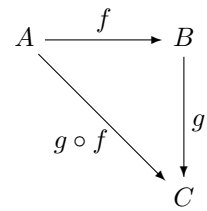
1

CATEGORIES

1.1 Introduction

What is category theory? As a first approximation, one could say that category theory is the mathematical study of (abstract) *algebras of functions.* Just as group theory is the abstraction of the idea of a system of permutations of a set or symmetries of a geometric object, category theory arises from the idea of a system of functions among some objects.



We think of the composition $g \circ f$ as a sort of "product" of the functions f and g, and consider abstract "algebras" of the sort arising from collections of functions. A category is just such an "algebra," consisting of objects $A, B, C, \ldots$ and arrows $f : A \to B$, $g : B \to C, \ldots$, that are closed under composition and satisfy certain conditions typical of the composition of functions. A precise definition is given later in this chapter.

A branch of abstract algebra, category theory was invented in the tradition of Felix Klein's *Erlanger Programm*, as a way of studying and characterizing different kinds of mathematical structures in terms of their "admissible transformations." The general notion of a category provides a characterization of the notion of a "structure-preserving transformation," and thereby of a species of structures admitting such transformations.

The historical development of the subject has been, very roughly, as follows:

1945 Eilenberg and Mac Lane's "General theory of natural equivalences" was the original paper, in which the theory was first formulated.

late 1940s The main applications were originally in the fields of algebraic topology, particularly homology theory, and abstract algebra.

1950s A. Grothendieck et al. began using category theory with great success in algebraic geometry.

1960s F.W. Lawvere and others began applying categories to logic, revealing some deep and surprising connections.

1970s Applications were already appearing in computer science, linguistics, cognitive science, philosophy, and many other areas.

One very striking thing about the field is that it has such wide-ranging applications. In fact, it turns out to be a kind of universal mathematical language like set theory. As a result of these various applications, category theory also tends to reveal certain connections between different fields—like logic and geometry. For example, the important notion of an *adjoint functor* occurs in logic as the existential quantifier and in topology as the image operation along a continuous function. From a categorical point of view these turn out to be essentially the same operation.

The concept of adjoint functor is in fact one of the main things that the reader should take away from the study of this book. It is a strictly category-theoretical notion that has turned out to be a conceptual tool of the first magnitude—on par with the idea of a continuous function.

In fact, just as the idea of a topological space arose in connection with continuous functions, so also the notion of a category arose in order to define that of a functor, at least according to one of the inventors. The notion of a functor arose—so the story goes on—in order to define natural transformations. One might as well continue that natural transformations serve to define adjoints:

Category
Functor
Natural transformation
Adjunction

Indeed, that gives a pretty good outline of this book.

Before getting down to business, let us ask why it should be that category theory has such far-reaching applications. Well, we said that it is the abstract theory of functions, so the answer is simply this:

Functions are everywhere!

And everywhere that functions are, there are categories. Indeed, the subject might better have been called *abstract function theory*, or, perhaps even better: *archery*.

1.2 Functions of sets

We begin by considering functions between sets. I am not going to say here what a function is, anymore than what a set is. Instead, we will assume a working knowledge of these terms. They can in fact be *defined* using category theory, but that is not our purpose here.

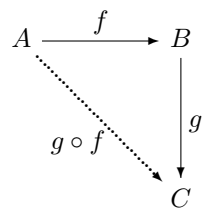
Let f be a function from a set A to another set B, we write

$$f : A \to B.$$

To be explicit, this means that f is defined on all of A and all the values of f are in B. In set theoretic terms,

$$\text{range}(f) \subseteq B.$$

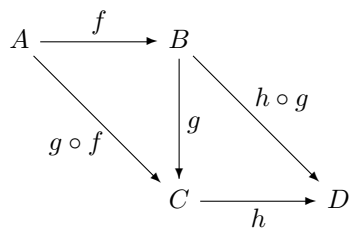
Now suppose we also have a function $g : B \to C$,



then there is a composite function $g \circ f : A \to C$, given by

$$(g \circ f)(a) = g(f(a)) \qquad a \in A. \tag{1.1}$$

Now this operation "$\circ$" of composition of functions is associative, as follows. If we have a further function $h : C \to D$



and form $h \circ g$ and $g \circ f$ then we can compare $(h \circ g) \circ f$ and $h \circ (g \circ f)$ as indicated in the above diagram. It turns out that these two functions are always identical,

$$(h \circ g) \circ f = h \circ (g \circ f)$$

since for any $a \in A$, we have

$$((h \circ g) \circ f)(a) = h(g(f(a))) = (h \circ (g \circ f))(a)$$

using (1.1).

By the way, this is of course what it means for two functions to be equal: for every argument, they have the same value.

