edge exists in $D_P$
Restricted Completeness of SLDNF-Resolution (I)

Theorem ([Lloyd, 1987])
Let $P$ be a hierarchical and allowed program and $Q$ be an allowed query.

If $\text{comp}(P) \models Q^\theta$ for some $\theta$ such that $Q^\theta$ is ground, then there exists a successful SLDNF-derivation of $P \cup \{Q\}$ with CAS $\theta$.

Note:
Theorem does not hold, if arbitrary selection rule is fixed!
Selection rule has to be safe!
Restricted Completeness of SLDNF-Resolution (II)

Theorem ([Cavedon and Lloyd, 1989])
Let $P$ be a stratified and allowed program and $Q$ be an allowed query, such that $P$ is strict w.r.t. $Q$.

If $\text{comp}(P) \models Q\theta$ for some $\theta$ such that $Q\theta$ is ground, then there exists a successful SLDNF-derivation of $P \cup \{Q\}$ with CAS $\theta$.

Note:
Theorem does not hold if arbitrary selection rule is fixed!
Selection rule has to be safe and fair!
(extended) selection rule $\mathcal{R}$ is fair $\iff$
for every SLDNF-tree $\mathcal{F}$ via $\mathcal{R}$ and for every branch $\xi$ in $\mathcal{F}$:

- either $\xi$ is failed
- or for every literal $L$ occurring in a query of $\xi$, (some further instantiated version of) $L$ is selected within a finite number of derivation steps

Example:

- selection rule “select leftmost literal” is unfair
- selection rule “select leftmost literal to the right of the literals introduced at the previous derivation step, if it exists; otherwise select leftmost literal” is fair
Extended Consequence Operator

Let $P$ be an extended program and $I$ a Herbrand interpretation. Then

$$T_P(I) :\iff \{H \mid H \leftarrow B \in \text{ground}(P), I \models B\}$$

In case $P$ is a definite program, we know that

- $T_P$ is monotonic,
- $T_P$ is continuous,
- $T_P$ has the least fixpoint $M(P)$,
- $M(P) = T_P^w$.

In case of extended programs all of these properties are lost!
Extended $T_P$-Characterization (I)

Lemma 4.3 ([Apt and Bol, 1994])
Let $P$ be an extended program and $I$ a Herbrand interpretation. Then

$I \models P$ iff $T_P(I) \subseteq I$.

Proof:

$I \models P$

iff for every $H \leftarrow B \in \text{ground}(P)$: $I \models B$ implies $I \models H$

iff for every $H \leftarrow B \in \text{ground}(P)$: $I \models B$ implies $H \in I$

iff for every ground atom $H$: $H \in T_{P}(I)$ implies $H \in I$

iff $T_{P}(I) \subseteq I$
Extended $T_P$-Characterization (II)

Definition
Let $F$ and $\Pi$ be ranked alphabets of function symbols and predicate symbols, respectively, let $\neq \notin \Pi$ be a binary predicate symbol ("equality"), and let $I$ be a Herbrand interpretation for $F$ and $\Pi$.

Then $I_{=} := I \cup \{ (= (t, t) \mid t \in HU_F) \}$ is called a standardized Herbrand interpretation for $F$ and $\Pi \cup \{ = \}$.

Lemma 4.4 ([Apt and Bol, 1994])
Let $P$ be an extended program and $I$ a Herbrand interpretation. Then

$$I_{=} \models comp(P) \text{ iff } T_P(I) = I.$$
Extended $T_P$-Characterization (III)

Proof Idea of Lemma 4.4:

$I_\equiv \models \text{comp}(P)$

iff (since $I_\equiv$ is a model for standard axioms of equality and inequality)

for every ground atom $H : I \models (H \leftrightarrow \bigvee_{(H \leftarrow B) \in \text{ground}(P)} B)$

iff for every ground atom $H : H \in I \iff I \models B$ for some $H \leftarrow B \in \text{ground}(P)$

iff for every ground atom $H : H \in I \iff H \in T_P(I)$

iff $T_P(I) = I$
Completion may be Inadequate

\[ ill \leftarrow \neg ill, \text{infection} \]
\[ \text{infection} \leftarrow \]

\( \text{comp}(P) \supseteq \{ \text{ill} \leftarrow \neg \text{ill}, \text{infection} \quad , \quad \text{infection} \leftarrow \text{true} \} \)

is inconsistent (it has no models).
Hence, \( \text{comp}(P) \models \text{healthy} \).

