Indexed Categories and Bottom-Up Semantics of Logic Programs

Gianluca Amato
Università di Udine

James Lipton
Wesleyan University

The World of Logic Programming

Several extensions of logic programs

- CLP
- abstract data types
- λ-Prolog

and different semantics

- correct answers
- resultants

Goal

An unique framework for all of them in order to

- compare different features
- suggest further extensions
- provide a clean variable-free semantics
- extend results from static analysis

Therefore, we need a three-side semantics

- operational
- declarative
- fixpoint

Previous Approaches

- **Rydeheard, Burstall '85**
  Categorical Unification
- **Asperti, Martini '89**
  Categorical Syntax
  Topos-theoretic Semantics
- **Asperti, Corradini, Montanari '92**
  Kinoshita, Power '96
  Indexed Categories as Models
- **Finkelstein, Freyd, Lipton '95**
  Fixpoint Semantics
  Yoneda embedding

Terms and Categories

A many-sorted first order language $T_V(\Sigma)$ is a finite product category $\mathbb{C}$ according to the correspondence

- objects as types
- arrows as terms (and substitutions)
- equalizers as m.g.u's
- pullbacks as m.g.u's of renamed apart terms

In general, we can forget syntax by using a category $\mathbb{C}$ as the domain of terms.

Logic Programming in a Topos

Categorical Syntax: atomic formulas are pairs (A, f) where A is a predicate symbol of sort σ, $f \in \mathrm{Hom}_{\mathbb{C}}(\_, \sigma)$

Interpretation in a Topos Ω: an interpretation is given by

- a finite product functor $I : \mathbb{C} \to \Omega$
- a subobject of $I(\sigma)$ for each predicate symbol $A : \sigma$

Semantics: the interpretation I is extended to

- atomic formulas: $I(A, f)$ as the pullback of $I(\alpha)$ along $I(f)$
- goals: $I((A_1, f_1)(A_2, f_2))$ is the meet of $I(A_1, f_1)$ and $I(A_2, f_2)$

Logic Programming in an Indexed Category

Categorical Syntax: atomic formulas are pairs (A, f) where A is a predicate symbol of sort σ, $f \in \mathrm{Hom}_{\mathbb{C}}(\_, \sigma)$

Interpretation in $\mathcal{P} : \mathbb{D} \to \mathrm{Cat}$:

- a finite product functor $I : \mathbb{C} \to \mathbb{D}$
- an object $I(A)$ of $\mathcal{P}(I(\sigma))$ for each predicate symbol $A : \sigma$

Semantics: the interpretation I is extended to

- atomic formulas: $I(A, f) = \mathcal{P}(f)(I(A))$
- goals: $I((A_1, f_1)(A_2, f_2))$ is the product of $I(A_1, f_1)$ and $I(A_2, f_2)$

We can use an indexed category as the language for formulas.

Categorical Syntax

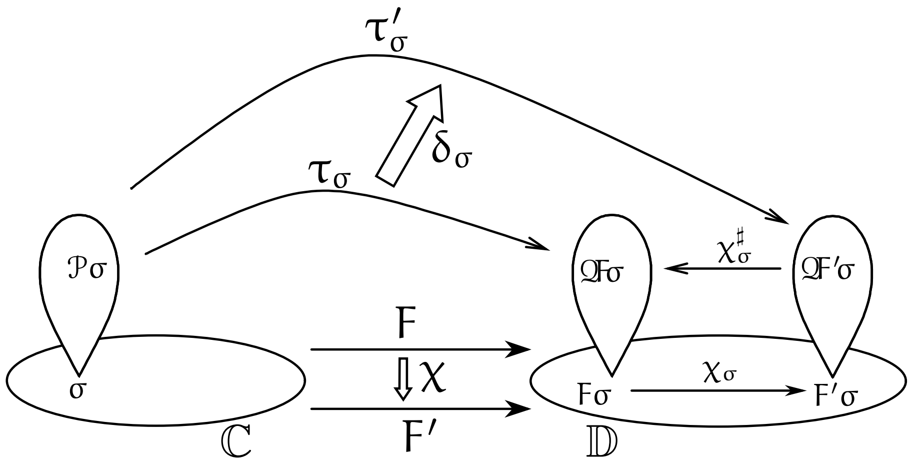
Syntax is given by an indexed category $\mathcal{P} : \mathbb{C} \to \mathsf{Cat}$ where