Finally, note that every set A has an identity function

$$1_A : A \to A$$

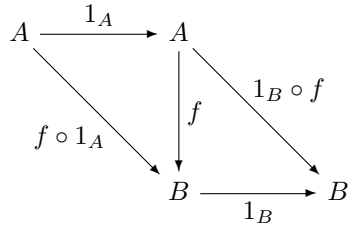
given by

$$1_A(a) = a.$$

These identity functions act as "units" for the operation $\circ$ of composition, in the sense of abstract algebra. That is to say,

$$f \circ 1_A = f = 1_B \circ f$$

for any $f : A \to B$.



These are all the properties of set functions that we want to consider for the *abstract* notion of function: composition and identities. Thus, we now want to "abstract away" everything else, so to speak. That is what is accomplished by the following definition.

1.3 Definition of a category

Definition 1.1. A *category* consists of the following data:

- *Objects*: $A, B, C, \ldots$
- *Arrows*: $f, g, h, \ldots$
- For each arrow f there are given objects:

  $$\mathrm{dom}(f), \qquad \mathrm{cod}(f)$$

  called the *domain* and *codomain* of f. We write:

  $$f : A \to B$$

  to indicate that $A = \mathrm{dom}(f)$ and $B = \mathrm{cod}(f)$.
- Given arrows $f : A \to B$ and $g : B \to C$, that is, with:

  $$\mathrm{cod}(f) = \mathrm{dom}(g)$$

  there is given an arrow:

  $$g \circ f : A \to C$$

called the *composite* of f and g.

- For each object A there is given an arrow:

$$1_A : A \to A$$

called the *identity arrow* of A.

These data are required to satisfy the following laws:

- Associativity:

$$h \circ (g \circ f) = (h \circ g) \circ f$$

for all $f : A \to B$, $g : B \to C$, $h : C \to D$.

- Unit:

$$f \circ 1_A = f = 1_B \circ f$$

for all $f : A \to B$.

A category is *anything* that satisfies this definition—and we will have plenty of examples very soon. For now I want to emphasize that, unlike in the previous section, the objects do not have to be sets and the arrows need not be functions. In this sense, a category is an *abstract* algebra of functions, or "arrows" (sometimes also called "morphisms"), with the composition operation "$\circ$" as primitive. If you are familiar with groups, you may think of a category as a sort of generalized group.

1.4 Examples of categories

1. We have already encountered the category **Sets** of sets and functions. There is also the category

$$\mathbf{Sets}_{\text{fin}}$$

of all finite sets and functions between them.

Indeed, there are many categories like this, given by restricting the sets that are to be the objects and the functions that are to be the arrows. For example, take finite sets as objects and injective (i.e., "1 to 1") functions as arrows. Since injective functions compose to give an injective function, and since the identity functions are injective, this also gives a category.

What if we take sets as objects and as arrows, those $f : A \to B$ such that for all $b \in B$, the subset

$$f^{-1}(b) \subseteq A$$

has at most two elements (rather than one)? Is this still a category? What if we take the functions such that $f^{-1}(b)$ is finite? infinite? There are lots of such restricted categories of sets and functions.

2. Another kind of example one often sees in mathematics is categories of *structured sets*, that is, sets with some further "structure" and functions which "preserve it," where these notions are determined in some independent way. Examples of this kind you may be familiar with are:

   - groups and group homomorphisms,
   - vector spaces and linear mappings,
   - graphs and graph homomorphisms,
   - the real numbers $\mathbb{R}$ and continuous functions $\mathbb{R} \to \mathbb{R}$,
   - open subsets $U \subseteq \mathbb{R}$ and continuous functions $f : U \to V \subseteq \mathbb{R}$ defined on them,
   - topological spaces and continuous mappings,
   - differentiable manifolds and smooth mappings,
   - the natural numbers $\mathbb{N}$ and all recursive functions $\mathbb{N} \to \mathbb{N}$, or as in the example of continuous functions, one can take partial recursive functions defined on subsets $U \subseteq \mathbb{N}$.
   - posets and monotone functions.

   Do not worry if some of these examples are unfamiliar to you. Later on, we will take a closer look at some of them. For now, let us just consider the last of the above examples in more detail.

3. A partially ordered set or *poset* is a set A equipped with a binary relation $a \leq_A b$ such that the following conditions hold for all $a, b, c \in A$:

   reflexivity: $a \leq_A a$,
   transitivity: if $a \leq_A b$ and $b \leq_A c$, then $a \leq_A c$,
   antisymmetry: if $a \leq_A b$ and $b \leq_A a$, then $a = b$.

   For example, the real numbers $\mathbb{R}$ with their usual ordering $x \leq y$ form a poset that is also *linearly* ordered: either $x \leq y$ or $y \leq x$ for any x, y.

