Cartesian closed categories

Andreas Abel, Fabian Ruch

November 2020, September 2021

Categories generalize the notion of functions between sets to *morphisms between objects*. This gives us excellent flexiblity for the interpretation of (functional) programming languages. Functions need not be identified with their set-theoretic interpretation (as infinite tables mapping inputs to outputs), but we can choose their representation, as long as we explain *how* they map inputs to outputs, or rather, *how they compose*.

For instance, we could represent functions by snippets of machine code and applications to arguments by calls to that machine code with arguments passed suitably (e.g. on the stack). Function composition could be a new machine code snippet that invokes the first function and passes its result as argument to the second.

Typically for categorical semantics of programming languages, we will however leave the representation of functions abstract and only work with the laws of function application/composition.

This note introduces categories and structure on and within categories needed to interpret simply-typed lambda-calculus.

Contents

1 Categories

The concept of a *category* is very versatile, generalizing sets, monoids, partial orders, and graphs. It is both used to define concrete indexed structures with a composition

operation, like the partial monoid of well-formed instructions of a stack machine, as well as organizing structures into a common picture, e.g., the category of Abelian groups.

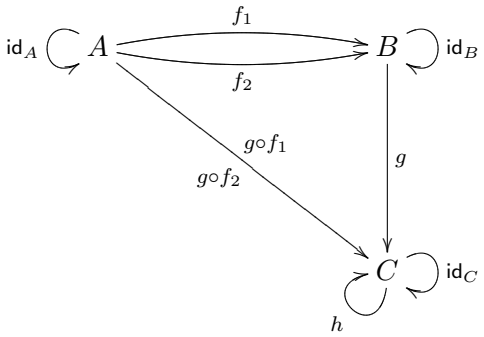


Figure 1: A category (quite small).

1.1 Definition and examples

A category is a *reflexive-transitive directed multigraph*:

Definition 1 (Category (Aczel 1995, Appendix 1.3))**.** A category $\mathcal{C}$ is given by the following data:

1. Types:
   a) A type Ob of *objects*.
   b) For each pair of objects $A, B : \mathsf{Ob}$, a type $\mathsf{Hom}(A, B)$ of (homo)morphisms $f : A \longrightarrow B$.
   c) For each pair of objects $A, B : \mathsf{Ob}$, an equivalence relation $\mathsf{Eq}(A, B)$ on $\mathsf{Hom}(A, B)$. Given $f, g : \mathsf{Hom}(A, B)$, we write $f = g$ for $\mathsf{Eq}(A, B)(f, g)$.
2. Operations:
   a) For each object $A : \mathsf{Ob}$ an automorphism $\mathsf{id}_A : A \longrightarrow A$ (identity).
   b) For each pair $f : A \longrightarrow B$ and $g : B \longrightarrow C$ of morphisms a morphism $g \circ f : A \longrightarrow C$ (composition).
3. Laws:
   a) For each morphism $f : A \longrightarrow B$ we have $\mathsf{id}_B \circ f = f$ (left identity) and $f \circ \mathsf{id}_A = f$ (right identity).
   b) For all morphisms $f : A \longrightarrow B$ and $g : B \longrightarrow C$ and $h : B \longrightarrow C$ we have $(h \circ g) \circ f = h \circ (g \circ f)$ (associativity).
   c) For all morphisms f, $f' : A \longrightarrow B$ such that $f = f'$ and g, $g' : B \longrightarrow C$ such that $g = g'$ we have $g \circ f = g' \circ f'$ (congruence).

The arrow $A \longrightarrow B$ is just a nice notation for $\mathsf{Hom}(A, B)$. It is also common to write $\mathcal{C}(A, B)$ to clarify that we mean the type $\mathsf{Hom}_{\mathcal{C}}(A, B)$ of morphisms of category $\mathcal{C}$. Also $A : \mathcal{C}$ is short for $A : \mathsf{Ob}_{\mathcal{C}}$.

Remark 1 (Homsetoid)**.** Since a type with an equivalence relation is called a *setoid*, which come with a notion of map and have products, we could just ask for a family $\mathsf{Hom} : \mathsf{Ob} \to \mathsf{Ob} \to \mathsf{Setoid}$ and setoid maps $\_ \circ \_ : \mathsf{Hom}(B, C) \times \mathsf{Hom}(A, B) \to \mathsf{Hom}(A, C)$.

The prime example for categories are collections of algebraic structures and their structure-preserving homomorphisms.