- $\mathbb{C}$ is the category of terms and types, as before
- objects of $\mathcal{P}(\sigma)$ are goals of type σ
- arrows in $\mathcal{P}(\sigma)$ are *constraints* between goals

Note that

- in principle, there are no concepts of *predicate symbol* or *atomic formulas*,
- given $\mathbb{C}$ and a set of predicate symbols, we can build $\mathcal{P} : \mathbb{C} \to \mathsf{Cat}$ where $\mathcal{P}(\sigma)$ is the discrete category of objects (A, t). (Power and Kinoshita)

Indexed Categories



Objects of $\mathbb{C}$	$\Longleftrightarrow$	Sorts
Arrows in $\mathbb{C}$	$\Longleftrightarrow$	Terms
Obects in $\mathcal{P}\sigma$	$\Longleftrightarrow$	Goals of sort σ
Arrows in $\mathcal{P}\sigma$	$\Longleftrightarrow$	Proofs of sort σ
Reindexing functors	$\Longleftrightarrow$	Instantiations

A Syntactic Category

Given $\mathbb{C}$ and a signature Π, we define $\mathcal{P}_{\Pi}$ as

- $\mathcal{P}_{\Pi}(\sigma)$ the discrete category with objects (A, t) with $A : \rho \in Pi$, $t :\in \mathrm{Hom}_{\mathbb{C}}(\sigma, \rho)$

- $\mathcal{P}_{\Pi}(f : \sigma \to \rho)$ maps (A, t) in $(A, t \circ f)$.

for binary logic programs.

Arrows on the Fibers

They are used to force properties of predicates at the the level of syntax.

If p and symp are goals, then

$$r_1 : p \to \mathsf{symp}$$
$$r_2 : p \to \mathsf{symp}(\langle \pi_2, \pi_1 \rangle)$$

force symp to the symmetric closure of p.

We plan to use constraint to treat

- abstract data type
- monads

Programs and Models

clause: pair of goals in $\mathcal{P}$ on the same fiber

program: set of clauses

model: is given by

- $\mathcal{Q} : \mathbb{D} \to$ Cat,
- an indexed functor $\tau : \mathcal{P} \to \mathcal{Q}$,
- an assignment ι from clauses $G_1 \leftarrow G_2 : \sigma$ to arrows in $\mathcal{Q}(F\sigma)$.

There is a *free* model.

An Example of Model

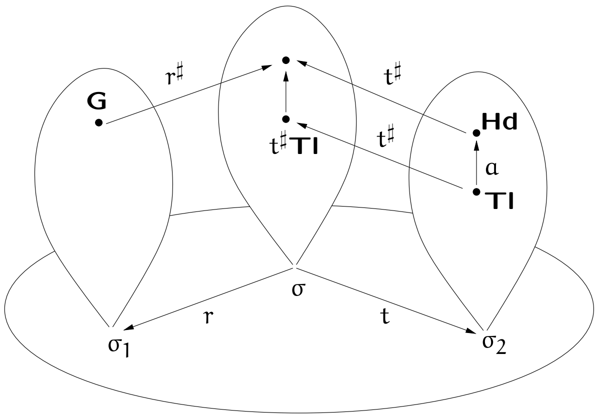
Given a category $\mathbb{C}$. we define $\mathcal{Q}$ as

- $\mathcal{Q}(\sigma) = \wp(\mathsf{Hom}_{\mathbb{C}}(1, \sigma))$
- $\mathcal{Q}(f : \sigma \to \rho)(X) = \{r \in \mathsf{Hom}_{\mathbb{C}}(1, \sigma) \mid f \circ r \in X\}$

Two (non) significant models for a program in $\mathcal{P}_{\Pi}$:

- $\tau(G : \sigma) = \emptyset$ (everything false)
- $\tau(G : \sigma) = \mathsf{Hom}(1, \sigma)$ (everything true)

Categorical Derivation



SLD step

$$\mathbf{G} \xrightarrow{\langle r,t,a\rangle} t^\sharp \mathbf{Tl}$$

Computed answer

$$ans(\mathbf{G}_1 \xrightarrow{\langle r_1,t_1,a_1\rangle} \dots \xrightarrow{\langle r_n,t_n,a_n\rangle} \mathbf{G}_{n+1}) = \\ = r_1 \circ \cdots \circ r_n$$

Correctness and Completeness

Correctness. If there is a derivation $\mathbf{G}_1 \rightsquigarrow \mathbf{G}_2$ with answer θ, then $\theta^\sharp\tau(\mathbf{G}_1) \leftarrow \tau(\mathbf{G}_2)$ is an arrow in every model.