   An arrow from a poset A to a poset B is a function

   $$m : A \to B$$

   that is *monotone*, in the sense that, for all $a, a' \in A$,

   $$a \leq_A a' \quad \text{implies} \quad m(a) \leq_B m(a').$$

   What does it take for this to be a category? We need to know that $1_A : A \to A$ is monotone, but that is clear since $a \leq_A a'$ implies $a \leq_A a'$. We also need to know that if $f : A \to B$ and $g : B \to C$ are monotone, then $g \circ f : A \to C$ is monotone. This also holds, since $a \leq a'$ implies $f(a) \leq f(a')$ implies $g(f(a)) \leq g(f(a'))$ implies $(g \circ f)(a) \leq (g \circ f)(a')$. So we have the category **Pos** of posets and monotone functions.

4. The categories that we have been considering so far are examples of what are sometimes called *concrete categories*. Informally, these are categories in which the objects are sets, possibly equipped with some structure, and the arrows are certain, possibly structure-preserving, functions (we shall see later on that this notion is not entirely coherent; see Remark 1.7). But in fact, one way of understanding what category theory is all about is "doing without elements", and replacing them by arrows instead. Let us now take a look at some examples where this point of view is not just optional, but essential.

   Let **Rel** be the following category: take sets as objects and take binary relations as arrows. That is, an arrow $f : A \to B$ is an arbitrary subset $f \subseteq A \times B$. The identity arrow on a set A is the identity relation.

$$1_A = \{(a, a) \in A \times A \mid a \in A\} \subseteq A \times A.$$

   Given $R \subseteq A \times B$ and $S \subseteq B \times C$, define composition $S \circ R$ by

$$(a, c) \in S \circ R \quad \text{iff} \quad \exists b.\ (a, b) \in R \ \&\ (b, c) \in S$$

   that is, the "relative product" of S and R. We leave it as an exercise to show that **Rel** is in fact a category. (What needs to be done?)

   For another example of a category in which the arrows are not "functions," let the objects be finite sets A, B, C and an arrow $F : A \to B$ is a rectangular matrix $F = (n_{ij})_{i<a, j<b}$ of natural numbers with $a = |A|$ and $b = |B|$, where $|C|$ is the number of elements in a set C. The composition of arrows is by the usual matrix multiplication, and the identity arrows are the usual unit matrices. The objects here are serving simply to ensure that the matrix multiplication is defined, but the matrices are not functions between them.

5. *Finite categories*
   Of course, the objects of a category do not have to be sets, either. Here are some very simple examples:

   - The category **1** looks like this:

$$*$$

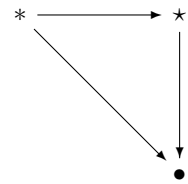
     It has one object and its identity arrow, which we do not draw.

   - The category **2** looks like this:

$$* \longrightarrow \star$$

     It has two objects, their required identity arrows, and exactly one arrow between the objects.

- The category **3** looks like this:



It has three objects, their required identity arrows, exactly one arrow from the first to the second object, exactly one arrow from the second to the third object, and exactly one arrow from the first to the third object (which is therefore the composite of the other two).

- The category **0** looks like this:

It has no objects or arrows.

As above, we will omit the identity arrows in drawing categories from now on.

It is easy to specify finite categories—just take some objects and start putting arrows between them, but make sure to put in the necessary identities and composites, as required by the axioms for a category. Also, if there are any loops, then they need to be cut off by equations to keep the category finite. For example, consider the following specification:

$$A \underset{g}{\overset{f}{\rightleftarrows}} B$$

Unless we stipulate an equation like $gf = 1_A$, we will end up with infinitely many arrows gf, $gfgf$, $gfgfgf$, ... This is still a category, of course, but it is not a *finite* category. We will come back to this situation when we discuss free categories later in this chapter.

6. One important slogan of category theory is,

It's the arrows that really matter!

So we should also look at the arrows or "mappings" between categories. A "homomorphism of categories" is called a functor.

Definition 1.2. A *functor*

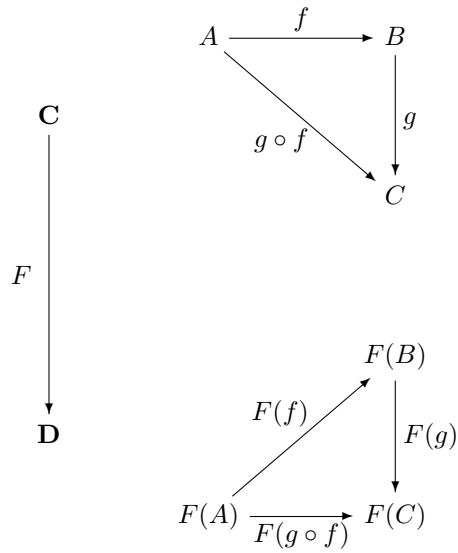
$$F : \mathbf{C} \to \mathbf{D}$$

between categories $\mathbf{C}$ and $\mathbf{D}$ is a mapping of objects to objects and arrows to arrows, in such a way that:

(a) $F(f : A \to B) = F(f) : F(A) \to F(B)$,

(b) $F(1_A) = 1_{F(A)}$,
(c) $F(g \circ f) = F(g) \circ F(f)$.

