Category Theory and Lambda-calculi

Inspiré des notes de cours du MPRI de Paul-André Melliès, CNRS/ENS Ulm

Categories

Definition. A **category** $\mathcal{C}$ comprises

1. a collection of objects
2. a set $\mathsf{Hom}(A,B)$ of arrows (morphisms) $f : A \to B$ for every pair of objects (A,B)
3. a associative composition law $\circ : \mathsf{Hom}(B,C) \times \mathsf{Hom}(A,B) \to \mathsf{Hom}(A,C)$
4. a neutral element identity morphism $\mathsf{id}_A : A \to A$ for every object A

Example. Category Set

- objects : sets
- arrows : total functions between sets
- composition : set-theoretic function composition
- identities : identity functions

Definition. Partial ordering $\leq_P$ on set P is a reflexive, transitive, antisymmetric relation

Definition. Function $f : (P, \leq_P) \to (Q, \leq_Q)$ is **order-preserving** (or **monotone**) if $p \leq_P p'$ then $f(p) \leq_Q f(p')$

Example. Category Poset

- objects : partially-ordered sets
- arrows : order-preserving total functions
- composition : set-theoretic function composition
- identities : identity functions

Remark. Generalization of universal algebra which studies the common properties of algebraic structures.

Category	Objects	Arrows
Set	sets	total functions
Pfn	sets	partial functions
FinSet	finite sets	finite total functions
Mon	monoids	monoid morphisms
Poset	posets	monotone functions
Grp	groups	group morphisms
Ω-Alg	algebras with sig Ω	Ω-morphisms
CPO	complete partial orders	continuous functions
Vect	vector spaces	linear tranforms
Met	metric spaces	contraction maps
Top	topological spaces	continuous functions

Example. Finite categories

- category 0 : no objects, no arrows
- category 1 : one object, one arrow
- category 2 : two objects, two identity arrows, arrow from one object to another
- category 3 : objects A, B, C, three identity arrows, three arrows $f : A \to B$, $g : B \to C$, $h : A \to C$

Remark. Categorical logic : objects as arbitrary category formulas, arrows $f : A \to B$ as proofs of $A \Rightarrow B$. Identity id_A is an instance of reflexivity and following inference rule asserts transitivity of $\Rightarrow$

$$\frac{f : A \to B \qquad g : B \to C}{g \circ f : A \to C}$$

Definition. $\mathcal{C}^{\mathrm{op}}$ is **dual category** of $\mathcal{C}$ if it has same objects and opposite arrows of $\mathcal{C}$

Definition. $\mathcal{C} \times \mathcal{D}$ is **product category** of $\mathcal{C}$ and $\mathcal{D}$ if it has objects pairs (A,B), arrow pairs (f,g). Pairwise defined composition and identity arrows.

Diagrams

Definition. Diagram in $\mathcal{C}$ is a collection of vertices and directed edges, consistently labeled with objects and arrows of $\mathcal{C}$.

Definition. Diagram in $\mathcal{C}$ **commutes** if, for every pair of vertices X and Y, all the paths in the diagram from X to Y are equal

Functors

Definition. A **functor** $F : \mathcal{C} \to \mathcal{D}$ comprises

1. for every A of $\mathcal{C}$, an object FA of $\mathcal{D}$
2. for every pair (A,B) of objects of $\mathcal{C}$, a function $F_{A,B} : \mathsf{Hom}_{\mathcal{C}}(A,B) \to \mathsf{Hom}_{\mathcal{D}}(FA, FB)$

preserving

1. composition : $FA \xrightarrow{Ff} FB \xrightarrow{Fg} FC \quad = \quad FA \xrightarrow{F(g \circ f)} FC$
2. identities : $FA \xrightarrow{F\mathsf{id}_A} FA \quad = \quad FA \xrightarrow{\mathsf{id}_{FA}} FA$

Natural transformations

Definition. A **natural transformation** $\theta : F \Rightarrow G$ from functors F to G (both from $\mathcal{C}$ to $\mathcal{D}$) is a family of morphisms $\left(\theta_A \ : \ FA \to GA\right)_{A \in \mathsf{Obj}(\mathcal{C})}$ such that for all $\mathcal{C}$-arrow $f : A \to B$, commutes in $\mathcal{D}$

$$\begin{tikzcd}
FA \arrow[r, "\theta_A"] \arrow[d, "Ff"'] & GA \arrow[d, "Gf"] \\
FB \arrow[r, "\theta_B"'] & GB
\end{tikzcd}$$

If every η_A of η is an isomorphism in $\mathcal{D}$ then η is a **natural isomorphism**.

