Fixpoint Semantics for Logic Programming

See Melvin Fitting 2002
- If you read and understand Fitting’s survey paper you have learned a sufficient amount of knowledge in this class.

- Note that some things are given a slightly different name – but mean the same as things we have learned here.
Simplifications

Given a logic program P with clauses C,
Construct P* with clauses C* by
- replace „A :-    .“ by „A :- true“,
- ground instantiate all clauses from C,
- if the ground atom A is not the head of any member of P*,
  add „A :- false“.

Example:

P(x) :- Q(x), R(x).
R(a).

Becomes P*

R(a) :- true.
P(a) :- Q(a), R(a).
Q(a) :- false.
Truth ordering

Minimize with respect to order, i.e. default to false:

Definition: The space \{false,true\} is given the truth ordering false \(\prec_t\) true, with \(x \prec_t y\) not holding in any other case. We use \(\leq_t\) as usual for \(\prec_t\) or =.

\[
\text{false} \quad \prec_t \quad \text{true}
\]

This ordering is extended to interpretations pointwise:

\(I_1 \leq_t I_2\) if and only if \(I_1(A) \leq_t I_2(A)\) for all ground atoms \(A\).
Side remark

$T_{P \downarrow \omega}$ is not necessarily the biggest fixpoint, but $T_{P \downarrow \alpha}$ for some $\alpha > \omega$
We know: Normal programs do not have one smallest fixpoint

Approach:
1. Consider two (or more) fixpoints
2. Consider multi-valued interpretations
Partial interpretations

We know: A classical interpretation assigns every ground atom a truth value from \{true, false\}.

Consider:
- P :- P.
- Q.

Smallest fixpoint: \{Q\}

Largest fixpoint: \{Q,P\}

Idea:
- What is true in both fixpoints is true.
- What is true in one fixpoint, but false in the other is uncertain. 

**Partial interpretation**

**Definition**: A partial valuation is a mapping $I$ from the set of ground atoms to the set $\{\bot, \text{false, true}\}$, meeting the conditions

- $I(\text{false}) = \text{false}$
- and
- $I(\text{true}) = \text{true}$

We often refer to partial valuations as three valued.
Three valued knowledge ordering

**Definition:** The space \( \{ \bot, \text{false}, \text{true} \} \) is given a knowledge ordering \( \bot <_k \text{false}, \bot <_k \text{true} \), with \( x <_k y \) not holding in any other case. Then \( \leq_k \) is defined as usual.

\[
\begin{align*}
\text{false} & \quad \leq_k \quad \text{true} \\
\bot & \quad <_k \\
& \quad <_k \\
& \quad <_k \\
& \quad <_k
\end{align*}
\]

The ordering is again extended to partial interpretations pointwise:

\[ I_1 \leq_k I_2 \text{ iff } I_1(A) \leq_k I_2(A) \text{ for all ground atoms } A. \]
Alternative notation

Describe three-valued interpretation $I$ as pair $(T,F)$ of true ground atoms $T$ and false ground atoms $F$.

Then $I_1 \leq_k I_2$ iff $T_1 \subseteq T_2$ and $F_1 \subseteq F_2$ ("$I_2$ knows more than $I_1$"
Mapping $\phi_P$

**Definition.** Let $P$ be a normal program. An associated mapping $\phi_P$, from partial interpretations to partial interpretations, is defined as follows.

$$\phi_P(I)=J$$

where $J$ is the unique partial interpretation determined by the following: for a ground atom $A$,

1. $J(A)=$true if there is a general ground clause $A \leftarrow B_1, \ldots, B_n$ in $P^*$ with head $A$, such that $I(B_1)=$true and $I(B_2)=$true and $\ldots$ and $I(B_n)=$true.
2. $J(A)=$false if for every general ground clause $A \leftarrow B_1, \ldots, B_n$ in $P^*$ with head $A$, $I(B_1)=$false, or $I(B_2)=$false, or $\ldots$, $I(B_n)=$false.
3. $J(A)=$⊥ otherwise.
### Kleene’s strong three-valued logic