That is, F preserves domains and codomains, identity arrows, and compostion. A functor $F : \mathbf{C} \to \mathbf{D}$ thus gives a sort of "picture"—perhaps distorted—of $\mathbf{C}$ in $\mathbf{D}$.



Now, one can easily see that functors compose in the expected way, and that every category $\mathbf{C}$ has an identity functor $1_{\mathbf{C}} : \mathbf{C} \to \mathbf{C}$. So we have another example of a category, namely **Cat**, the category of all categories and functors.

7. A *preorder* is a set P equipped with a binary relation $p \leq q$ that is both reflexive and transitive: $a \leq a$, and if $a \leq b$ and $b \leq c$, then $a \leq c$. Any preorder P can be regarded as a category by taking the objects to be the elements of P and taking a unique arrow,

$$a \to b \quad \text{if and only if} \quad a \leq b. \tag{1.2}$$

The reflexive and transitive conditions on $\leq$ ensure that this is indeed a category.

Going in the other direction, any category with at most one arrow between any two objects determines a preorder, simply by defining a binary relation $\leq$ on the objects by (1.2).

8. A poset is evidently a preorder satisfying the additional condition of antisymmetry: if $a \leq b$ and $b \leq a$, then $a = b$. So, in particular, a poset is also a category. Such *poset categories* are very common; for example, for

any set X, the powerset $\mathcal{P}(X)$ is a poset under the usual inclusion relation $U \subseteq V$ between the subsets U, V of X.

What is a functor $F : P \to Q$ between poset categories P and Q? It must satisfy the identity and composition laws .... Clearly, these are just the monotone functions already considered above.

It is often useful to think of a category as a kind of *generalized poset*, one with "more structure" than just $p \leq q$. Thus, one can also think of a functor as a generalized monotone map.

9. *An example from topology:* Let X be a topological space with collection of open sets $\mathcal{O}(X)$. Ordered by inclusion, $\mathcal{O}(X)$ is a poset category. Moreover, the points of X can be preordered by *specialization* by setting $x \leq y$ iff $x \in U$ implies $y \in U$ for every open set U, i.e. y is contained in every open set that contains x. If X is sufficiently separated ("T_1"), then this ordering becomes trivial, but it can be quite interesting otherwise, as happens in the spaces of algebraic geometry and denotational semantics. It is an exercise to show that T_0 spaces are actually posets under the specialization ordering.

10. *An example from logic:* Given a deductive system of logic, there's an associated category *category of proofs*,in which the objects are formulas:

$$\varphi, \psi, \ldots$$

An arrow from φ to ψ is a deduction of ψ from the (uncanceled) assumption φ.

$$\begin{array}{c} \varphi \\ \hline \vdots \\ \hline \psi \end{array}$$

Composition of arrows is given by putting together such deductions in the obvious way, which is clearly associative. (What should the identity arrows 1_φ be?) Observe that there can be many different arrows

$$p : \varphi \to \psi,$$

since there may be many different such proofs. This category turns out to have a very rich structure, which we will consider later in connection with the lambda-calculus.

11. *An example from computer science:* Given a functional programming language L, there is an associated category, where the objects are the data types of L, and the arrows are the computable functions of L ("processes," "procedures," "programs"). The composition of two such programs $X \xrightarrow{f} Y \xrightarrow{g} Z$ is given by applying g to the output of f, sometimes

also written

$$g \circ f = f; g.$$

The identity is the "do nothing" program.

Categories such as this are basic to the idea of denotational semantics of programming languages. For example, if $\mathbf{C}(L)$ is the category just defined, then the denotational semantics of the language L in a category $\mathbf{D}$ of, say, Scott domains is simply a functor

$$S : \mathbf{C}(L) \to \mathbf{D}$$

since S assigns domains to the types of L and continuous functions to the programs. Both this example and the previous one are related to the notion of "cartesian closed category" that is considered later.

12. Let X be a set. We can regard X as a category $\mathbf{Dis}(X)$ by taking the objects to be the elements of X and taking the arrows to be just the required identity arrows, one for each $x \in X$. Such categories, in which the only arrows are identities, are called *discrete*. Note that discrete categories are just very special posets.

13. A *monoid* (sometimes called a *semigroup with unit*) is a set M equipped with a binary operation $\cdot : M \times M \to M$ and a distinguished "unit" element $u \in M$ such that for all $x, y, z \in M$,

$$x \cdot (y \cdot z) = (x \cdot y) \cdot z$$

and

$$u \cdot x = x = x \cdot u.$$

Equivalently, a monoid is a category with just one object. The arrows of the category are the elements of the monoid. In particular, the identity arrow is the unit element u. Composition of arrows is the binary operation $m \cdot n$ of the monoid.

Monoids are very common: there are the monoids of numbers like $\mathbb{N}$, $\mathbb{Q}$ or $\mathbb{R}$ with addition and 0, or multiplication and 1. But also for any set X, the set of functions from X to X, written

$$\mathrm{Hom}_{\mathbf{Sets}}(X, X)$$

is a monoid under the operation of composition. More generally, for any object C in any category $\mathbf{C}$, the set of arrows from C to C, written as $\mathrm{Hom}_{\mathbf{C}}(C, C)$, is a monoid under the composition operation of $\mathbf{C}$.