Completeness. If $\tau(\mathbf{G}_1) \leftarrow \tau(\mathbf{G}_2)$ is an arrow, then $\mathbf{G}_1 \rightsquigarrow \mathbf{G}_2$.

Herbrand Model

A new model of P in $\mathcal{P}_{\prod}$ on $\mathcal{Q}$ is

$$\tau(\mathbf{G} : \sigma) = \{f \in \mathsf{Hom}_{\mathbb{C}}(1, \sigma) \mid f^{\sharp}(\mathbf{G}) \rightsquigarrow \top\}$$

where $\top : 1$ is a goal which represents true.

This is the standard Herbrand model.

We want a fixpoint construction!

Fixpoint Semantics

We use *semantic* indexed categories $\mathcal{Q}$ such that

- fibers have coproducts and colimits of ω-chains,
- reindexing functors have left adjoints $\exists^{\mathcal{Q}}_{\mathfrak{t}}$,
- $\exists^{\mathcal{Q}}_{\mathfrak{t}}$ preserves colimits of ω-chains on the nose

We use *goal free* syntactic indexed categories, i.e. generated by a base category $\mathbb{C}$ and a predicate signature.

The T_P operator

$\mathcal{P} : \mathbb{C} \to$ Cat a goal-free syntactic category
$\mathcal{Q} : \mathbb{C} \to$ Cat a semantic category
τ an interpretation

Define $\tau' = T_P(\tau)$ as

$$\tau'(A) = \tau(A) \vee \bigvee_{A(t) \leftarrow \mathbf{Tl} \in P} \exists_t^{\mathcal{Q}} \tau(\mathbf{Tl})$$
$$\tau'(A(t)) = t^{\sharp}(\tau'(A))$$

There an indexed natural transformation

$$\nu_A : \tau(A) \xrightarrow{inj} \tau'(A) = \tau(A) \vee \ldots$$

Fixpoint

We have the ω-chain

$$\tau \to T_P(\tau) \to T_P^2(\tau) \to \ldots$$

We can find the colimit T_P^ω of the chain.

The interpretation T_P^ω can be extended to a model of P.

A Semantic Indexed Category

We extend Ω to a semantic indexed category with

- colimits given by unions
- if $t : \rho \to \sigma$, $\exists_t(X) = \{t \circ f \in f \in \mathrm{Hom}_{\mathbb{C}}(1, \rho)\}$

We obtain the standard T_P of van Emden and Kowalski.

CLP

A *constraint system* is an indexed category $\mathcal{P} : \mathbb{C} \to \mathsf{Cat}$ such that

- each fiber is a meet semilattice,
- reindexing functors have left adjoints

We define $\mathcal{Q} : \mathbb{D} \to \mathsf{Cat}$ where

- object of $\mathbb{D}$ are pairs $\langle \sigma, c \rangle$, c constraint of sort σ.
- $f : \langle \sigma_1, c_1 \rangle \to \langle \sigma_2, c_2 \rangle$ if $c_1 \leq f^\sharp c_2$.
- objects in $\mathcal{Q}(\langle \sigma, c \rangle)$ are pairs $\langle A, t \rangle$ with $A : \rho \in \Pi$, $t : \sigma \to \rho$

Results

- we have the three semantics of logic programs

- we can treat several different languages

- we can treat several different semantics

- we can treat selection rules (with pseudo-monoidal structures)

- syntax is categorical (as long as no fix-point semantics is considered)

Future Works

- abstract data types and monads
- alternative approaches to CLP
- a more liberal fixpoint construction
- extensions to hereditary Harrop formulas