But \( I = \{ \text{infection, ill} \} \) is (the only) Herbrand model of \( P \).
Hence, \( P \not\models \text{healthy} \).
Non-Intended Minimal Herbrand Models

\[ P_1: \quad p \leftarrow \neg q \]

\( P_1 \) has three Herbrand models:
\( M_1 = \{p\}, \ M_2 = \{q\}, \) and \( M_3 = \{p, q\} \)

\( P_1 \) has no least, but two minimal Herbrand models: \( M_1 \) and \( M_2 \)

However: \( M_1 \), and not \( M_2 \), is the “intuitive” model of \( P_1 \).
Supported Herbrand Interpretations

A Herbrand interpretation \( I \) is supported

\[ \iff \]

for every \( H \subseteq I \) there exists some \( H \leftarrow B \in \text{ground}(P) \) such that \( I \models B \)

(Intuition: \( B \) is an “explanation” for \( H \))

Example:

\( M_1 \) is a supported model of \( P_1 \). (\( \neg q \) is explanation for \( p \))

\( M_2 \) is no supported model of \( P_1 \).

Also note (cf. Lemma 4.3) that \( T_{P_1}(M_2) = \emptyset \subseteq M_2 \), but in particular \( T_{P_1}(M_1) = M_1 \).
Extended $T_P$-Characterization (IV)

Lemma 6.2 ([Apt and Bol, 1994])

Let $P$ be an extended program and $I$ a Herbrand interpretation. Then

$$I \models P \text{ and } I \text{ supported} \iff T_P(I) = I.$$ 

Proof Idea:

- $I \models P$ and $I$ supported
- iff for every $(H \leftarrow B) \in \text{ground}(P)$: $I \models B$ implies $I \models H$
- and for every $H \in I$:
  - $I \models \bigvee_{(H \leftarrow B) \in \text{ground}(P)} B$
- iff for every ground atom $H$:
  - $I \models (H \leftarrow \bigvee_{(H \leftarrow B) \in \text{ground}(P)} B)$
  - and $I \models (H \rightarrow \bigvee_{(H \leftarrow B) \in \text{ground}(P)} B)$
- iff for every ground atom $H$:
  - $I \models (H \leftrightarrow \bigvee_{(H \leftarrow B) \in \text{ground}(P)} B)$
- iff $I$ model for $\text{comp}(P)$
- iff (Lemma 4.4) $T_P(I) = I$
Non-Intended Supported Models

\[ P_2: \]
\[
p \leftarrow \neg q
\]
\[
q \leftarrow q
\]

\( P_2 \) has three Herbrand models:

\( M_1 = \{p\}, \; M_2 = \{q\}, \; \text{and} \; M_3 = \{p, q\} \)

\( P_2 \) has two supported Herbrand models: \( M_1 \) and \( M_2 \)

However: \( M_1 \), and not \( M_2 \), is the “intended” model of \( P_2 \).

\( M_1 \) is called the standard model of \( P_2 \) (cf. slide VII/35).
Stratifications

$P$ extended program and $D_P$ dependency graph of $P$:

- **predicate symbol** $p$ **defined in** $P$:
  $\iff P$ contains a clause $p(t_1, \ldots, t_n) \leftarrow B$

- $P_1 \cup \ldots \cup P_n = P$ **stratification** of $P$ $\iff$
  - $P_i \neq \emptyset$ for every $i \in [1, n]$
  - $P_i \cap P_j = \emptyset$ for every $i, j \in [1, n]$ with $i \neq j$
  - for every $p$ defined in $P_i$ and edge $p \xrightarrow{+} q$ in $D_P$: $q$ not defined in $\bigcup_{j=i+1}^{n} P_j$
  - for every $p$ defined in $P_i$ and edge $p \xrightarrow{-} q$ in $D_P$: $q$ not defined in $\bigcup_{j=i}^{n} P_j$

**Lemma 6.5 ([Apt and Bol, 1994])**

An extended program is stratified if and only if it admits a stratification.

