Lecture 7: Reasoning with Incomplete Knowledge

2-AIN-144/2-IKV-131 Knowledge Reperesentation & Reasoning

Martin Baláž, Martin Homola

Department of Applied Informatics
Faculty of Mathematics, Physics and Informatics
Comenius University in Bratislava

4 Apr 2013

Example

Locally unstratified normal logic program:

$$
\begin{aligned}
man(dilbert) &\leftarrow \\
single(dilbert) &\leftarrow man(dilbert), not\ husband(dilbert) \\
husband(dilbert) &\leftarrow man(dilbert), not\ single(dilbert)
\end{aligned}
$$

Two intuitive meanings:

$$
\begin{aligned}
M_1 &= \{man(dilbert), single(dilbert)\} \\
M_2 &= \{man(dilbert), husband(dilbert)\}
\end{aligned}
$$

Stable Models

Definition (Program Reduct)

Let P be a normal logic program and I be an interpretation. The reduct of P (with respect to I) is a definite logic program P^I obtained from P by

- removing rules containing a default literal L in the budy such that $I \not\models L$
- removing remaining default literals L, i.e. default literals with $I \models L$

Definition (Stable Model)

An interpretation I is a *stable model* of a normal logic program P iff I is the smallest model of P^I.

Stable Models and Negative Cycles

Example (Negative Cycles of Even Length)

Generator:

$$\begin{aligned} a &\leftarrow \textit{not } b \\ b &\leftarrow \textit{not } a \end{aligned}$$

Two stable models:

$$\begin{aligned} M_1 &= \{a\} \\ M_2 &= \{b\} \end{aligned}$$

Example (Negative Cycles of Odd Length)

Constraint:

$$a \leftarrow \textit{not } a$$

No stable model!

Properties of Stable Models

Proposition

Stable models of a normal logic program are minimal.

Proposition

Let P be a locally stratified normal logic program. Then the only stable model of P coincides with the locally stratified model.

Stable Models in Practice

- Problem Input
- Problem Specification
  1. Generate possible candidates
  2. Test if they are solutions

Stable Models in Practice

```
% problem input
col(1..8).
row(1..8).

% generate
at(C,R) :- col(C), row(R), not empty(C,R).
empty(C,R) :- col(C), row(R), not at(C,R).
at_col(C) :- at(C,R).
at_row(R) :- at(C,R).

% test
i :- at(X, Y1), at(X, Y2), Y1 != Y2, not i.
i :- at(X1, Y), at(X2, Y), X1 != Y2, not i.
i :- col(C), not at_col(C), not i.
i :- row(R), not at_row(R), not i.
```

Example

Normal logic program:

$$
\begin{aligned}
man(dilbert) &\leftarrow \\
single(dilbert) &\leftarrow man(dilbert), not\ husband(dilbert) \\
husband(dilbert) &\leftarrow man(dilbert), not\ single(dilbert)
\end{aligned}
$$

Three possible meanings:

$$
\begin{aligned}
M_1 &= \{man(dilbert), single(dilbert), not\ husband(dilbert)\} \\
M_2 &= \{man(dilbert), husband(dilbert), not\ single(dilbert)\} \\
M_3 &= \{man(dilbert)\}
\end{aligned}
$$

3-Valued Interpretation

Definition (3-Valued Interpretation)

A *3-valued interpretation* is a pair $I = (T, F)$ where T and F are disjoint subsets of the Herbrand base $\mathcal{B}$.

Given a 3-valued interpretation $I = (T, F)$, an atom $A \in \mathcal{B}$ is

- *false* if $A \in F$
- *unknown* if $A \notin T \cup F$
- *true* if $A \in T$

A 3-valued interpretation $I = (T, F)$ can be equivalently viewed as

- a mapping $I \colon \mathcal{B} \mapsto \{0, \frac{1}{2}, 1\}$ such that

$$I(A) = \begin{cases} 0 & \text{if } A \in F \\ \frac{1}{2} & \text{if } A \notin T \cup F \\ 1 & \text{if } A \in T \end{cases}$$

- a set $T \cup \mathit{not}\ F$ where $\mathit{not}\ F = \{\mathit{not}\ A \mid A \in F\}$

Truth Valuation

Definition (Truth Valuation)

The *truth valuation* corresponding to a 3-valued interpretation I is a mapping $val_I : \mathcal{L} \mapsto \{0, \frac{1}{2}, 1\}$ defined as follows:

- if A is a ground atom, then $val_I(A) = I(A)$
- if ϕ is a closed formula, then $val_I(\neg\phi) = 1 - val_I(\phi)$
- if ϕ and ψ are closed formulae, then

$$val_I(\phi \wedge \psi) = \min\{val_I(\phi), val_I(\psi)\}$$
$$val_I(\phi \vee \psi) = \max\{val_I(\phi), val_I(\psi)\}$$
$$val_I(\phi \leftarrow \psi) = \begin{cases} 1 & \text{if } val_I(\phi) \geq val_I(\psi) \\ 0 & \text{if } val_I(\phi) < val_I(\psi) \end{cases}$$