Example. $\mathsf{rev}_S : \mathsf{List}\ S \to \mathsf{List}\ S$ is a natural transformation

- if $f : S \to T$ then $\mathsf{rev}_T \circ \mathsf{maplist}\ f = \mathsf{maplist}\ f \circ \mathsf{rev}_S$

Adjunctions

Definition. An **adjunction** (L, R, ϕ) comprises

1. functors $L : \mathcal{C} \to \mathcal{D}$, $R : \mathcal{D} \to \mathcal{C}$
2. a family ϕ of natural bijections in $\mathcal{C}$ and $\mathcal{D}$, for every $\mathcal{C}$-object C and $\mathcal{D}$-object D, $\quad \mathsf{Hom}_{\mathcal{D}}(LC, D) \cong \mathsf{Hom}_{\mathcal{C}}(C, RD)$ written $\dfrac{LC \to_{\mathcal{D}} D}{C \to_{\mathcal{C}} RD} \phi_{A,B}$

Notation. $L \dashv R$: L is **left adjoint to** R

Monads

Definition. A **monad** (T, μ, η) comprises

1. an endofunctor $T : \mathcal{C} \to \mathcal{C}$
2. natural transformations $\mu : T \circ T \Rightarrow T$ and $\eta : \mathsf{id}_{\mathcal{C}} \Rightarrow T$ such that commutes

$$\begin{tikzcd}
T \circ T \circ T \arrow[r, Rightarrow, "T\mu"] \arrow[d, Rightarrow, "\mu_T"'] & T \circ T \arrow[d, Rightarrow, "\mu"] \\
T \circ T \arrow[r, Rightarrow, "\mu"'] & T
\end{tikzcd}
\qquad
\begin{tikzcd}
T \arrow[r, Rightarrow, "\eta_T"] \arrow[dr, Rightarrow, "\mathsf{id}_T"'] & T \circ T \arrow[d, Rightarrow, "\mu"] & T \arrow[l, Rightarrow, "T\eta"'] \arrow[dl, Rightarrow, "\mathsf{id}_T"] \\
 & T &
\end{tikzcd}$$

Definition. Every set S induces the **state monad** $X \ \mapsto \ S \Rightarrow (S \times X) \ : \ \mathsf{Set} \to \mathsf{Set}$ by the adjunction

$$\begin{tikzcd}
\mathsf{Set} \arrow[r, bend left, "L"] \arrow[r, phantom, "\perp"] & \mathsf{Set} \arrow[l, bend left, "R"]
\end{tikzcd}
\quad \text{where} \quad
\begin{array}{lcl} L & : & X \mapsto S \times X \\ R & : & X \mapsto S \Rightarrow X \end{array}$$

Definition. Kleisli category $\mathcal{C}_T$ of monad (T, μ, η) has

1. same objects as $\mathcal{C}$
2. morphisms $A \twoheadrightarrow B$ as morphisms $A \to TB$ in $\mathcal{C}$
3. identities $\mathsf{id}_A : A \twoheadrightarrow A$ given by $\eta_A : A \to TA$
4. compositions $g \circ_{\mathsf{K}} f$ of $f : A \twoheadrightarrow B$, $g : B \twoheadrightarrow C$ as

$$\begin{tikzcd}
A \arrow[r, "f"] & TB \arrow[r, "Tg"] & TTC \arrow[d, "\mu_C"] \\
 & B \arrow[r, "g"] & TC
\end{tikzcd}$$

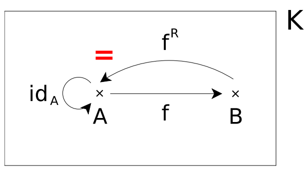
Example. Monads in OCaml

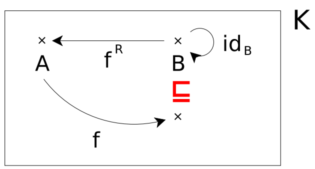
```
Monads can be viewed as a standard programming interface
to various data or control structures, which is captured
by the 'a t type. All common monads are members of it:

(>>=)  : 'a t -> ('a -> 'b t) -> 'b t      (* bind/unit *)
return : 'a   -> 'a t
run    : 'a t -> 'a

All instances of monad should obey the following equations:

return a >>= f           = f a              (* left unit *)
a >>= fun x -> return x = a                 (* right unit *)
(a >>= fun x -> b) >>= fun y -> c
      = a >>= fun x -> b >>= fun y -> c  (* associativity *)
```

Domain theory

Definition. Pair of maps $X \underset{f^R}{\overset{f}{\rightleftarrows}} Y$ is an **embedding** of X into Y if $f^R \circ f = \mathsf{id}_X$ and $f \circ f^R \sqsubseteq \mathsf{id}_Y$.

Remark. $f \sqsubseteq g$: f *approximates* g in some ordering representing their information content





LaTeX cheat sheet template made by Winston Chang
http://www.stdout.org/∼winston/latex/