Since monoids are structured sets, there is a category **Mon** whose objects are monoids and whose arrows are functions that preserve the monoid structure. In detail, a homomorphism from a monoid M to a monoid N is a function $h : M \to N$ such that for all $m, n \in M$,

$$h(m \cdot_M n) = h(m) \cdot_N h(n)$$

and

$$h(u_M) = u_N.$$

Observe that a monoid homomorphism from M to N is the same thing as a functor from M regarded as a category to N regarded as a category. In this sense, categories are also generalized monoids, and functors are generalized homomorphisms.

1.5 Isomorphisms

Definition 1.3. In any category $\mathbf{C}$, an arrow $f : A \to B$ is called an *isomorphism* if there is an arrow $g : B \to A$ in $\mathbf{C}$ such that

$$g \circ f = 1_A \quad \text{and} \quad f \circ g = 1_B.$$

Since inverses are unique (proof!), we write $g = f^{-1}$. We say that A is *isomorphic* to B, written $A \cong B$, if there exists an isomorphism between them.

The definition of isomorphism is our first example of an *abstract*, category theoretic definition of an important notion. It is abstract in the sense that it makes use only of the category theoretic notions, rather than some additional information about the objects and arrows. It has the advantage over other possible definitions that it applies in any category. For example, one sometimes defines an isomorphism of sets (monoids, etc.) as a *bijective* function (resp. homomorphism), i.e., one that is "1-1 and onto"—making use of the *elements* of the objects. This is *equivalent* to our definition in some cases, such as sets and monoids. But note that, for example in **Pos**, the category theoretic definition gives the right notion, while there are "bijective homomorphisms" between non-isomorphic posets. Moreover, in many cases *only* the abstract definition makes sense, as for example, in the case of a monoid regarded as a category.

Definition 1.4. A *group* G is a monoid with an inverse g^{-1} for every element g. Thus G is a category with one object, in which every arrow is an isomorphism.

The natural numbers $\mathbb{N}$ do not form a group under either addition or multiplication, but the integers $\mathbb{Z}$ and the positive rationals $\mathbb{Q}^+$, respectively, do. For any set X, we have the group $\mathrm{Aut}(X)$ of automorphisms (or "permutations") of X, that is, isomorphisms $f : X \to X$. (Why is this closed under "$\circ$"?) A *group of permutations* is a subgroup $G \subseteq Aut(X)$ for some set X, that is, a group of (some) automorphisms of X. Thus the set G must satisfy the following:

1. The identity function 1_X on X is in G.
2. If $g, g' \in G$, then $g \circ g' \in G$.
3. If $g \in G$, then $g^{-1} \in G$.

A *homomorphism* of groups $h : G \to H$ is just a homomorphism of monoids, which then necessarily also preserves the inverses (proof!).

Now consider the following basic, classical result about abstract groups:

Theorem (Cayley). *Every group G is isomorphic to a group of permutations.*

Proof. (sketch)

1. First, define the Cayley representation $\bar{G}$ of G to be the following group of permutations of a set: the set is just G itself, and for each element $g \in G$, we have the permutation $\bar{g} : G \to G$, defined for all $h \in G$ by "acting on the left":

$$\bar{g}(h) = g \cdot h.$$

   This is indeed a permutation, since it has the action of g^{-1} as an inverse.
2. Next define homomorphisms $i : G \to \bar{G}$ by $i(g) = \bar{g}$, and $j : \bar{G} \to G$ by $j(\bar{g}) = \bar{g}(u)$.
3. Finally show that $i \circ j = 1_{\bar{G}}$ and $j \circ i = 1_G$.

□

Warning 1.5. Note the two different levels of isomorphisms that occur in the proof of Cayley's theorem. There are permutations of the set of elements of G, which are isomorphisms in **Sets**, and there is the isomorphism between G and $\bar{G}$, which is in the category **Groups** of groups and group homomorphisms.

Cayley's theorem says that any abstract group can be represented as a "concrete" one, that is, a group of permutations of a set. The theorem can in fact be generalized to show that any category that is not "too big" can be represented as one that is "concrete," that is, a category of sets and functions. (There is a technical sense of not being "too big" that will be introduced in Section 1.8.)

Theorem 1.6. *Every category* $\mathbf{C}$ *with a set of arrows is isomorphic to one in which the objects are sets and the arrows are functions.*

Proof. (sketch) Define the Cayley representation $\bar{\mathbf{C}}$ of $\mathbf{C}$ to be the following concrete category:

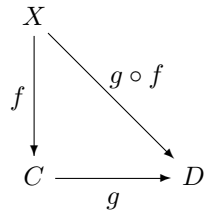
- objects are sets of the form

$$\bar{C} = \{f \in \mathbf{C} \mid \operatorname{cod}(f) = C\}$$

  for all $C \in \mathbf{C}$,
- arrows are functions

$$\bar{g} : \bar{C} \to \bar{D}$$

for $g : C \to D$ in $\mathbf{C}$, defined for any $f : X \to C$ in $\bar{C}$ by $\bar{g}(f) = g \circ f$.



□

Remark 1.7. This shows us what is *wrong* with the naive notion of a "concrete" category of sets and functions: while not every category has special sets and functions as its objects and arrows, every category is isomorphic to such a one. Thus, the only special properties such categories can possess are ones that are categorically irrelevant, such as features of the objects that do not affect the arrows in any way (like the difference between the real numbers constructed as Dedekind cuts or as Cauchy sequences). A better attempt to capture what is intended by the rather vague idea of a "concrete" category is that arbitrary arrows $f : C \to D$ are completely determined by their composites with arrows $x : T \to C$ from some "test object" T, in the sense that $fx = gx$ for all such x implies $f = g$. As we shall see later, this amounts to considering a particular representation of the category, determined by T. A category is then said to be "concrete" when this condition holds for T a "terminal object," in the sense of Section 2.2; but there are also good reasons for considering other objects T, as we see in the next chapter.

Note that the condition that $\mathbf{C}$ have a *set* of arrows is needed to ensure that the collections $\{f \in \mathbf{C} \mid \mathrm{cod}(f) = C\}$ really are *sets*—we return to this point in Section 1.8.

1.6 Constructions on categories

Now that we have a stock of categories to work with, we can consider some constructions that produce new categories from old.

1. The *product* of two categories $\mathbf{C}$ and $\mathbf{D}$, written

$$\mathbf{C} \times \mathbf{D}$$

has objects of the form (C, D), for $C \in \mathbf{C}$ and $D \in \mathbf{D}$, and arrows of the form

$$(f, g) : (C, D) \to (C', D')$$

for $f : C \to C' \in \mathbf{C}$ and $g : D \to D' \in \mathbf{D}$. Composition and units are defined componentwise; that is,

$$(f', g') \circ (f, g) = (f' \circ f, g' \circ g)$$
$$1_{(C,D)} = (1_C, 1_D).$$

There are two obvious *projection functors*

$$\mathbf{C} \xleftarrow{\pi_1} \mathbf{C} \times \mathbf{D} \xrightarrow{\pi_2} \mathbf{D}$$

defined by $\pi_1(C, D) = C$ and $\pi_1(f, g) = f$, and similarly for π_2.

The reader familiar with groups will recognize that for groups G and H, the product category $G \times H$ is the usual (direct) product of groups.

2. The *opposite* (or "dual") category $\mathbf{C}^{\mathrm{op}}$ of a category $\mathbf{C}$ has the same objects as $\mathbf{C}$, and an arrow $f : C \to D$ in $\mathbf{C^{op}}$ is an arrow $f : D \to C$ in C. That is $\mathbf{C}^{\mathrm{op}}$ is just $\mathbf{C}$ with all of the arrows formally turned around.

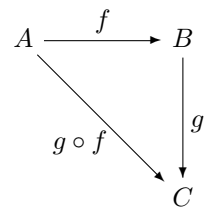
It is convenient to have a notation to distinguish an object (resp. arrow) in $\mathbf{C}$ from the same one in $\mathbf{C^{op}}$. Thus, let us write

$$f^* : D^* \to C^*$$

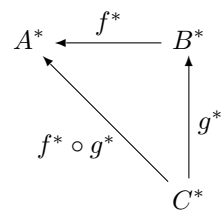
in $\mathbf{C}^{\mathrm{op}}$ for $f : C \to D$ in $\mathbf{C}$. With this notation we can define composition and units in $\mathbf{C}^{\mathrm{op}}$ in terms of the corresponding operations in $\mathbf{C}$, namely,

$$1_{C^*} = (1_C)^*$$
$$f^* \circ g^* = (g \circ f)^*.$$

Thus, a diagram in $\mathbf{C}$



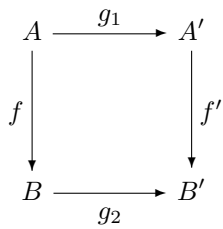
looks like this in $\mathbf{C}^{\mathrm{op}}$



Many "duality" theorems of mathematics express the fact that one category is (a subcategory of) the opposite of another. An example of this sort which

we will prove later is that **Sets** is dual to the category of complete, atomic Boolean algebras.

