Institution-independent logic programming paradigms

Dorel Lucanu
dlucanu@info.uaic.ro

Universitatea “Alexandru Ioan Cuza”, Iași

Joint work with Yuan Fang Li and Jin Song Dong from NUS
and Traian Șerbănuță from UIUC
In memory of Joseph …
Motivation: the jungle of SW Languages

- SWRL FOL
- OWL Full
- OWL
- F-Logic
- OWL Light
- RDF
- SHIQ
- OWL Full
- SHIQ
- DAML-OIL
- WRL
- DAML-OIL
- Datalog
- SWRL
- OWL Light
- OWL DL
- RDF Schema
- SHOIN
- OWL DL
- DLP
- OWL Flight
Motivation

- an integrating mathematical structure for Semantic Web Languages (SWL)
- translating Web ontologies into other formalisms
- a safe way to walk in the jungle
- disputes on layering of SWL
- Open World Assumption (OWA) vs Closed World Assumption (CWA)
- soundness of the reasoners for Web ontologies
- finding the real meaning of Semantic Web Stack
Outline

- institution independent logic programming paradigms
  - institutions
  - institution independent first order logic
  - institution independent logic programming paradigms
    - open world assumption (OWA)
    - closed world assumption (CWA)
- logic programming viewpoints on web ontologies
  - institution of description logic
  - description logic and logic programming paradigms
- an institutional approach of SW stack
  - institutional meaning of RDF layer
  - institutional meaning of ontology layer
  - institutional meaning of layering
- conclusion
Outline

- institution independent logic programming paradigms
  - institutions
- institution independent first order logic
- institution independent logic programming paradigms
  - open world assumption (OWA)
  - closed world assumption (CWA)
- logic programming viewpoints on web ontologies
  - institution of description logic
  - description logic and logic programming paradigms
- an institutional approach of SW stack
  - institutional meaning of RDF layer
  - institutional meaning of ontology layer
  - institutional meaning of layering
- conclusion
Institutions

J. Goguen, R. Burstal. 1984

- formalize the notion of "a logic"
- study the properties of a logic
  - representation
- implementation
- translation of logics
Institutions: ingredients

Informally an institution consists of

- a collection of **signatures**: vocabularies
- a collection of **models**: structures interpreting the symbols (names) from a signature
- a collection of **sentences**: formulas built with symbols from a signature expressing specific properties
- a **satisfaction relation**: says when a given sentence holds in a given model (both corresponding to the same signature)

Formally, \( \mathcal{S} = (\text{Sign}, \text{Mod}, \text{sen}, \models) \), where

- \( \text{Sign} \) is the category of signatures,
- \( \text{Mod} : \text{Sign}^{op} \to \text{Cat} \)
- \( \text{sen} : \text{Sign} \to \text{Set} \)
- \( \models = (\models_\Sigma \mid \Sigma \in \text{Sign}) \), \( \models_\Sigma \subseteq \text{Mod}(\Sigma) \times \text{sen}(\Sigma) \)
Institution of first order logic ($\text{FOL}$): signatures

$$\Sigma = (OP, RL)$$

*OP* - operation symbols

*RL* - relation (predicate) symbols

<table>
<thead>
<tr>
<th>NAT</th>
<th>LIST</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Sigma_{NAT} = (OP_{NAT}, RL_{NAT})$</td>
<td>$\Sigma_{LIST} = (OP_{LIST}, RL_{LIST})$</td>
</tr>
<tr>
<td>$OP_{NAT} = {\text{zero, } s}$</td>
<td>$OP_{LIST} = {\text{nil, } a, b, \text{cons}}$</td>
</tr>
<tr>
<td>$RL_{NAT} = {\text{plus}}$</td>
<td>$RL_{LIST} = {\text{cat}}$</td>
</tr>
</tbody>
</table>

$\phi := \Sigma_{NAT} \rightarrow \Sigma_{LIST} = (\phi^\text{op} : OP_{NAT} \rightarrow OP_{LIST}, \phi^\text{rl} : RL_{NAT} \rightarrow RL_{LIST})$

- $\phi^\text{op}(\text{zero}) = \text{nil}$
- $\phi^\text{op}(s(x)) = \text{cons}(a, \phi^\text{op}(x))$
- $\phi^\text{rl}(\text{plus}(x, y, z)) = \text{cat}(\phi^\text{op}(x), \phi^\text{op}(y), \phi^\text{op}(z))$
### FOL: models

<table>
<thead>
<tr>
<th>NAT</th>
<th>LIST</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M = \omega$</td>
<td>$M' = {a, b}^*$</td>
</tr>
<tr>
<td>$M_{zero} = 0$</td>
<td>$M'_\text{nil} = \varepsilon$</td>
</tr>
<tr>
<td>$M_s(n) = n + 1$</td>
<td>$M_{\text{cons}}(x, w) = xw$</td>
</tr>
<tr>
<td>$M_{\text{plus}}(n', n', n'') \equiv n'' = n + n'$</td>
<td>$M'_{\text{cat}}(w, w', w'') \equiv w'' = ww'$</td>
</tr>
</tbody>
</table>

$\phi := \Sigma_{\text{NAT}} \rightarrow \Sigma_{\text{LIST}}$

$\text{Mod}(\phi) : \text{Mod(\text{LIST})} \rightarrow \text{Mod(\text{NAT})}$

$\text{Mod}(\phi)(M') = M'\upharpoonright_\phi$ (by notation)

- $M'\upharpoonright_\phi = M'$ (as carrier sets)
- $(M'\upharpoonright_\phi)_{\text{zero}} = \varepsilon$
- $(M'\upharpoonright_\phi)_s(w) = aw$
- $(M'\upharpoonright_\phi)_{\text{plus}}(w, w', w'') \equiv w'' = ww'$
\( F_{\text{NAT}} \)

\[
(\forall n)\text{plus}(0, n, n) \\
(\forall n, n', n'') \text{plus}(n, n', n'') \rightarrow \text{plus}(s(n), n', s(n''))
\]

\( F_{\text{LIST}} \)

\[
(\forall w) \text{cat}(\text{nil}, w, w) \\
(\forall x, w, w', w'') \text{cat}(w, w', w'') \rightarrow \text{cat}(\text{cons}(x, w), w', \text{cons}(x, w''))
\]

\( \phi \) is extended to sentences

\[
\text{sen}(\phi) : \text{sen}(\Sigma_{\text{NAT}}) \rightarrow \text{sen}(\Sigma_{\text{LIST}})
\]

e.g., \( \text{sen}(\phi)((\forall n)\text{plus}(0, n, n)) = (\forall w)\text{cat}(\text{nil}, w, w) \)
**FOL**: satisfaction relation

\[ M \models \sum_{\text{nat}} (\forall \ n) \text{plus}(0, \ n, \ n) \]
\[ M \models \sum_{\text{nat}} (\forall \ n, \ n', \ n'') \text{plus}(n, \ n', \ n'') \rightarrow \text{plus}(s(n), \ n', \ s(n'')) \]
\[ M' \models \sum_{\text{list}} (\forall \ w) \text{cat}(\text{nil}, \ w, \ w) \]
\[ M' \models \sum_{\text{list}} (\forall \ x, \ w, \ w', \ w'') \text{cat}(w, \ w', \ w'') \rightarrow \text{cat}(\text{cons}(x, \ w), \ w', \ \text{cons}(x, \ w'')) \]

It is the subject of the **satisfaction condition** which expresses the invariance of truth under change of notation.

\[ M' \models \sum_{\text{list}} (\forall \ w) \text{cat}(\text{nil}, \ w, \ w) \text{iff} \ M' \upharpoonright \phi \models \sum_{\text{nat}} (\forall \ n) \text{plus}(0, \ n, \ n) \]

**Notation** \(\widehat{\text{FOLR}}\) is similar to \(\widehat{\text{FOL}}\) excepting OP(\(\Sigma\)) that includes only constants and relations for each \(\Sigma\), and \(\widehat{\text{HL}}\) is the institution of **Horn Logic**.
Institutions: Specifications and Theories

- A **specification** is a pair \((\Sigma, F)\), where \(\Sigma\) is a signature and \(F\) is a set of sentences.
  