Example 1 (Groups)**.** Grp is the category of groups and group homomorphisms. More precisely, the objects of Grp are groups, and an element $f : \mathsf{Grp}(A, B)$ is a function $f : A \to B$ mapping the unit of A to the unit of B and the A-composition of two elements of A to the B-composition of their images under f.

Less abstractly, a group morphism $f : (A, 0, +, -) \longrightarrow (B, 1, \times, ^{-1})$ has to satisfy $f(0) = 1$ and $f(a + a') = f(b) \times f(b')$.

Exercise 1 (Groups)**.**

1. Give an example for a group morphism f.
2. Show that a group morphism automatically preserves inverses, i.e., $f(-a) = (f(a))^{-1}$.

Analogously to groups, other algebraic structures can be organized as categories as well (monoids, rings, fields). We exhibit the most basic examples:

Example 2 (Sets)**.** Set is the category of types A and functions $f : A \to B$.

Example 3 (Setoids)**.** Setoid is the category of setoids $(A, \approx_A)$ and $\approx$-preserving functions, i.e., $f : A \longrightarrow B$ must satisfy $f(a) \approx_B f(a')$ whenever $a \approx_A a'$.

Besides organizing algebraic structures, categories can also *implement* structures.

Example 4 (Monoid)**.** Each monoid $(M, e, \cdot)$ can be presented as category $\mathcal{C}_M$ with a single object 1 and $\mathsf{Hom}(1, 1) = M$. Then $\mathsf{id}_1 = e$ and $f \circ g = f \cdot g$.

See Figure 2 for a categorical presentation of the monoid $(\{\mathsf{true}, \mathsf{false}\}, \mathsf{true}, \_\wedge\_)$.

Figure 2: The conjunctive Boolean monoid.

Exercise 2 (Partial monoid)**.** Can any partial semigroup with identity be represented as category as well? If yes, how? If no, give a counterexample! What about partial monoids?

We call a set M with a distinguished element $e : M$ and a partial binary operation $\_ \circ \_ : M \times M \rightharpoonup M$ a *partial semigroup with identity* if 1. $(x \circ y) \circ z = x \circ (y \circ z)$ if $x \circ y$ and $(x \circ y) \circ z$, or $y \circ z$ and $x \circ (y \circ z)$ are defined, and 2. $e \circ x = x = x \circ e$ for each $x : M$.

Let us call an element o such that $z = x$ whenever $z = o \circ x$ or $z = x \circ o$ a *partial identity*. With this in mind, we call a set N with two unary operations $l, r : N \to N$ and a partial binary operation $\_ \circ \_ : N \times N \rightharpoonup N$ a *partial monoid* if 1. $(x \circ y) \circ z = x \circ (y \circ z)$ if $x \circ y$ and $(x \circ y) \circ z$, $y \circ z$ and $x \circ (y \circ z)$, or $x \circ y$ and $y \circ z$ are defined, and 2. $l(x)$ and $r(x)$ are partial identities such that $l(x) \circ x$ and $x \circ r(x)$ are defined. See Mac Lane (1998, p. 9).

Example 5 (Preorder)**.** Any preorder $(A, \leq)$ can be presented as a thin category with $\mathsf{Ob} = A$ and $\mathsf{Hom}(a, b) = \{0 \mid a \leq b\}$. Identity is reflexivity and composition is transitivity.

A category $\mathcal{C}$ is called *thin* if each homset $\mathsf{Hom}_{\mathcal{C}}(A, B)$ has *at most one* inhabitant, that is for all pairs of parallel morphisms f, $f' : A \longrightarrow B$ in $\mathcal{C}$ we have $f = f'$.

Example 6 (Relations)**.** The category Rel has types as objects and binary relations as morphisms: $\mathsf{Rel}(A, B) = \mathcal{P}(A \times B)$.

Example 7 (Contexts and substitutions)**.** Take the typing contexts Γ of simply-typed lambda-calculus as objects, $\mathsf{Ob} = \mathsf{Cxt}$, and the set of substitutions $\mathsf{Sub}\,\Gamma\,\Delta$ as morphisms from Γ to Δ.