3. The *arrow category* $\mathbf{C}^{\to}$ of a category $\mathbf{C}$ has the arrows of $\mathbf{C}$ as objects, and an arrow g from $f : A \to B$ to $f' : A' \to B'$ in $\mathbf{C}^{\to}$ is a "commutative square"



where g_1 and g_2 are arrows in $\mathbf{C}$. That is, such an arrow is a pair of arrows $g = (g_1, g_2)$ in $\mathbf{C}$ such that

$$g_2 \circ f = f' \circ g_1.$$

The identity arrow 1_f on an object $f : A \to B$ is the pair $(1_A, 1_B)$. Composition of arrows is done componentwise:

$$(h_1, h_2) \circ (g_1, g_2) = (h_1 \circ g_1, h_2 \circ g_2)$$

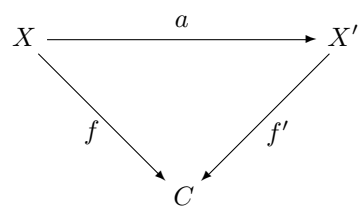
The reader should verify that this works out by drawing the appropriate commutative diagram.

Observe that there are two functors:

$$\mathbf{C} \xleftarrow{\textbf{dom}} \mathbf{C}^{\to} \xrightarrow{\textbf{cod}} \mathbf{C}$$

4. The *slice category* $\mathbf{C}/C$ of a category $\mathbf{C}$ over an object $C \in \mathbf{C}$ has:

   - objects: all arrows $f \in \mathbf{C}$ such that $\mathrm{cod}(f) = C$,
   - arrows: an arrow a from $f : X \to C$ to $f' : X' \to C$ is an arrow $a : X \to X'$ in $\mathbf{C}$ such that $f' \circ a = f$, as indicated in

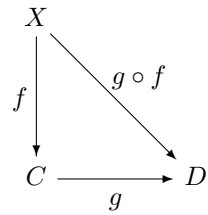


The identity arrows and composites are inherited from those of $\mathbf{C}$, just as in the arrow category. Note that there is a functor $U : \mathbf{C}/C \to \mathbf{C}$ that "forgets about the base object C."

If $g : C \to D$ is any arrow, then there is a composition functor,

$$g_* : \mathbf{C}/C \to \mathbf{C}/D$$

defined by $g_*(f) = g \circ f$,



and similarly for arrows in $\mathbf{C}/C$. Indeed, the whole construction is a functor,

$$\mathbf{C}/(-) : \mathbf{C} \to \mathbf{Cat}$$

as the reader can easily verify. Compared to the Cayley representation, this functor gives a "representation" of $\mathbf{C}$ as a category of categories and functors — rather than sets and fuctions. Of course, the Cayley representation was just this one followed by the forgetful functor $U : \mathbf{Cat} \to \mathbf{Sets}$ which takes a category to its underlying set of objects.

If $\mathbf{C} = \mathbf{P}$ is a poset category and $p \in \mathbf{P}$, then

$$\mathbf{P}/p \cong {\downarrow}(p)$$

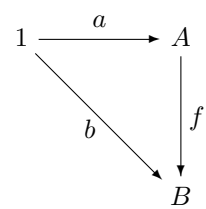
the slice category $\mathbf{P}/p$ is just the "principal ideal" ${\downarrow}(p)$ of elements $q \in \mathbf{P}$ with $q \leq p$. We will have more examples of slice categories soon.

The *coslice* category $C/\mathbf{C}$ of a category $\mathbf{C}$ under an object C of $\mathbf{C}$ has as objects all arrows f of $\mathbf{C}$ such that $\mathrm{dom}(f) = C$, and an arrow from $f : C \to X$ to $f' : C \to X'$ is an arrow $h : X \to X'$ such that $h \circ f = f'$. The reader should now carry out the rest of the definition of the coslice category by analogy with the definition of the slice category. How can the coslice category be defined in terms of the slice category and the opposite construction?

Example 1.8. The category $\mathbf{Sets}_*$ of *pointed sets* consists of sets A with a distinguished element $a \in A$, and arrows $f : (A, a) \to (B, b)$ are functions $f : A \to B$ that preserves the "points," $f(a) = b$. This is isomorphic to the coslice category,

$$\mathbf{Sets}_* \cong 1/\mathbf{Sets}$$

of Sets "under" any singleton $1 = \{*\}$. Indeed, functions $a : 1 \to A$ correspond uniquely to elements, $a(*) = a \in A$, and arrows $f : (A, a) \to (B, b)$ correspond exactly to commutative triangles:



1.7 Free categories

Free monoid. Start with an "alphabet" A of "letters" $a, b, c, \ldots$, i.e. a set,

$$A = \{a, b, c, \ldots\}.$$

A *word* over A is a finite sequence of letters:

$$thisword, \quad categoriesarefun, \quad asddjbnzzfj, \ldots$$

We write "-" for the empty word. The "Kleene closure" of A is defined to be the set

$$A^* = \{\text{words over } A\}.$$

Define a binary operation "$*$" on A^* by $w * w' = ww'$ for words $w, w' \in A^*$. Thus, "$*$" is just *concatenation*. The operation "$*$" is thus associative, and the empty word "-" is a unit. Thus, A^* is a monoid—called the *free monoid* on the set A. The elements $a \in A$ can be regarded as words of length one, so we have a function

$$i : A \to A^*$$

defined by $i(a) = a$, and called the "insertion of generators." The elements of A "generate" the free monoid, in the sense that every $w \in A^*$ is a $*$-product of a's, that is, $w = a_1 * a_2 * \cdots * a_n$ for some $a_1, a_2, ..., a_n$ in A.