e.g., \(NAT = (\Sigma_{NAT}, F_{NAT})\), \(LIST = (\Sigma_{LIST}, F_{LIST})\)

- Semantical consequences: \((\Sigma, F) \models \varphi\) iff 
  \[(\forall M)(M \models_\Sigma F \Rightarrow M \models_\Sigma \varphi)\]

- A **theory** is a specification \((\Sigma, F)\) s.t.
  \[(\forall \varphi)(\Sigma, F) \models \varphi \Rightarrow \varphi \in F\]

- The inclusion \(\text{Th} \rightarrow \text{Spec}\) is an equivalence of categories

- Theoroidal (spec-oidal) institutions \(\mathfrak{S}^{th}\):
  - Signatures are theories (specifications) \((\Sigma, F)\)
  - A \((\Sigma, F)\)-sentence is a \(\Sigma\)-sentence
  - \((\Sigma, F)\)-models are \(\Sigma\)-models satisfying \(F\)
  - \(M \models_{(\Sigma, F)} \varphi\) iff \(M \models_\Sigma \varphi\)
Relating Institutions

**morphism**: capture the way in which a “richer” institution is built over a “simpler” one

e.g., \((\Phi, \beta, \alpha) : \widehat{\text{FOL}} \rightarrow \widehat{\text{FOLR}}\)

\(\Phi : \text{Sign}(\widehat{\text{FOL}}) \rightarrow \text{Sign}(\widehat{\text{FOLR}})\) forgets non-constant operations

\(\beta = (\beta_\Sigma : \text{Mod}(\widehat{\text{FOL}})(\Sigma) \rightarrow \text{Mod}(\widehat{\text{FOLR}})(\Phi(\Sigma)) \mid \Sigma \in |\Sigma(\widehat{\text{FOL}})|),\)

\(\beta_\Sigma(M) = M\upharpoonright_{\Phi(\Sigma)}\)

\(\alpha = (\alpha_\Sigma : \text{sen}(\widehat{\text{FOLR}})(\Phi(\Sigma)) \leftrightarrow \text{sen}(\widehat{\text{FOL}})(\Sigma) \mid \Sigma \in |\Sigma(\widehat{\text{FOL}})|)\)

we also have a morphism \(\widehat{\text{FOL}} \rightarrow \widehat{\text{HL}}\)
**Relating Institutions**

**comorphism:** capture the way in which a “simpler” institution is embedded (encoded) into a “richer” one.  

\[ \Phi, \beta, \alpha : \widehat{DL} \rightarrow \widehat{FOL} \]  

\[ \Phi : \text{Sign}(\widehat{DL}) \rightarrow \text{Sign}(\widehat{FOL}) \] encodes a DL signature into a FOL signature

\[ \beta = (\beta_\Sigma : \text{Mod}(\widehat{FOL})(\Phi(\Sigma)) \rightarrow \text{Mod}(\widehat{DL})(\Sigma) \mid \Sigma \in \Sigma(\widehat{DL})) \]

\[ \alpha = (\alpha_\Sigma : \text{sen}(\widehat{DL})(\Sigma) \rightarrow \text{sen}(\widehat{FOL})(\Phi(\Sigma)) \mid \Sigma \in \Sigma(\widehat{FOL})) \]

- both are the subject of a corresponding satisfaction condition

- there exist a variety of definitions for morphisms and a variety of definitions for comorphisms in literature

---

^\[a^] is the institution of description logic
Outline

- institution independent logic programming paradigms
  - institutions
  - institution independent first order logic
  - institution independent logic programming paradigms
    - open world assumption (OWA)
    - closed world assumption (CWA)

- logic programming viewpoints on web ontologies
  - institution of description logic
  - description logic and logic programming paradigms

- an institutional approach of SW stack
  - institutional meaning of RDF layer
  - institutional meaning of ontology layer
  - institutional meaning of layering

- conclusion
A sentence $\neg \phi$ is the **negation** of the $\Sigma$-sentence $\phi$ iff
\[(\forall M) M \models_{\Sigma} \neg \phi \iff M \not\models_{\Sigma} \phi\]

A sentence $\phi_1 \land \phi_2$ is the **conjunction** of the $\Sigma$-sentences $\phi_1$ and $\phi_2$ iff
\[(\forall M) M \models_{\Sigma} \phi_1 \land \phi_2 \iff (M \models_{\Sigma} \phi_1 \land M \models_{\Sigma} \phi_2)\]

$\Sigma$ has **negation/conjunction** iff there is $\phi' \in \text{sen}(\Sigma)$ semantically equivalent to $\neg \phi/(\phi_1 \land \phi_2)$ for each $\phi \in \text{sen}(\Sigma)/\phi_1, \phi_2 \in \text{sen}(\Sigma)$, respectively.

The other logical connectives like disjunction, implication, equivalence are defined as usually.
Institution independent quantifiers

A. Tarlecki 1986, R. Diaconescu 2004

variables as signature morphisms

\((\forall x_1, x_2) p(x_1, x_2)\)

\(X = \{x_1, x_2\}, \Sigma(X) = \Sigma \cup X\), where the variables \(X\) are seen as constants

\(X\) can be seen as a signature morphism \(X : \Sigma \rightarrow \Sigma(X)\) and \(p(x_1, x_2)\) as a \(\Sigma(X)\)-sentence.
variables as signature morphisms

\[(\forall x_1, x_2)p(x_1, x_2)\]

\[X = \{x_1, x_2\}, \Sigma(X) = \Sigma \cup X, \text{ where the variables } X \text{ are seen as constants}\]

\[X \text{ can be seen as a signature morphism } X : \Sigma \rightarrow \Sigma(X) \text{ and } p(x_1, x_2) \text{ as a } \Sigma(X) \text{-sentence.}\]

An \(X\)-expansion of a \(\Sigma\)-model \(M\) is a \(\Sigma(X)\)-model \(M'\) s. t. \(M' \upharpoonright_X = M\).

\(M\) satisfies \((\forall x_1, x_2)p(x_1, x_2)\) iff any of its \(X\)-expansions \(M'\) satisfies \(p(x_1, x_2)\).
Institution independent quantifiers

The abstract notion of FOL variable is captured by a representable signature morphism $\chi : \Sigma \rightarrow \Sigma'$.
The abstract notion of FOL variable is captured by a representable signature morphism \( \chi : \Sigma \to \Sigma' \)

A \( \Sigma \)-sentence \((\forall X)\varphi'\) is the universal quantification of the \( \Sigma' \)-sentence \( \varphi' \) iff

\[
M \models_\Sigma (\forall X)\varphi' \iff (\forall M' \text{ a } \Sigma'\text{-model}) M' \upharpoonright X = M \Rightarrow M' \models_{\Sigma'} \varphi'
\]
Institution independent atomic formulas

A set $F$ of $\Sigma$-sentences is **basic** iff there is a $\Sigma$-model $M_F$ s.t. $M \models \Sigma F$ iff there is a homomorphism $M_F \rightarrow M$.

e.g., $(\exists x)\rho(x)$ is basic ($\rho$ a unary predicate symbol)

If $M_F \rightarrow M$ is unique, then $F$ is **epic basic**.

e.g., $\rho(a)$ is epic basic ($a$ a constant)
A set $F$ of $\Sigma$-sentences is **basic** iff there is a $\Sigma$-model $M_F$ s.t. $M \models \Sigma F$ iff there is a homomorphism $M_F \rightarrow M$.

e.g., $(\exists x)p(x)$ is basic ($p$ a unary predicate symbol)

If $M_F \rightarrow M$ is unique, then $F$ is **epic basic**.

e.g., $p(a)$ is epic basic ($a$ a constant)

A **Horn clause**: $(\forall X)F \rightarrow F'$ s.t. $F$ is epic basic, $F'$ is basic, and $X : \Sigma \rightarrow \Sigma'$ is representable (Diaconescu, 2004)

**Horn specification**: $(\Sigma, F)$ with $F$ a set of Horn clauses
Adding FOL (HL) structure to arbitrary inst.

if $\mathcal{S}$ has no FOL structure, then we can define $\text{FOL}(\mathcal{S})$ ($\text{HL}(\mathcal{S})$):

- identify (epic) basic sentences
- identify representable signature morphisms
- add FOL (HL) formulas to basic sentences

$\text{FOL}(\mathcal{S}) = \mathcal{S} \sqcup \text{FOL}(\mathcal{S})$ ($\text{HL}(\mathcal{S})$ is similarly defined)

in fact $\mathcal{S} \leftarrow \overline{\text{FOL}(\mathcal{S})} \rightarrow \text{FOL}(\mathcal{S})$ is the pullback of $\mathcal{S} \rightarrow \text{BS}(\mathcal{S}) \leftarrow \text{FOL}(\mathcal{S})$ in the category $\text{Ins}$ of institutions with morphisms as arrows, where $\text{BS}(\mathcal{S})$ is the subinstitution corresponding to basic sentences.
Outline

- institution independent logic programming paradigms
  - institutions
  - institution independent first order logic
  - institution independent logic programming paradigms
    - open world assumption (OWA)
    - closed world assumption (CWA)
- logic programming viewpoints on web ontologies
  - institution of description logic
  - description logic and logic programming paradigms
- an institutional approach of SW stack
  - institutional meaning of RDF layer
  - institutional meaning of ontology layer
  - institutional meaning of layering
- conclusion
Logic programs are Horn specifications

The following example is from Nilson and Maluszynski's book:

\[ \begin{align*}
\text{proud}(X) & \leftarrow \text{parent}(X, Y), \text{newBorn}(Y) \\
\text{parent}(X, Y) & \leftarrow \text{father}(X, Y) \\
\text{parent}(X, Y) & \leftarrow \text{mother}(X, Y) \\
\text{father}(Adam, Mary) \\
\text{newBorn}(Mary)
\end{align*} \]

\[ \Sigma(P) = (\{Adam, Mary\}, \{\text{proud}, \text{parent}, \text{newborn}, \text{father}, \text{mother}\}) \]

\[ F(P) \text{ is the set of the above Horn clauses} \]

\[ P = (\Sigma(P), F(P)) \]
Definite Logic Programming institution with base institution $\mathcal{S}$, $\mathcal{LP}(\mathcal{S})$, is defined as follows:

1. identify the (epic) basic sentences;
2. construct Horn clauses;
3. the signatures are Horn specifications $\text{HSpec}$;
4. the model functor is $\text{Mod}(\mathcal{S})$ extended to Horn specifications;
5. the $\Sigma$-sentences are Horn queries $(\exists X)\varphi$ with $\varphi$ basic;
6. the satisfaction is given by $M \models_{(\Sigma,F)} (\exists X)\varphi$ iff $M \models_{\Sigma} (\exists X)\varphi$. 


Entailment systems

An entailment system (Meseguer, 1989) consists of:
- a category of signatures $\text{Sign}$
- a sentence functor $\text{sen} : \text{Sign} \to \text{Set}$ associating to each signature a set of sentences
- a function $\vdash$ associating to each signature $\Sigma$ an entailment relation $\vdash^\Sigma \subseteq \mathcal{P}(\text{sen}(\Sigma)) \times \text{sen}(\Sigma)$ s.t.:
  - reflexivity: $\{\varphi\} \vdash^\Sigma \varphi$
  - monotonicity: if $F \vdash^\Sigma \varphi$ and $F \subset F'$, then $F' \vdash^\Sigma \varphi$
  - tranzititivity: if $F \vdash^\Sigma \varphi_i$ for $i \in I$, and $F \cup \{\varphi_i \mid i \in I\} \vdash^\Sigma \varphi$, then $F \vdash^\Sigma \varphi$
  - $\vdash$-translation: if $F \vdash^\Sigma \varphi$, then for each $\phi : \Sigma \to \Sigma'$ in $\text{Sign}$, $\text{sen}(\phi)(F) \vdash^\Sigma \text{sen}(\phi)(\varphi)$

A non-monotonic entailment system satisfies only reflexivity, tranzititivity, and $\vdash$-translation
Sound and complete entailment systems

- $\mathcal{S} = (\text{Sign}, \text{sen}, \text{Mod}, \models)$ an institution
- $\mathcal{E} = (\text{Sign}, \text{sen}, \vdash)$ an entailment system for $\mathcal{S}$
- $\mathcal{E}$ is **sound** iff $F \vdash_{\Sigma} \varphi$ implies $(\Sigma, F) \models_{\Sigma} \varphi$
- $\mathcal{E}$ is **complete** iff $(\Sigma, F) \models_{\Sigma} \varphi$ implies $F \vdash_{\Sigma} \varphi$
- consider definite logic programs over FOL, $\mathcal{LP}(\widehat{\text{FOL}})$
- SLD-resolution is a sound and complete entailment system for $\mathcal{LP}(\widehat{\text{FOL}})$
Infering entailment systems for definite LP

**Herbrand Theorem** (Diaconescu, 2004). In an arbitrary institution consider a specification \((\Sigma, F)\) which has an initial model \(0_{\Sigma,F}\). Then for each query \((\exists X)\varphi\)

\[(\Sigma, F) \models (\exists X)\varphi \text{ iff } 0_{\Sigma,F} \models (\exists X)\varphi\]

- each Herbrand spec \((\Sigma, F)\) has an initial model \(0_{\Sigma,F}\) (Makowsky, 1987)
- For Horn specifications over FOL, \(0_{\Sigma,F}\) is the least Herbrand model
- define \((\Sigma, F) \vdash \varphi\) iff \(0_{\Sigma,F} \models \varphi\)
- \(\vdash\) is sound and complete
Soundness and Completeness reformulated

- $\text{Mod}^\vdash (\Sigma, F) = \text{the set of } \Sigma\text{-models that satisfy all sentences } \varphi \text{ s.t. } (\Sigma, F) \vdash \varphi$

- $\mathcal{E}$ is **sound** iff
  $\text{Mod}^\vdash (\Sigma, F) \subseteq \text{Mod}(\Sigma, F)$

- $\mathcal{E}$ is **complete** iff
  $\text{Mod}(\Sigma, F) \subseteq \text{Mod}^\vdash (\Sigma, F)$

- for definite logic programming we have
  $\text{Mod}(\Sigma, F) = \text{Mod}^\vdash (\Sigma, F)$
Logic Programming with negation

\textbf{CWA}: If we cannot prove \((\Sigma, F) \models \varphi\), then we add \(\neg \varphi\) to \(F\) or, equivalently, restrict \(\text{Mod}^\models (\Sigma, F)\) to those models satisfying \(\neg \varphi\).

Generally, the problem of showing \((\Sigma, F) \nvdash \varphi\) is not decidable and therefore some practical solutions were proposed.

**general Horn clauses:**
\[(\forall X) \varphi_1 \land \cdots \land \varphi_m \land \neg \varphi_{m+1} \land \cdots \land \neg \varphi_n \rightarrow \varphi_0\]
such that \(\varphi_i\) is epic basic, for each \(i = 0, \ldots, n\), and \(X : \Sigma \rightarrow \Sigma'\) is representable.

\(\text{GHSpec} = \text{general Horn specifications (general logic prgms)}\)
Some solutions

- **negation as failure**
  \[ \text{(NAF)} \quad \frac{\text{if } \varphi \text{ has no a finitely failed SLD-tree}}{(\Sigma, F) \vdash \neg \varphi} \]

- **stable semantics** (Gelfond and Lifschitz, 1988)
  if \( M \) is an Herbrand model, then \((\Sigma, F)_M\) is the program obtained from \((\Sigma, F)\) by deleting
  - any general clause having a \( \neg \varphi \) in its body with \( \varphi \in M \), and
  - all negative literals in the remaining clause bodies

\((\Sigma, F)_M\) is a definite program, so it has the least Herbrand model \( M_{\Sigma,F} \)

\( M \) is **stable** if \( M = M_{\Sigma,F} \)

\[
\begin{align*}
tired & \leftarrow \neg \text{works} \\
\text{works} & \leftarrow \neg \text{tired}
\end{align*}
\]

we may have several stable models

\( M_1 = \{ \text{tired} \} \) and \( M_2 = \{ \text{works} \} \)

**cautious entailment**: \((\Sigma, F) \vdash \varphi \) iff \( M \models \varphi \) for all stable models
Some solutions