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>A ∧ B</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>true</td>
<td>⊥</td>
<td>⊥</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>⊥</td>
<td>false</td>
</tr>
<tr>
<td>⊥</td>
<td>true</td>
<td>⊥</td>
</tr>
<tr>
<td>⊥</td>
<td>false</td>
<td>⊥</td>
</tr>
<tr>
<td>⊥</td>
<td>⊥</td>
<td>⊥</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>A ∨ B</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>true</td>
<td>⊥</td>
<td>true</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>⊥</td>
<td>⊥</td>
</tr>
<tr>
<td>⊥</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>⊥</td>
<td>false</td>
<td>⊥</td>
</tr>
<tr>
<td>⊥</td>
<td>⊥</td>
<td>⊥</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>¬ A</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>True</td>
</tr>
<tr>
<td>⊥</td>
<td>⊥</td>
</tr>
</tbody>
</table>
Monotonicity of $\phi_P$

**Proposition:** For a general program $P$, the operator $\phi_P$ is monotone with respect to $\leq_k$:

$I_1 \leq_k I_2$ implies $\phi_P(I_1) \leq_k \phi_P(I_2)$.

**Note:** The smallest fixed point of $\phi_P$ supplies the Fitting semantics (also called Kripke-Kleene semantics) with

\[
\begin{align*}
\phi_P \uparrow 0 &= \bot \\
\phi_P \uparrow \alpha + 1 &= \phi_P(\phi_P \uparrow \alpha) \\
\phi_P \uparrow \lambda &= \bigcup \{\phi_P \uparrow \alpha | \alpha < \lambda\}
\end{align*}
\]

with $\lambda$ being a limit ordinal, but $\bigcup$ is with respect to $\leq_k$.
Differences and Commonalities between $T_P$ and $\phi_P$

Q :- Q.  
Fixpoint for $T_P$ is $\{\}$, i.e. $I(Q) = false$

Q :- not Q.  
No fixpoint.

Q :- Q.  
Fixpoint for $\phi_P$ is $\{\},\{\}$, i.e. $I(Q) = \bot$.

Q :- not Q.  
Fixpoint for $\phi_P$ is $\{\},\{\}$, i.e. $I(Q) = \bot$.

**Proposition**: Let $P$ be a definite program. Let $I_k$ be the smallest fixed point of $P$ (with respect to $\leq_k$), and let $j_t$ and $J_t$ be the smallest and the biggest fixed points of $T_P$ (with respect to $\leq_t$). Then, for a ground atom $A$,

If $j_t(A) = J_t(A)$, then $I_k(A)$ has this common value.

If $j_t(A) \neq J_t(A)$ then $I_k(A) = \bot$. 

Belnap’s four-valued Logic
Belnap’s four-valued Logic

Knowledge and truth ordering

Default f: closed world, default ⊥: open world
Truth values for Belnap’s logic

$\bot = \{\}$
false = \{false\}
true=\{true\}
T=\{true, false\}

$\leq_k$ is now simply defined by $\subseteq$ over $I=(T,F)$

$\leq_k$ is a lattice, $\leq_t$ is a lattice; their combination is a bi-lattice.
Logical connectives formalizable as (infinitely distributive) functions on this ordering:

\[ a \lor b = \sup_t(a, b) \]
\[ a \land b = \inf_t(a, b) \]
\[ a \oplus b = \sup_k(a, b) \]
\[ a \otimes b = \inf_k(a, b) \]

\[ \neg a = \begin{cases} 
  f, & \text{if } a = t \\ 
  t, & \text{if } a = f \\ 
  a, & \text{otherwise} 
\end{cases} \]

Four binary operations, all distributive laws hold.
Newly define interpretations

\[ I(A \land B) = I(A) \land I(B) \]
\[ I(A \otimes B) = I(A) \otimes I(B) \]

etc.

**Definition.** Let \( P \) be a normal program. Let \( P^* \) be its grounding as defined before. Let \( P^{**} \) be the completion of \( P^* \) (with possibly infinitely long ground clauses).

\[ \phi_P(I) = J, \]

where \( J \) is the unique interpretation determined by the following:

- if \( A \leftarrow B \) is in \( P^{**} \), then \( J(A) = I(B) \),

where we use Belnap’s logic to evaluate \( I(B) \).
Proposition 19: Let $i_t$ and $I_t$ be the smallest and biggest fixed points of the four-valued operator $\phi_P$ with respect to the $\leq_t$ ordering, where $P$ is a definite program. Likewise, let $j_k$ and $J_k$ be the smallest and biggest fixed points of $\phi_P$ with respect to the $\leq_k$ ordering.

