Institution-independent logic programming paradigms

Dorel Lucanu

dlucanu@info.uaic.ro

Universitatea “Alexandru Ioan Cuza”, Iaşi

Joint work with Yuan Fang Li and Jin Song Dong from NUS

and Traian Şerbănuţă from UIUC

In memory of Joseph . . .

Motivation: the jungle of SW Languages

SWRL FOL

Datalog

SWRL

OWL Full−

OWL

F−Logic

OWL Light−

OWL DL

OWL Light

RDF Schema

RDF

SHIQ

OWL Full

SHOIN

DHL

OWL DL−

DL

SHOIQ

DLP

DAML−OIL

DAML−OIL

WRL

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 cosists 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, $\Im = (\mathsf{Sign}, \mathsf{Mod}, \mathsf{sen}, \models)$, where

- Sign is the category of signatures,
- $\mathsf{Mod} : \mathsf{Sign}^{op} \longrightarrow \mathsf{Cat}$
- $\mathsf{sen} : \mathsf{Sign} \longrightarrow \mathsf{Set}$
- $\models\ = (\models_\Sigma \mid \Sigma \in \mid \mathsf{Sign} \mid)$, $\models_\Sigma \subseteq \mathsf{Mod}(\Sigma) \times \mathsf{sen}(\Sigma)$

Institution of first order logic ($\widehat{\mathsf{FOL}}$): signatures

$\Sigma = (OP, RL)$

OP - operation symbols

RL - relation (predicate) symbols

NAT	$LIST$
$\Sigma_{NAT} = (OP_{NAT}, RL_{NAT})$	$\Sigma_{LIST} = (OP_{LIST}, RL_{LIST})$
$OP_{NAT} = \{zero, s\}$	$OP_{LIST} = \{nil, a, b, cons\}$
$RL_{NAT} = \{plus\}$	$RL_{LIST} = \{cat\}$

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

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

$\widehat{\text{FOL}}$: models

NAT	*LIST*
$M = \omega$	$M' = \{a, b\}^*$
$M_{zero} = 0$	$M'_{nil} = \varepsilon$
$M_s(n) = n + 1$	$M_{cons}(x, w) = xw$
$M_{plus}(n', n', n'') \equiv n'' = n + n'$	$M'_{cat}(w, w', w'') \equiv w'' = ww'$

$\phi := \Sigma_{NAT} \to \Sigma_{LIST}$

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

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

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

$\widehat{\mathsf{FOL}}$: sentences

F_{NAT}
$(\forall n)plus(0, n, n)$ $(\forall n, n', n'')plus(n, n', n'') \to plus(s(n), n', s(n''))$
F_{LIST}
$(\forall w)cat(nil, w, w)$ $(\forall x, w, w', w'')cat(w, w', w'') \to cat(cons(x, w), w', cons(x, w''))$

ϕ is extended to sentences

$sen(\phi) : sen(\Sigma_{NAT}) \to sen(\Sigma_{LIST})$

e.g., $sen(\phi)((\forall n)plus(0, n, n)) = (\forall w)cat(nil, w, w)$

$\widehat{\text{FOL}}$: satisfaction relation

- $M \models_{\Sigma_{NAT}} (\forall n) plus(0, n, n)$
  $M \models_{\Sigma_{NAT}} (\forall n, n', n'') plus(n, n', n'') \rightarrow plus(s(n), n', s(n''))$
  $M' \models_{\Sigma_{LIST}} (\forall w) cat(nil, w, w)$
  $M' \models_{\Sigma_{LIST}} (\forall x, w, w', w'') cat(w, w', w'') \rightarrow$
  $cat(cons(x, w), w', cons(x, w''))$
- it is the subject of the **satisfaction condition** which expresses the invariance of truth under change of notation

$$M' \models_{\Sigma_{LIST}} (\forall w) cat(nil, w, w) \text{ iff } M' \restriction_{\phi} \models_{\Sigma_{NAT}} (\forall n) plus(0, n, n)$$

Notation $\widehat{\text{FOLR}}$ is similar to $\widehat{\text{FOL}}$ excepting $OP(\Sigma)$ that includes only constants and relations for each Σ, and $\widehat{\text{HL}}$ is the institution of Horn Logic

Institutions: Specifications and Theories

- a **specification** is a pair (Σ, F), where Σ is a signature and F is a set of sentences
  e.g., $NAT = (\Sigma_{NAT}, \mathsf{F}_{NAT})$, $LIST = (\Sigma_{LIST}, \mathsf{F}_{LIST})$
- semantical consequences: $(\Sigma, \mathsf{F}) \models \varphi$ iff
  $(\forall M)(M \models_\Sigma \mathsf{F} \Rightarrow M \models_\Sigma \varphi)$
- a **theory** is a specfication (Σ, F) s.t.
  $(\forall \varphi)(\Sigma, \mathsf{F}) \models \varphi \Rightarrow \varphi \in \mathsf{F}$
- the inclusion $\mathsf{Th} \to \mathsf{Spec}$ is an equivalence of categories
- theoroidal (spec-oidal) institutions $\Im^{th}$:
  - signatures are theories (specifications) (Σ, F)
  - a (Σ, F)-sentence is a Σ-sentence
  - (Σ, F)-models are Σ-models satisfying F
  - $M \models_{(\Sigma, \mathsf{F})} \varphi$ iff $M \models_\Sigma \varphi$

Relating Institutions

morphism: capture the way in which a "richer" institution is built over a "simpler" one

e.g., $(\Phi, \beta, \alpha) : \underline{\widehat{\mathsf{FOL}}} \longrightarrow \underline{\widehat{\mathsf{FOLR}}}$

$\Phi : \mathsf{Sign}(\underline{\widehat{\mathsf{FOL}}}) \longrightarrow \mathsf{Sign}(\underline{\widehat{\mathsf{FOLR}}})$ forgets non-constant operations

$\beta = (\beta_\Sigma : \mathsf{Mod}(\underline{\widehat{\mathsf{FOL}}})(\Sigma) \longrightarrow \mathsf{Mod}(\underline{\widehat{\mathsf{FOLR}}})(\Phi(\Sigma)) \mid \Sigma \in |\Sigma(\underline{\widehat{\mathsf{FOL}}})|)$,

$\beta_\Sigma(M) = M{\restriction}_{\Phi(\Sigma)}$

$\alpha = (\alpha_\Sigma : \mathsf{sen}(\underline{\widehat{\mathsf{FOLR}}})(\Phi(\Sigma)) \hookrightarrow \mathsf{sen}(\underline{\widehat{\mathsf{FOL}}})(\Sigma) \mid \Sigma \in |\Sigma(\underline{\widehat{\mathsf{FOL}}})|)$

we also have a morphism $\underline{\widehat{\mathsf{FOL}}} \to \underline{\widehat{\mathsf{HL}}}$

Relating Institutions

comorphism: capture the way in which a "simpler" institution is embedded (encoded) into a "richer" one

e.g., $(\Phi, \beta, \alpha) : \widehat{\underline{\mathsf{DL}}}$[a] $\rightarrow \widehat{\underline{\mathsf{FOL}}}$

$\Phi : \mathsf{Sign}(\widehat{\underline{\mathsf{DL}}}) \rightarrow \mathsf{Sign}(\widehat{\underline{\mathsf{FOL}}})$ encodes a DL signature into a FOL signature

$\beta = (\beta_\Sigma : \mathsf{Mod}(\widehat{\underline{\mathsf{FOL}}})(\Phi(\Sigma)) \rightarrow \mathsf{Mod}(\widehat{\underline{\mathsf{DL}}})(\Sigma) \mid \Sigma \in |\Sigma(\widehat{\underline{\mathsf{DL}}})|)$

$\alpha = (\alpha_\Sigma : \mathsf{sen}(\widehat{\underline{\mathsf{DL}}})(\Sigma) \rightarrow \mathsf{sen}(\widehat{\underline{\mathsf{FOL}}})(\Phi(\Sigma)) \mid \Sigma \in |\Sigma(\widehat{\underline{\mathsf{FOL}}})|)$