**Note:** A stratified program may have different stratifications.
Example (I)

\(P: \)

\[
\begin{align*}
    \text{zero}(0) & \leftarrow \\
    \text{positive}(x) & \leftarrow \text{num}(x), \neg \text{zero}(x) \\
    \text{num}(0) & \leftarrow \\
    \text{num}(s(x)) & \leftarrow \text{num}(x)
\end{align*}
\]

\(P_1 \cup P_2 \cup P_3\) is a stratification of \(P\), where

\[
\begin{align*}
P_1 &= \{\text{num}(0) \leftarrow, \text{num}(s(x)) \leftarrow \text{num}(x)\} \\
P_2 &= \{\text{zero}(0) \leftarrow\} \\
P_3 &= \{\text{positive}(x) \leftarrow \text{num}(x), \neg \text{zero}(x)\}
\end{align*}
\]
Example (II)

\[
P: \quad \begin{align*}
&num(0) \leftarrow \\
&num(s(x)) \leftarrow num(x) \\
&even(0) \leftarrow \\
&even(x) \leftarrow \neg odd(x), \ num(x) \\
&odd(s(x)) \leftarrow even(x)
\end{align*}
\]

\[P \text{ admits no stratification.}\]
Standard Models (Stratified Programs)

\[ I \mid \Pi \Leftrightarrow I \cap \{ p(t_1, \ldots, t_n) \mid p \in \Pi, t_1, \ldots, t_n \text{ ground terms} \} \]

Let \( P_1 \cup \ldots \cup P_n \) be stratification of extended program \( P \).

\[ M_1 : \Leftrightarrow \text{least Herbrand model of } P_1 \text{ such that} \]
\[ M_1 \mid \{ p \mid p \text{ not defined in } P \} = \emptyset \]

\[ M_2 : \Leftrightarrow \text{least Herbrand model of } P_2 \text{ such that} \]
\[ M_2 \mid \{ p \mid p \text{ defined nowhere or in } P_1 \} = M_1 \]

\[ \vdots \]

\[ M_n : \Leftrightarrow \text{least Herbrand model of } P_n \text{ such that} \]
\[ M_n \mid \{ p \mid p \text{ defined nowhere or in } \bigcup_{i=1}^{n-1} P_i \} = M_{n-1} \]

We call \( M_P = M_n \) the standard model of \( P \).
Example (I)

Let $P_1 \cup P_2 \cup P_3$ with

\[ P_1 = \{ \text{num}(0) \leftarrow, \text{num}(s(x)) \leftarrow \text{num}(x) \} \]
\[ P_2 = \{ \text{zero}(0) \leftarrow \} \]
\[ P_3 = \{ \text{positive}(x) \leftarrow \text{num}(x), \neg \text{zero}(x) \} \]

be stratification of $P$.

Then:

\[ M_1 = \{ \text{num}(t) \mid t \in HU_{\{s,0\}} \} \]
\[ M_2 = \{ \text{num}(t) \mid t \in HU_{\{s,0\}} \} \cup \{ \text{zero}(0) \} \]
\[ M_3 = \{ \text{num}(t) \mid t \in HU_{\{s,0\}} \} \cup \{ \text{zero}(0) \} \cup \{ \text{positive}(t) \mid t \in HU_{\{s,0\}} - \{0\} \} \]

Hence $M_P = M_3$ is the standard model of $P$. 
Properties of Standard Models

Theorem 6.7 ([Apt and Bol, 1994])
Consider a stratified program \( P \). Then,
- \( M_P \) does not depend on the chosen stratification of \( P \),
- \( M_P \) is a minimal model of \( P \),
- \( M_P \) is a supported model of \( P \).

Corollary
For a stratified program \( P \), \( \text{comp}(P) \) admits a Herbrand model.
Objectives

- First-Order Formulas and Logical Truth
- The Completion semantics
- Soundness and restricted completeness of SLDNF-Resolution
- Extended consequence operator
- An alternative semantics: Standard models