- if $\phi(X)$ is a formula with one unbounded variable X, then

$$val_I(\forall X \phi(X)) = \min\{val_I(\phi(t)) \mid t \in \mathcal{U}\}$$
$$val_I(\exists X \phi(X)) = \max\{val_I(\phi(t)) \mid t \in \mathcal{U}\}$$

Truth and Knowledge Orderings

Natural ordering on truth values: $0 \leq \frac{1}{2} \leq 1$

Two kinds of orderings on interpretations $I = (T_I, F_I)$ and $J = (T_J, F_J)$:

- Truth ordering

$$I \preceq_t J \text{ iff } T_I \subseteq T_J \text{ and } F_I \supseteq F_J$$

- Knowledge ordering

$$I \preceq_k J \text{ iff } T_I \subseteq T_J \text{ and } F_I \subseteq F_J$$

An interpretation I is t-minimal (resp. k-minimal) iff there does not exist an interpretation $J \neq I$ such that $J \preceq_t I$ (resp. $J \preceq_k I$).

An interpretation I is t-least (resp. k-least) iff for all interpretations $J \neq I$ holds $I \preceq_t J$ (resp. $I \preceq_k J$).

Positive Logic Programs

Definition (Immediate Consequence Operator)

Let P be a positive logic program and I be a 3-valued interpretation. By $T_P(I)$ we denote an interpretation defined as follows:

$$T_P(I)(A) = \max\{val_I(body(r)) \mid r \in P, head(r) = A\}$$

The immediate consequence operator can be equivalently defined as

- $T_P(I)(A) = 1$ if there exists a rule $r \in P$ such that $head(r) = A$ and $val_I(body(r)) = 1$
- $T_P(I)(A) = \frac{1}{2}$ if there exists a rule $r \in P$ such that $head(r) = A$ and $val_I(body(r)) = \frac{1}{2}$ but there does not exist a rule $r \in P$ such that $head(r) = A$ and $val_I(body(r)) = 1$
- $T_P(I)(A) = 0$ if there does not exist a rule $r \in P$ such that $head(r) = A$ and $val_I(body(r)) = 1$ or $val_I(body(r)) = \frac{1}{2}$

Immediate Consequence Operator

Proposition

Every positive logic program P has the t-least model.

Proposition

Let P be a positive logic program. Then $T_P \uparrow \omega(\emptyset, \mathcal{B})$ is the t-least model of P.

Example

$$\begin{aligned} c &\leftarrow \\ a &\leftarrow c, \mathbf{u} \\ b &\leftarrow b, \mathbf{u} \end{aligned}$$

Partial Stable Model

Definition (Program Reduct)

Let P be a normal logic program and $I = (T, F)$ be a 3-valued interpretation. The reduct of P (with respect to I) is a non-negative logic program P^I obtained from P by replacing each default literal *not* A with

- a constant f if $A \in T$
- a constant t if $A \in F$
- a constant u if $A \notin T \cup F$

Definition (Partial Stable Model)

A 3-valued interpretation I is a *partial stable model* of a normal logic program P iff I is the t-least model of P^I.

Partial Stable Model and Negative Cycles

Example (Negative Cycles of Even Length)

Generator:

$$\begin{aligned} a &\leftarrow \textit{not}\ b \\ b &\leftarrow \textit{not}\ a \end{aligned}$$

Three partial stable models:

$$M_1 = \{a, \textit{not}\ b\},\ M_2 = \{b, \textit{not}\ a\}, \text{ and } M_3 = \emptyset.$$

Example (Negative Cycles of Odd Length)

Constraint:

$$a \leftarrow \textit{not}\ a$$

One partial stable model:

$$M_1 = \emptyset$$

Properties of Partial Stable Models

Proposition

Partial stable models of a normal logic program are t-minimal.

Proposition

Let P be a normal logic program. Stable models of P coincide with 2-valued partial stable models of P.

Proposition

Let P be a locally stratified normal logic program. Then the only partial stable model coincides with the locally stratified model.

Well-Founded Model and Maximal Stable Model

Definition (Well-Founded Model)

Let P be a normal logic program. The well-founded model of P is the k-least partial stable model of P.

Definition (Maximal Stable Model)

Let P be a normal logic program. A maximal stable model of P is a k-maximal partial stable model of P.

Proposition

Let P be a normal logic program. Then each 2-valued partial stable model of P is a maximal stable model of P.

Iterative Characterization of Well-Founded Model

Progressive Immediate Consequence Operator:

$$T_P^*(I) = T_P(I) \cup I$$

Proposition

Let P be a normal logic program. Then $T_P^* \uparrow \omega(\emptyset, \emptyset)$ is the well-founded model of P.

Example

$$\begin{array}{lll} a & \leftarrow & \\ c & \leftarrow & not\ b, a \\ b & \leftarrow & not\ c \\ e & \leftarrow & not\ d \\ f & \leftarrow & e \\ f & \leftarrow & not\ a \end{array}$$