- **well-founded semantics** (Gelder et al., 1991)

  \[
  \begin{align*}
  \text{odd}(s(s(x))) & \leftarrow \text{odd}(x) \\
  \text{even}(x) & \leftarrow \neg\text{odd}(x) \\
  \text{odd}(s(0)) &
  \end{align*}
  \]

  \[M = \emptyset\]

  \[T_{\Sigma,F}(M) = \{\text{odd}(s(0))\}\] (the greatest “computed” set)

  \[U_{\Sigma,F}(M) = \{\text{odd}(s^{2n}(0))\}\] (the greatest “unfounded” set)

  \[W_{\Sigma,F}(M) = T_{\Sigma,F}(M) \cup \neg U_{\Sigma,F}(M)\]

  the well founded model \(W = \) the least fixpoint of \(W_{\Sigma,F}\)

  **entailment**: \((\Sigma, F) \vdash F\) iff \(W \models F\)

- **stratified semantics** (Apt, Blair and Walker, 1988)

  stratified programs \((\Sigma_1, F_1) \cup \cdots \cup (\Sigma_n, F_n)\)

  if \(\neg p(\ldots)\) (or \(p(\ldots)\)) occurs in a body in strata \(P_i\), then \(p\) is the head of clause in \((\Sigma_1, F_1) \cup \cdots \cup (\Sigma_{i-1}, F_{i-1}) (\Sigma_1, F_1) \cup \cdots \cup (\Sigma_i, F_i)\)

  a stratified program has a standard model \(M\) (also minimal)

  **entailment**: \((\Sigma, F) \vdash F\) iff \(M \models F\), where \(M\) is the standard model
Problems with the monotonicity

we may have $(\Sigma, F) \vdash \varphi$, $F \subset F'$, and $(\Sigma, F') \nvdash \varphi$ (e.g., $\varphi = \neg \varphi'$ and $F' = F \cup \{\varphi'\}$)

therefore not any morphism $\phi : (\Sigma, F) \rightarrow (\Sigma', F')$ is appropriate for general logic programs

$\text{GHSpec}^\rightarrow$ is the full subcategory of $\text{GHSpec}$ corresponding to the morphisms $\phi : (\Sigma, F) \rightarrow (\Sigma', F')$ such that for each $M'$ in $\text{Mod}^\rightarrow (\Sigma', F')$, $M' \upharpoonright \phi$ is in $\text{Mod}^\rightarrow (\Sigma, F)$

in this way we restore the satisfaction condition:
$M' \upharpoonright \phi \models (\Sigma, F) \varphi \iff M' \models (\Sigma', F') \text{sen}(\phi)(\varphi)$
General LP in arbitrary institutions

\[ \mathcal{S} = (\text{Sign}, \text{Mod}, \text{sen}, \models) \]

\[ \mathcal{E} = (\text{Sign}, \text{GHClause}, \vdash) \] a (non-monotonic) entailment system

define \( \mathcal{LP}^\vdash(\mathcal{S}) \), where

- the sentences are \( \text{GHSpec}^\vdash \);
- the model functor is \( \text{Mod}^\vdash \) restricted to \( \text{GHSpec}^\vdash \);
- sentences are Horn queries;
- the satisfaction relation \( \models^\vdash(\Sigma, F) \) is \( \models(\Sigma, \text{setsen}) \) restricted to \( \text{Mod}^\vdash(\Sigma, F) \)

we have \( F \vdash^\Sigma \varphi \) iff \( (\Sigma, F) \models^\vdash \varphi \)
Datalog

This example is inspired from Foundations of Databases by Abiteboul, Hull, and Vianu:

<table>
<thead>
<tr>
<th>Links</th>
<th>StReachable(x, x)</th>
<th>StReachable(x, y) ← stReachable(x, z), Links(u, z, y)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IC820 Zürich</td>
<td>Bern</td>
<td></td>
</tr>
<tr>
<td>IC820 Bern</td>
<td>Thum</td>
<td></td>
</tr>
<tr>
<td>S5 15542 Bern</td>
<td>Gümenen</td>
<td></td>
</tr>
<tr>
<td>S5 15542 Gümenen</td>
<td>Kerzer</td>
<td></td>
</tr>
<tr>
<td>BUS 5206 Gümenen</td>
<td>Münchenwiller</td>
<td></td>
</tr>
<tr>
<td>BUS 5206 Münchenwiller</td>
<td>Murten</td>
<td></td>
</tr>
</tbody>
</table>

\[ \Sigma^{xtl} = \{ \text{Links} \}, \Sigma^{ntl} = \{ \text{StReachable} \}, \Sigma = \Sigma^{xtl} \cup \Sigma^{ntl} \]

the database is a \( \Sigma^{xtl} \)-model (knowledge base \( kb \))