- both are the subject of a corresponding satisfaction condition
- there exist a variety of definitions for morphisms and variety of definitions for comorphisms in literature

[a] $\widehat{\underline{\mathsf{DL}}}$ 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

Institution independent prop. connectives

A sentence $\neg\varphi$ is the negation of the Σ-sentence φ iff

$$(\forall M)M \models_\Sigma \neg\varphi \Leftrightarrow M \not\models_\Sigma \varphi$$

A sentence $\varphi_1 \wedge \varphi_2$ is the conjuction of the Σ-sentences φ_1 and φ_2 iff

$$(\forall M)M \models_\Sigma \varphi_1 \wedge \varphi_2 \Leftrightarrow (M \models_\Sigma \varphi_1 \wedge M \models_\Sigma \varphi_2)$$

$\Im$ has negation/conjuction iff there is $\varphi' \in \mathsf{sen}(\Sigma)$ semantically equivalent to $\neg\varphi$/$(\varphi_1 \wedge \varphi_2)$ for each $\varphi \in \mathsf{sen}(\Sigma)$/$\varphi_1, \varphi_2 \in \mathsf{sen}(\Sigma)$, 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 $X : \Sigma \to \Sigma(X)$ and $p(x_1, x_2)$ as a $\Sigma(X)$-sentence.

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 $X : \Sigma \to \Sigma(X)$ and $p(x_1, x_2)$ as a $\Sigma(X)$-sentence.

An X-expansion of a Σ-model M is a $\Sigma(X)$-model M' s. t. $M'\restriction_X = M$.
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 $x : \Sigma \rightarrow \Sigma'$

Institution independent quantifiers

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

A Σ-sentence $(\forall \chi)\varphi'$ is the universal quantification of the Σ'-sentence φ' iff

$$M \models_\Sigma (\forall \chi)\varphi' \text{ iff } (\forall M' \text{ a } \Sigma'\text{-model}) M'\upharpoonright_\chi = M \Rightarrow M' \models_{\Sigma'} \varphi'$$

Institution independent atomic formulas

A set F of Σ-sentences is basic iff there is a Σ-model M_F s.t. $M \models_\Sigma F$ iff there is a homomorphism $M_F \to M$.

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

If $M_F \to M$ is unique, then F is epic basic.
e.g., $p(a)$ is epic basic (a a constant)

Institution independent atomic formulas

A set F of Σ-sentences is basic iff there is a Σ-model M_F s.t. $M \models_\Sigma F$ iff there is a homomorphism $M_F \to M$.

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

If $M_F \to M$ is unique, then F is epic basic.
e.g., $p(a)$ is epic basic (a a constant)

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

Horn specification: (Σ, F) with F a set of Horn clauses

Adding FOL (HL) structure to arbitrary inst.

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

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

$\overline{\mathcal{FOL}}(\Im)$ = $\Im \sqcup \mathcal{FOL}(\Im)$ ($\overline{\mathcal{HL}}(\Im)$ is similarly defined)

in fact $\Im \leftarrow \overline{\mathcal{FOL}}(\Im) \rightarrow \mathcal{FOL}(\Im)$ is the pullback of $\Im \rightarrow \mathcal{BS}(\Im) \leftarrow \mathcal{FOL}(\Im)$ in the category Ins of institutions with morphisms as arrows, where $\mathcal{BS}(\Im)$ 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:

$$proud(X) \leftarrow parent(X, Y), newBorn(Y)$$

$$parent(X, Y) \leftarrow father(X, Y)$$

$$parent(X, Y) \leftarrow mother(X, Y)$$

$$father(Adam, Mary)$$

$$newBorn(Mary)$$

$\Sigma(P) = (\{Adam, Mary\}, \{proud, parent, newborn, father, mother\})$
$\mathsf{F}(P)$ is the set of the above Horn clauses
$P = (\Sigma(P), \mathsf{F}(P))$

Institution indep. Definite Logic Programming

Definite Logic Programming institution with base institution $\Im$, $\mathcal{LP}(\Im)$, is defined as follows:

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

Entailment systems

An **entailment system** (Meseguer, 1989) consists of:

- a category of signatures Sign
- a sentence functor $\mathsf{sen} : \mathsf{Sign} \longrightarrow \mathsf{Set}$ associating to each signature a set of sentences
- a function $\vdash$ associating to each signature Σ an **entailment relation** $\vdash_\Sigma \subseteq \mathcal{P}(\mathsf{sen}(\Sigma)) \times \mathsf{sen}(\Sigma)$ s.t.:
  - reflexivity: $\{\varphi\} \vdash_\Sigma \varphi$
  - monotonicity: if $F \vdash_\Sigma \varphi$ and $F \subset F'$, then $F' \vdash_\Sigma \varphi$
  - tranzitivity: 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 Sign, $\mathsf{sen}(\phi)(F) \vdash_{\Sigma'} \mathsf{sen}(\phi)(\varphi)$

A **non-monotonic entailment system** satisfies only reflexivity, tranzitivity, and $\vdash$-translation

Sound and complete entailment systems

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

Infering entailment systems for definite LP

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

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

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

Soundness and Completeness reformulated

- $\mathsf{Mod}^{\vdash}(\Sigma, \mathsf{F})$ = the set of Σ-models that satisfy all sentences φ s.t. $(\Sigma, \mathsf{F}) \vdash \varphi$
- $\mathcal{E}$ is **sound** iff
  $\mathsf{Mod}^{\vdash}(\Sigma, \mathsf{F}) \subseteq \mathsf{Mod}(\Sigma, \mathsf{F})$
- $\mathcal{E}$ is **complete** iff
  $\mathsf{Mod}(\Sigma, \mathsf{F}) \subseteq \mathsf{Mod}^{\vdash}(\Sigma, \mathsf{F})$
- for definite logic programming we have
  $\mathsf{Mod}(\Sigma, \mathsf{F}) = \mathsf{Mod}^{\vdash}(\Sigma, \mathsf{F})$

Logic Programming with negation

CWA: If we cannot prove $(\Sigma, \mathsf{F}) \vdash \varphi$, then we add $\neg\varphi$ to F

$$(CWA)\ \frac{(\Sigma, \mathsf{F}) \not\vdash \varphi}{(\Sigma, \mathsf{F}) \vdash \neg\varphi}$$

or, equivalently, restrict $\mathsf{Mod}^{\vdash}(\Sigma, \mathsf{F})$ to those models satisfying $\neg\varphi$

generally, the problem of showing $(\Sigma, \mathsf{F}) \not\vdash \varphi$ is not decidabile and therefore some practical solutions were proposed

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

GHSpec = general Horn specifications (general logic prgms)

Some solutions

- negation as failure

$$(NAF)\ \frac{\text{if } \varphi \text{ has no a finitely failed SLD-tree}}{(\Sigma, \mathsf{F}) \vdash \neg\varphi}$$

- stable semantics (Gelfond and Lifschitz, 1988)

  if M is an Herbrand model, then $(\Sigma, \mathsf{F})_M$ is the program obtained from (Σ, 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

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

  M is stable if $M = M_{\Sigma,\mathsf{F}}$

  $tired \leftarrow \neg works$

  $works \leftarrow \neg tired$

  we may have several stable models

  $M_1 = \{tired\}$ and $M_2 = \{works\}$

  cautious entailment: $(\Sigma, \mathsf{F}) \vdash \varphi$ iff $M \models \varphi$ for all stable models

Some solutions

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

  $odd(s(s(x))) \leftarrow odd(x)$

  $even(x) \leftarrow \neg odd(x)$

  $odd(s(0))$

  $M = \emptyset$

  $T_{\Sigma,F}(M) = \{odd(s(0))\}$ (the greatest "computed" set)

  $U_{\Sigma,F}(M) = \{odd(s^{2n}(0)\}$ (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 $(\Sigma, \mathsf{F}) \vdash \varphi$, $\mathsf{F} \subset \mathsf{F}'$, and $(\Sigma, \mathsf{F}') \not\vdash \varphi$ (e.g., $\varphi = \neg\varphi'$ and $\mathsf{F}' = \mathsf{F} \cup \{\varphi'\}$)

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

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

- in this way we restore the satisfaction condition:
$M'\restriction_{\phi} \models_{(\Sigma,\mathsf{F})} \varphi \Leftrightarrow M' \models_{(\Sigma',\mathsf{F}')} \mathsf{sen}(\phi)(\varphi)$

General LP in arbitrary institutions

$\Im = (\mathsf{Sign}, \mathsf{Mod}, \mathsf{sen}, \models)$

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

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

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

we have $\mathsf{F} \vdash_{\Sigma} \varphi$ iff $(\Sigma, \mathsf{F}) \models^{\vdash} \varphi$

Datalog

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

Links

IC820	Zürich	Bern
IC820	Bern	Thum
S5 15542	Bern	Gümenen
S5 15542	Gümenen	Kerzer
BUS 5206	Gümenen	Münchenwiller
BUS 5206	Münchenwiller	Murten

$$StReachable(x, x)$$
$$StReachable(x, y) \leftarrow stReachable(x, z), Links(u, z, y)$$

$\Sigma^{xtl} = \{Links\}$, $\Sigma^{ntl} = \{StReachable\}$, $\Sigma = \Sigma^{xtl} \cup \Sigma^{ntl}$

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

$((\Sigma, \mathsf{F}), kb)$-model is a Σ-model M s.t. $M\upharpoonright_{\Sigma^{xtl}} = kb$

Extensional sign. in arbitrary instutions

- we assume that Sign has pushouts
- (Σ, F) a Horn specification
- Σ^{xtl} is extensional for (Σ, F) if
  1. there exists $\iota : \Sigma^{xtl} \to \Sigma$
  2. any basic sentence occuring only in the bodies of the clauses is the image of a $\Sigma^{xtl}(X)$-sentence, where $X : \Sigma^{xtl} \to \Sigma^{xtl}(X)$ is representable signature morphism and $\Sigma^{xtl}(X) \to \Sigma(X) \leftarrow \Sigma$ is the pushout of (X, ι)
  3. if $\phi : (\Sigma, \mathsf{F}) \to (\Sigma', \mathsf{F}')$, then $\phi(\Sigma^{xtl}) \subseteq \Sigma'^{xtl}$
  4. if $\Sigma^{xtl'}$ satisfies 1-3, then there exists $\Sigma^{xtl} \to \Sigma^{xtl'}$

Definite LP with Knowledge Bases (KB)

$\Im = (\mathsf{SignMod}, \mathsf{sen}, \models)$
consider $\mathsf{KB} : \mathsf{HSpec}^{op} \to \mathsf{Cat}$ that maps each Horn specification (Σ, F) into $\mathsf{Mod}(\Sigma^{xtl})$ (a (Σ, F)-knowledge base is a Σ^{xtl}-model)
define $\mathcal{LP}^{KB}(\Im)$, where

- the category of signatures is Grothendieck category $\mathsf{KB}^{\#}$, i.e., a signature is a pair $((\Sigma, \mathsf{F}), kb)$, where $(\Sigma, \mathsf{F}) \in \mathsf{HSpec}$, and kb is a (Σ, F)-knowledge base;
- the model functor maps each signature $((\Sigma, \mathsf{F}), kb)$ into the subcategory of $\mathsf{Mod}(\Sigma, \mathsf{F})$ induced by the set of models M with $M\!\upharpoonright_{\Sigma^{xtl}} = kb$;
- the sentences are Horn queries;
- the satisfaction is given by $M \models_{((\Sigma,\mathsf{F}),kb)} \varphi$ iff $M \models_{(\Sigma,\mathsf{F})} \varphi$.

General LP with KB in arbitrary institutions

$\Im = (\mathsf{Sign}, \mathsf{Mod}, \mathsf{sen}, \models)$

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

$\mathsf{KB} : \mathsf{GHSpec}^{\vdash\, op} \longrightarrow \mathsf{Cat}$

define $\mathcal{LP}^{KB,\vdash}(\Im)$, where

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

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 instutions
- we take the advantage of these constructions to formulate the main logic programming paradigms in arbitrary institutions

Questions?

$\mathcal{T}$hank you!