Definition 2 (Subcategory)**.** A category $\mathcal{D}$ is a *subcategory* of $\mathcal{C}$ if $\mathsf{Ob}_{\mathcal{D}} \subseteq \mathsf{Ob}_{\mathcal{C}}$, $\mathsf{Hom}_{\mathcal{D}}(A, B) \subseteq \mathsf{Hom}_{\mathcal{C}}(A, B)$ for all $A, B : \mathsf{Ob}_{\mathcal{D}}$, $\mathsf{id}_{\mathcal{D},A} = \mathsf{id}_{\mathcal{C},A}$ for all $A : \mathsf{Ob}_{\mathcal{D}}$, and $g \circ_{\mathcal{D}} f = g \circ_{\mathcal{C}} f$ for all $f : \mathsf{Hom}_{\mathcal{D}}(A, B)$ and $g : \mathsf{Hom}_{\mathcal{D}}(B, C)$.

If $\mathsf{Ob}_{\mathcal{D}} = \mathsf{Ob}_{\mathcal{C}}$, the subcategory is *wide*.

If $\mathsf{Hom}_{\mathcal{D}}(A, B) = \mathsf{Hom}_{\mathcal{C}}(A, B)$ for all $A, B : \mathsf{Ob}_{\mathcal{D}}$, the subcategory is *full*.

In other words, a subcategory $\mathcal{D}$ of $\mathcal{C}$ is a selection of objects and morphisms from $\mathcal{C}$ that still forms a category, i.e., is closed under identity and composition.

1.2 On the equality of objects

Our definition of category does not include an equivalence relation on Ob. This is by intention, speaking about object equality is not considered pure category-theoretic spirit. All category-theoretic notions should respect isomorphic objects.

Definition 3 (Isomorphism)**.** An *isomorphism* (short *iso*) between two objects A and B is a pair of morphisms $f : A \longrightarrow B$ and $g : B \longrightarrow A$ such that $g \circ f = \mathsf{id}_A$ and $f \circ g = \mathsf{id}_B$. The existence of an isomorphism is written $A \cong B$, and the set of isomorphisms is denoted by $\mathsf{Iso}(A, B)$.

Lemma 1 (Inverse)**.** *Fixing f, the inverse g is uniquely determined and denoted by f^{-1}.*

In other words, being an isomorphism is not a property of a pair of morphisms but a single morphism, that is for each pair of objects A and B the function

$$\mathsf{Iso}(A, B) \to \{f \in \mathsf{Hom}(A, B) \mid \exists g.\, g \circ f = \mathsf{id} \wedge f \circ g = \mathsf{id}\} \quad (f, g) \mapsto f$$

is a bijection of sets with inverse $f \mapsto (f, f^{-1})$.

Exercise 3. Prove this!

Exercise 4 (Subcategory of isomorphisms)**.** Show that the isomorphisms of a category constitute a wide subcategory.

Exercise 5. Does the concept *subcategory* (Definition 2) honor the ideal that no category-theoretic concept should distinguish between isomorphic objects?

If not, suggest a modification of the definition, or defend the current definition against the ideal.

1.3 Operations on categories

Some operations on the object types can be lifted to categories.

1. The product $\mathcal{C} \times \mathcal{D}$ of two categories forms again a category with $\mathsf{Ob}_{\mathcal{C}\times\mathcal{D}} = \mathsf{Ob}_{\mathcal{C}} \times \mathsf{Ob}_{\mathcal{D}}$.
2. The latter can be generalized to nullary, finite, and even infinite products $\prod_{i:I} \mathcal{C}_i$.
3. Any type can be turned into a *discrete* category where the identities are the only morphisms.

Definition 4 (Opposite category)**.** Given a category $\mathcal{C}$, its *opposite* $\mathcal{C}^{\mathsf{op}}$ has the same objects but flipped morphisms, $\mathcal{C}^{\mathsf{op}}(A, B) = \mathcal{C}(B, A)$, and thus flipped composition: $f \circ_{\mathcal{C}^{\mathsf{op}}} g = g \circ_{\mathcal{C}} f$.

Remark 2. The opposite category is really just the original category with morphisms relabeled so that source and target are formally exchanged.

Exercise 6. Show that $\mathcal{C}^{\mathsf{op}}$ is indeed a category.
Show that $(\mathcal{C}^{\mathsf{op}})^{\mathsf{op}} = \mathcal{C}$.