We can state that:

$$j_k = i_t \otimes I_t$$

$$J_k = i_t \oplus I_t$$

$$i_t = j_k \land J_k$$

$$I_t = j_k \lor J_k$$
On the Semantics of Trust on the Semantic Web

Simon Schenk
ISWC 2008, Karlsruhe, Germany
“Quantum of Solace“

„Olga Kurylenko toughest Bond-Girl ever.“

olga: GoodActor
qos: GoodAction

WELT ONLINE

„Olga Kurylenko flat like a stale Martini.“

olga: ¬GoodActor
qos: GoodAction

Spiegel U Welt globally inconsistent.

To judge, whether Quantum of Solace is a good action movie, we need

paraconsistent reasoning:

olga: GoodActor → T  qos: GoodAction → ⊤
"Quantum of Solace"

**SPIEGEL ONLINE**
olga:GoodActor
qos:GoodAction

**WELT ONLINE**
Olga:¬ GoodActor
qos:GoodAction

**Mail Online**
Olga:¬ GoodActor
Qos:¬ GoodAction
daniel:GoodActor

Trust in News Sources

qos:GoodAction → t_{SO,W}
olga:GoodActor → T_{SO,W,M}

General Trust Order
Other Examples

- Collaborative Ontology Editing
  - Editors trusted differently
  - Personal relation
  - Even if possible, strict trust order for employees might be illegal

- Caching
  - Distinguish between certain and possibly outdated information

...
Overview

- Motivation
- Logical Bilattices
- „Trust Bi-Lattices“
- SROIQ on bilattices
- Outlook and Conclusion
Logical Bi-lattices

Knowledge and truth ordering

Logical connectives formalizable as (infinitely distributive) functions on this ordering:

\[ a \lor b = \sup_t(a, b) \]
\[ a \land b = \inf_t(a, b) \]
\[ a \oplus b = \sup_k(a, b) \]
\[ a \otimes b = \inf_k(a, b) \]

\[ \neg a = \begin{cases} f, & \text{if } a = t \\ t, & \text{if } a = f \\ a, & \text{otherwise} \end{cases} \]

Default f: closed world, default \( \perp \): open world
Other bilattices

- e.g. **designed** for default reasoning
**Generate** logical bilattice based on trust order

Lukasiewicz:
Derive (distributive) bilattice from two (distributive) lattices as follows:

Given two distributive lattices $L_1$ and $L_2$, create a bilattice $L$, where the nodes have values from $L_1 \times L_2$, such that:

- $(a,b) \leq_k (x,y)$ iff $a \leq_{L_1} x \land b \leq_{L_2} y$
- $(a,b) \leq_t (x,y)$ iff $a \leq_{L_1} x \land y \leq_{L_2} b$

E.g. FOUR = $\{0,t\} \times \{0,f\}$:

\[
\begin{array}{c}
(0,0) \quad (0,f) \\
(t,0) \quad (t,f)
\end{array}
\]
Generate Lattice from Trust Order

Derive $L_1$ and $L_2$ from trust order $T$ over information sources $S_i$:

$L_1 = L_2 = \{(f_i, t_i) | (f_i, t_i) \in S \} \cup \{(t_i, t_j) | (i, j) \in T \} \cup \{(f_i, f_j) | (j, i) \in T \}$

Problem: $t_b \oplus t_c =? t_\infty$
Augmented Trust Order

Derive $L_1$ and $L_2$ from \textit{augmented} trust order $T$ over information sources $S$:

$$L_1 = L_2 = \{(f_i, t_i) | i \in S \} \cup \{(t_i, t_j) | (i, j) \in T \} \cup \{(f_i, f_j) | (j, i) \in T \}$$
Use trust order to derive a logical bilattice.

Example for comparable information sources:
leads to:

a) comparable sources

leads to:

b) incomparable sources
Application: Inconsistency Resolution

Reasons for Inconsistencies:
\[ tv(a) = t_x : \quad a \leftarrow A \]
\[ tv(a) = f_y : \quad a \leftarrow B \]

\[ f_x \land t_y = T_{xy} \text{ (inconsistent)} \]

Subscript of \( T \) reflects the maximally and minimally trusted information sources, which cause the inconsistency.

Possible resolution: Find minimal inconsistent subontology
Drop minimally trusted axioms.
Application: Inconsistency Resolution

Minimally and maximally trusted source contributing to the inconsistency

Drop minimally trusted axioms

Not possible for olga:GoodActor!
Conclusion and Future Work

- Go watch „Quantum of Solace“ (Simon’s recommendation)

- Trust based reasoning on logical bilattices
  - Derived from any partial trust order
  - Applicable to a broad variety of languages

- Operationalization
  - Efficient debugging of large ontologies based on differently trusted and/or time-stamped ontology changes: