Syntactic cat for 1st order logic

Alg th. → contexts → objects of Syn(T).
Alg th. → terms → arrows of Syn(T).

1st order logic → contexts → objects
1st order logic → terms → arrows → only between objects corresp. to context.
1st order logic → formulas → subobjects (of the objects corresponding to context.
|| more objects than the ones corresp. to context.

need also arrows between the new objects assoc to formula

Syntax

$\varphi(x,y,z) \qquad\qquad \psi(u,v)$

$$(x,y,z) \xrightarrow{(s(x,y,z),\, t(x,y,z))} (u,v)$$

context $\qquad\qquad$ context

Syn(T)

$$\begin{tikzcd}
{[\varphi]} \arrow[d, "\text{mono}", "p \text{ is } \varphi"'] & & {[\psi]} \arrow[d, "p \text{ is } \psi \;\; \text{mono}"] \\
{[x,y,z]} \arrow[rr, "{[s,t]}"] & & {[u,v]}
\end{tikzcd}$$

p is $(u = s \wedge t = v)$

in orange : new arrows.

$$[\varphi] \overset{?}{\dashrightarrow} [\psi]$$

arbitrary p

Syn(T)

objects = formula in context $(\vec{x}, \varphi)$ (if $\varphi = T$ this is the object corresponding to the context)

morph. $(\vec{x}, \varphi) \xrightarrow{f} (\vec{y}, \psi)$: "functional relations" in context $(\vec{x}, \vec{y})$

f: $(\vec{x}, \vec{y}, \rho)$ such that

- $\rho \vdash_{\vec{x}, \vec{y}} \varphi \wedge \psi$
- $\varphi \vdash_{\vec{x}} \exists \vec{y} \; \rho(\vec{x}, \vec{y})$
- $\rho(\vec{x}, \vec{y}), \rho(\vec{x}, \vec{y}') \vdash_{\vec{x}, \vec{y}, \vec{y}'} \vec{y} = \vec{y}'$

__identity maps__ of $(\vec{x}, \varphi)$ is $(\vec{x}, \vec{x}', \rho)$

$\rho(x, x')$ is $x = x' \wedge \varphi(x)$

$\left(\text{or} \quad x = x' \wedge \varphi(x) \wedge \varphi(x') \right)$

__composition__

$$\begin{tikzcd}
(\vec{x}, \varphi) \arrow[r, "(\vec{x}\vec{y}, \rho)"] \arrow[rr, bend right, "(\vec{x}, \vec{z}, \mu)"'] & (\vec{y}, \psi) \arrow[r, "(\vec{y}, \vec{z}, \lambda)"] & (\vec{z}, \zeta)
\end{tikzcd}$$

$\mu(\vec{x}, \vec{z})$ is $\exists \vec{y}, \rho(\vec{x}, \vec{y}) \wedge \lambda(\vec{y}, \vec{z}) \wedge \psi(y)$

Remark 1: that the definition of $\mathrm{Syn}(T)$ uses

- the $\wedge$ connector
- the $\exists$ quantifier.

Hence it make sense only for regular theories T

Remark 2: if the theory has more constructors $(0, \vee, \Rightarrow, \neg, \forall)$ the definition of $\mathrm{Syn}(T)$ is the same, we simply get more objects and more formulas.

<u>Theorem</u>: • if T is regular (coh, Heyt, Boolean), then $\mathrm{Syn}(T)$ is a regular (coh, Heyt, or Boolean) category.
(details in Johnstone's Elephant D.1.4)

• soundness and completeness: $\mathrm{Syn}(T)$ has a universal model U of T
and for a formula φ $\quad T \vdash \varphi \iff U \vDash \varphi$

(φ derivable from axioms) $\quad$ φ is true in the universal model

• $(\mathrm{Syn}(T), U)$ represents the functors of models of T

C reg cat: $\quad \mathrm{Mod}(T, C) \underset{\text{eq-cat}}{\simeq} \mathrm{Fun}^{\text{reg}}(\mathrm{Syn}(T), C)$

- Gödel completeness result for

T finitary classical (Boolean) first order theory

$(\top, \wedge, \bot, \vee, \exists, \forall, \Rightarrow, \neg)$

$$T \vdash \varphi \iff M \models \varphi$$

φ is derivable from the axiom

for any any model in <u>Set</u>

(topological interpretation in terms of topoi by Deligne)

Beyond 1st order logic

The focus of 1st order logic is formula.

→ many operations on formulas $\top, \wedge, \bot, \vee, \exists, \forall, \Rightarrow, \neg$

few operation on contexts / sort / types : concatenation / cart. products

to go beyond : add more operation / constructor on sorts / types

- function types $X \to Y$ (λ-calculus)
- Σ, Π types (in DTT)

these theories beyond 1^{st} order logic are also associated to syntactic categories (Crole : Categories for Types)

But in general these theories do not have completeness results (with respect to models in Set)

This is why 1^{st} ord. logic is so important : because it is complete.

the Lawvere revolution

1) <u>model theory</u>

logical th = language (syntax, proof theory)

<u>model</u> : translation into another language (e.g ZF ...)

2) <u>Categorical semantics</u>

language $\xrightarrow[\text{interpretation}]{\text{models}}$ in a category

heterogeneous

3) <u>Lawvere</u> : (syntactic) category $\xrightarrow[\text{functor}]{\text{models}}$ category

We've seen how to construct a category from a logical theory.

There is a process the opposite way:

$$\begin{tikzcd}[column sep=huge] \text{(regular) logical theories} \arrow[r, bend left, "\text{syntactic category}"] & \text{(regular) categories} \arrow[l, bend left, "\text{internal language of a category.}"] \end{tikzcd}$$

$$\begin{tikzcd}[column sep=huge] \text{cat of reg theories} \arrow[r, shift left, "\mathrm{Syn}(-)"] \arrow[r, phantom, "\text{adjoints}"] & \text{cat of reg cat} \arrow[l, shift left, "\text{Int-lang}"] \end{tikzcd}$$

<u>Example</u> $\mathcal{C}$ is a regular category.

can construct a regular theory.

Σ : $\underset{\frown}{S}$ = all objects of $\mathcal{C}$.

F = all arrows of $\mathcal{C}$.

R = all monomorphisms.

$\rightarrow$ context / terms / formulas for regular logic.

the signature has a canonical / tautological model in $\mathcal{C}$

A = axioms : all the sequent of Σ that are true in this model

(Σ, A) is the internal language of $\mathcal{C}$.