2 Functors and Natural Transformations

A *functor* $F : [\mathcal{C}, \mathcal{D}]$ is a category morphism:

Definition 5 (Functor)**.** Given categories $\mathcal{C}$ and $\mathcal{D}$ a functor $F : [\mathcal{C}, \mathcal{D}]$ is given by the following data:

1. Maps:
   a) A function $F_0 : \mathsf{Ob}_{\mathcal{C}} \to \mathsf{Ob}_{\mathcal{D}}$.
   b) For any pair of objects $A, B : \mathcal{C}$, a function $F_1 : \mathsf{Hom}_{\mathcal{C}}(A, B) \to \mathsf{Hom}_{\mathcal{D}}(F_0 A, F_0 B)$.
2. Laws:
   a) For any object $A : \mathcal{C}$ we have $F_1(\mathsf{id}_A) = \mathsf{id}_{F_0 A}$.
   b) For any pair of morphisms $f : \mathcal{C}(A, B)$ and $g : \mathcal{C}(B, C)$ we have $F_1(g \circ_{\mathcal{C}} f) = F_1 g \circ_{\mathcal{D}} F_1 f$.

c) For any pair of parallel morphisms f, $f' : \mathcal{C}(A, B)$ such that $f = f'$ we have $F_1(f) = F_1(f')$.

It is common to drop the indices 0 and 1 and simply write, e.g., $Ff : FA \longrightarrow FB$. Also, since there is little chance of confusion, one often writes $F : \mathcal{C} \to \mathcal{D}$ instead of $F : [\mathcal{C}, \mathcal{D}]$.

Example 8 (List functor)**.** In the category Set of sets and functions, $\mathsf{List} : \mathsf{Set} \to \mathsf{Set}$ is the (endo)functor that

1. maps a set A to the set $\mathsf{List}\, A$ of lists with elements in A, and
2. maps a function $f : A \longrightarrow B$ to the function $\mathsf{List}(f) : \mathsf{List}\, A \longrightarrow \mathsf{List}\, B$ (aka $\mathsf{map}\, f$) that applies f to each element, i.e., $\mathsf{List}(f)[a_1, \ldots, a_n] = [f(a_1), \ldots, f(a_n)]$.

Example 9 (Forgetful functor)**.** "Forgetting" algebraic structure gives rise to trivial functors, the so-called *forgetful functors*, often denoted by U. For example, $U : \mathsf{Grp} \to \mathsf{Set}$ maps groups to their carriers, and group morphisms to their underlying functions on the carriers.

A forgetful functor does nothing to the "values", only changes their "types".

Exercise 7. Define the duplication functor $\mathsf{Dup} : [\mathcal{C}, \mathcal{C} \times \mathcal{C}]$ from a category to its square.

Since functors are not mathematical structures (such as groups and categories) it is not obvious what the notion of morphism between two functors $F, G : [\mathcal{C}, \mathcal{D}]$ should be. The definition states that it is a family of morphisms $FA \longrightarrow GA$ parametric in A:

Definition 6 (Natural transformation)**.** Given functors $F, G : [\mathcal{C}, \mathcal{D}]$, a *natural transformation* $\eta : F \mathrel{\dot{\to}} G$ is a family of morphisms $\eta_A : FA \longrightarrow GA$ indexed by $A : \mathcal{C}$ such that for all $f : A \longrightarrow B$ we have $Gf \circ \eta_A = \eta_B \circ Ff$.

Diagrammatically, the commutation law can be depicted as follows:

$$\begin{array}{ccc} A & & FA \xrightarrow{\eta_A} GA \\ \downarrow{\scriptstyle f} & \qquad & \downarrow{\scriptstyle Ff} \qquad\qquad \downarrow{\scriptstyle Gf} \\ B & & FB \xrightarrow{\eta_B} GB \end{array}$$

Exercise 8 (Reverse is natural)**.** Show: In Set, the set-indexed family of functions $\mathsf{reverse}_A : \mathsf{List}\, A \longrightarrow \mathsf{List}\, A$ where $\mathsf{reverse}_A[a_1, \ldots, a_n] = [a_n, \ldots, a_1]$ is a natural transformation $\mathsf{reverse} : \mathsf{List} \mathrel{\dot{\to}} \mathsf{List}$.

Exercise 9 (Functor category)**.** Show that functors in $[\mathcal{C}, \mathcal{D}]$ form a category with natural transformations as morphisms.

Definition 7 (Cat)**.** Taking categories $\mathcal{C}$ as objects themselves and functor sets $[\mathcal{C}, \mathcal{D}]$ as homsets, we arrive at the category Cat of categories!

For consistency reasons, $\mathsf{Ob}_{\mathsf{Cat}}$ needs to be a large type containing categories $\mathcal{C}$ whose $\mathsf{Ob}_{\mathcal{C}}$ is a small type.

Exercise 10. Prove that functors are indeed closed under composition and that Cat is indeed a category.