\( ((\Sigma, F), kb) \)-model is a \( \Sigma \)-model \( M \) s.t. \( M|_{\Sigma^{xtl}} = kb \)
we assume that $\text{Sign}$ has pushouts

$(\Sigma, F)$ a Horn specification

$\Sigma^{xtl}$ is \textbf{extensional} for $(\Sigma, F)$ if

1. there exists $\iota : \Sigma^{xtl} \rightarrow \Sigma$

2. any basic sentence occurring only in the bodies of the clauses is the image of a $\Sigma^{xtl}(X)$-sentence, where $X : \Sigma^{xtl} \rightarrow \Sigma^{xtl}(X)$ is representable signature morphism and $\Sigma^{xtl}(X) \rightarrow \Sigma(X) \leftarrow \Sigma$ is the pushout of $(X, \iota)$

3. if $\phi : (\Sigma, F) \rightarrow (\Sigma', F')$, then $\phi(\Sigma^{xtl}) \subseteq \Sigma'^{xtl}$

4. if $\Sigma^{xtl'}$ satisfies 1-3, then there exists $\Sigma^{xtl} \rightarrow \Sigma^{xtl'}$
Definite LP with Knowledge Bases (KB)

\[ \mathfrak{S} = (\text{SignMod}, \text{sen}, \models) \]

consider KB : \text{HSpec}^{op} \to \text{Cat} that maps each Horn specification \((\Sigma, F)\) into \text{Mod}(\Sigma^{x\text{tl}}) (a (\Sigma, F)-knowledge base is a \(\Sigma^{x\text{tl}}\)-model)

define \(L^\mathcal{P}^{KB}(\mathfrak{S})\), where

- the category of signatures is Grothendieck category \(KB^\#\), i.e., a signature is a pair \(((\Sigma, F), kb)\), where \((\Sigma, F) \in \text{HSpec, and } kb\) is a \((\Sigma, F)\)-knowledge base;
- the model functor maps each signature \(((\Sigma, F), kb)\) into the subcategory of \text{Mod}(\Sigma, F) induced by the set of models \(M\) with \(M\models^{\Sigma^{x\text{tl}}} kb\);
- the sentences are Horn queries;
- the satisfaction is given by \(M \models ((\Sigma, F), kb) \varphi \iff M \models (\Sigma, F) \varphi\).
General LP with KB in arbitrary institutions

$$\mathcal{S} = (\text{Sign}, \text{Mod}, \text{sen}, \models)$$

$$\mathcal{E} = (\text{Sign}, \text{GHClause}, \vdash)$$ a (non-monotonic) entailment system

$$\text{KB} : \text{GHSpec} \vdash^{op} \rightarrow \text{Cat}$$

define $$\mathcal{LP}^{\text{KB}, \vdash}_{\mathcal{S}}$$, where

- the signature are given by $$\text{KB}^\#$$;
- the model functor maps each signature $$((\Sigma, F), kb)$$ into the subcategory of $$\text{Mod}^\vdash(\Sigma, F)$$ induced by the set of models $$M$$ with $$M \models_{\Sigma^{xt}} kb$$;
- the sentences are Horn queries;
- the satisfaction relation $$\models^\vdash((\Sigma, F), kb)$$ is the restriction of $$\models((\Sigma, F), kb)$$ to $$\text{Mod}^\vdash((\Sigma, F), kb)$$. 
Conclusion

- institutions formalizes the intuitive notion of logical system using a categorical abstract model theory
- the main FOL concepts (FOL variable, atomic formulas, propositional connectors, and FOL quantifiers) can be defined in arbitrary institutions
- we take the advantage of these constructions to formulate the main logic programming paradigms in arbitrary institutions
Questions?

Thank you!