Remark 3 (2-categories)**.** In Cat, the functor types $[\mathcal{C}, \mathcal{D}]$ are only taken as sets, but they are categories themselves! Categories whose homsets are categories again are called *2-categories* or *bicategories*. These have extra structure—we won't dive further into this now.

3 Cartesian Categories

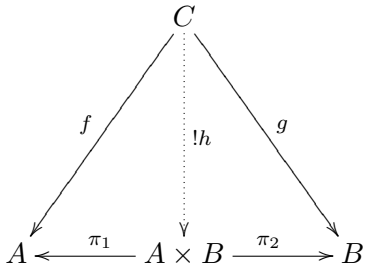
Category theory rarely studies pure categories, but usually categories with extra structure.

Definition 8 (Product)**.** Given $A, B : \mathcal{C}$, a *product* of A and B is given by the following data:

1. An object $P : \mathcal{C}$, and
2. a pair of morphisms $\pi_1 : P \longrightarrow A$ and $\pi_2 : P \longrightarrow B$, such that
3. for each object C and morphisms $f : C \longrightarrow A$ and $g : C \longrightarrow B$ there is a unique morphism $h : C \longrightarrow P$ such that $\pi_1 \circ h = f$ and $\pi_2 \circ h = g$.

The uniqueness of h justifies the notation $h = \langle f, g \rangle$. Since P is unique up to isomorphism (see below), the notation $P = A \times B$ is justified.

The so-called *universal property* that defines the product can be diagrammatically displayed as follows:



Example 10.

1. The cartesian product is the product in Set, Setoid, Grp etc.
2. In Sub, the cartesian product is context concatenation.

Exercise 11. What is a product in a preorder? Under which conditions do preorders have all products?

Exercise 12 (Uniqueness of product)**.** Let (P, π_1, π_2) and (Q, q_1, q_2) be both products of A and B. Show that $P \cong Q$.

Exercise 13 (Commutativity)**.** Show that $A \times B \cong B \times A$.

Exercise 14 (Derived laws)**.** Proof the following theorems using the universal property:

1. $\langle \pi_1, \pi_2 \rangle = \mathsf{id}$.
2. $\langle f, g \rangle \circ h = \langle f \circ h,\ g \circ h \rangle$.

Exercise 15 (Morphism product)**.** Given $f_1 : A_1 \longrightarrow B_1$ and $f_2 : A_2 \longrightarrow B_2$, define $f_1 \times f_2 : A_1 \times A_2 \longrightarrow B_1 \times B_2$.

The nullary product is called the terminal object.

Definition 9 (Terminal object)**.** An object $T : \mathcal{C}$ is *terminal* if for any object C there is a unique morphism $h : C \longrightarrow T$.

The uniqueness of h justifies the notation $h = !_C$. Since T is unique up to isomorphism (see below), it is usually denoted by 1.

Exercise 16. Give, if it exists, the terminal object in the categories Set, Setoid, Grp, Rel.

Because $\mathsf{Set}(1, A) \cong A$, we can represent *elements* of a set A by morphisms $1 \longrightarrow A$. This technique yields a notion of *element* also in categories whose objects are not sets.

Exercise 17. What is a terminal object in a preorder?

Exercise 18. The terminal object is unique up to isomorphism.

Exercise 19. Show $\mathsf{Set}(1, A) \cong A$.

Exercise 20 (Naturality of !)**.** Show that ! is a natural transformation from Id to $\mathsf{K1}$ where $\mathsf{Id} : A \mapsto A$ is the identity functor and $\mathsf{K1} : A \mapsto 1$ the constant functor returning the terminal object.

Exercise 21 (Naturality of pairing)**.** Let $\mathcal{C}$ be a category that has binary products.

1. Complete the definition of the product functor $\_\times\_ : [\mathcal{C} \times \mathcal{C}, \mathcal{C}]$, $\_\times\_(A, B) = A \times B$ with its action $\_\times\_$ on morphisms (see Exercise 15) and prove the functor laws.
2. Formulate (if possible) a naturality statement for pairing $\langle \_, \_ \rangle$ and prove naturality.

Definition 10 (Cartesian (monoidal) category)**.** A *cartesian category*, more precisely, a *cartesian monoidal category*, has finite products (including the nullary one).

Example 11 (Multi-sorted first-order language)**.** A first-order language is given by a set S of sorts and a signature Σ mapping constants c to their first-order function type $\Sigma(c) \in \mathsf{List}(S) \times S$. We write $c : s_1 \times \cdots \times s_n \to s$ when $\Sigma(c) = ([s_1, \ldots, s_n], s)$.

Such a first-order language can be interpreted in a cartesian category $\mathcal{C}$ that has an object A_s for each sort $s \in S$ and a morphism $A_c : A_{s_1} \times \cdots \times A_{s_n} \longrightarrow A_s$ for each constant $c : s_1 \times \cdots \times s_n \to s$.

We can also directly take a cartesian category as the definition of a first-order language, although this forces arbitrary products to be considered sorts and arbitrary projections to be constants of the language.

Exercise 22. Describe the language of integers (with ring operations) and arrays of integers (with the usual operations of initialization, lookup, and update) as a cartesian category.

The special case of a single-sorted language is known as Lawvere theory.

Definition 11 (Lawvere theory)**.** A *Lawvere theory* is a cartesian monoidal category T where each object is isomorphic to a power X^n of a distinguished object X, called the generic object for T.

A *model* A of T is a product-preserving functor $A : [T, \mathsf{Set}]$ in the sense that for each $n \in \mathbb{N}$ the set $A(X^n)$ together with the morphisms $A(\pi_i) : A(X^n) \to A(X)$ for $i \leq n$ is a product of n copies of the set $A(X)$.

Example 12. The Lawvere theory of groups has morphism $e : X^0 \longrightarrow X$ and $op : X^2 \longrightarrow X$ and $inv : X \longrightarrow X$. A specific group can be represented as a model of this theory, e.g., $\mathsf{Int}(X) = \mathbb{Z}$ and $\mathsf{Int}(e) = 0$ and $\mathsf{Int}(op)(i, j) = i + j$ and $\mathsf{Int}(inv)(i) = -i$.

4 Cartesian Closed Categories

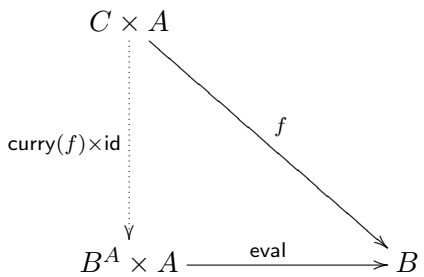
In a cartesian category, we can represent first-order functions as morphisms $f : A_1 \times \cdots \times A_n \longrightarrow B$. To get higher-order functions as in simply-typed lambda-calculus, we need to be able to internalize homsets as objects.

Definition 12. Given $A, B : \mathcal{C}$, an *exponential* of B to the A is given by the following data:

1. An object $E : \mathcal{C}$ with
2. a morphism $\mathsf{eval} : E \times A \longrightarrow B$, such that
3. for each C and $f : C \times A \longrightarrow B$ there is a unique $h : C \longrightarrow E$ such that $\mathsf{eval} \circ (h \times \mathsf{id}_A) = f$.

The uniqueness of h justifies the notation $h = \mathsf{curry}(f)$ (also: $h = \Lambda(f)$ or $h = \lambda(f)$). Since E is unique up to isomorphism, the notation $E = B^A$ or $E = (A \Rightarrow B)$ is justified.

The universal property of exponentials is visualized as follows:



Exercise 23. Explain the exponentials of Set and Setoid! Does Grp have exponentials?

Exercise 24. Give an example of a preorder that has exponentials.

Exercise 25. Show that the exponential is unique up to isomorphism!

Exercise 26 (Derived laws)**.** Prove these laws about exponentials:

1. $\mathsf{curry}(f) \circ h = \mathsf{curry}(f \circ (h \times \mathsf{id}))$.
2. $\mathsf{curry}(\mathsf{eval}) = \mathsf{id}_{B^A}$.
3. $\mathsf{curry}(\mathsf{eval} \circ (f \times \mathsf{id}_A)) = f : C \to B^A$.

Definition 13 (CCC)**.** A *cartesian closed category* has finite products and exponentials.

Exercise 27. Show that Cat is cartesian closed.

References

Aczel, Peter (June 1995). *Galois: A Theory Development Project*. Tech. rep. Available at `http://www.cs.man.ac.uk/~petera/galois.ps.gz`. Departments of Computer Science and Mathematics, Manchester University.

Mac Lane, Saunders (1998). *Categories for the working mathematician*. Second. Vol. 5. Graduate Texts in Mathematics. Springer-Verlag, New York, pp. xii+314. ISBN: 0-387-98403-8.