Now what does "free" mean here? Any guesses? One sometimes sees definitions in "baby algebra" books along the following lines:

A monoid M is *freely generated* by a subset A of M, if the following conditions hold.

1. Every element $m \in M$ can be written as a product of elements of A

$$m = a_1 \cdot_M \ldots \cdot_M a_n \ , \quad a_i \in A.$$

2. No "nontrivial" relations hold in M, that is, if $a_1 \ldots a_j = a'_1 \ldots a'_k$, then this is required by the axioms for monoids.

The first condition is sometimes called "no junk," while the second condition is sometimes called "no noise." Thus, the free monoid on A is a monoid containing A and having no junk and no noise. What do you think of this definition of a free monoid?

I would object to the reference in the second condition to "provability," or something. This must be made more precise for this to succeed as a definition. In category theory, we give a precise definition of "free"—capturing what is meant in the above—which avoids such vagueness.

First, every monoid N has an underlying set $|N|$, and every monoid homomorphism $f : N \to M$ has an underlying function $|f| : |N| \to |M|$. It is easy to see that this is a functor, called the "forgetful functor." The free

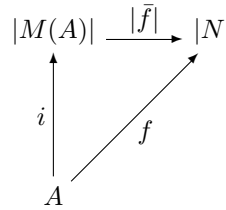
monoid $M(A)$ on a set A is by definition "the" monoid with the following so called *universal mapping property*, or UMP!

Universal Mapping Property of $M(A)$
There is a function $i : A \to |M(A)|$, and given any monoid N and any function $f : A \to |N|$, there is a *unique* monoid homomorphism $\bar{f} : M(A) \to N$ such that $|\bar{f}| \circ i = f$, all as indicated in the following diagram:

in **Mon***:*

$$M(A) \overset{\bar{f}}{\dashrightarrow} N$$

in **Sets***:*



Proposition 1.9. *A^* has the UMP of the free monoid on A.*

Proof. Given $f : A \to |N|$, define $\bar{f} : A^* \to N$ by

$$\bar{f}(-) = u_N, \quad \text{the unit of } N$$
$$\bar{f}(a_1 \dots a_i) = f(a_1) \cdot_N \dots \cdot_N f(a_i).$$

Then $\bar{f}$ is clearly a homomorphism with

$$\bar{f}(a) = f(a) \quad \text{for all } a \in A.$$

If $g : A^* \to N$ also satisfies $g(a) = f(a)$ for all $a \in A$, then for all $a_1 \dots a_i \in A^*$:

$$\begin{aligned} g(a_1 \dots a_i) &= g(a_1 * \dots * a_i) \\ &= g(a_1) \cdot_N \dots \cdot_N g(a_i) \\ &= f(a_1) \cdot_N \dots \cdot_N f(a_i) \\ &= \bar{f}(a_1) \cdot_N \dots \cdot_N \bar{f}(a_i) \\ &= \bar{f}(a_1 * \dots * a_i) \\ &= \bar{f}(a_1 \dots a_i). \end{aligned}$$

So, $g = \bar{f}$, as required. □

Think about why the above UMP captures precisely what is meant by "no junk" and "no noise." Specifically, the existence part of the UMP captures the vague notion of "no noise" (because any equation that holds between algebraic combinations of the generators must also hold anywhere they can be mapped to,

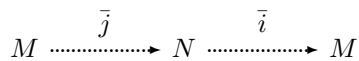
and thus everywhere), while the uniqueness part makes precise the "no junk" idea (because any extra elements not combined from the generators would be free to be mapped to *different* values).

Using the UMP, it is easy to show that the free monoid $M(A)$ is determined uniquely up to isomorphism, in the following sense.

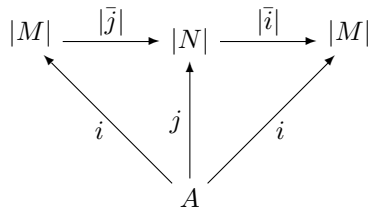
Proposition 1.10. *Given monoids M and N with functions $i : A \to |M|$ and $j : A \to |N|$, each with the UMP of the free monoid on A, there is a (unique) monoid isomorphism $h : M \cong N$ such that $|h|i = j$ and $|h^{-1}|j = i$.*

Proof. From j and the UMP of M, we have $\bar{j} : M \to N$ with $|\bar{j}|i = j$ and from i and the UMP of N, we have $\bar{i} : N \to M$ with $|\bar{i}|j = i$. Composing gives a homomorphism $\bar{i} \circ \bar{j} : M \to M$ such that $|\bar{i} \circ \bar{j}|i = i$. Since $1_M : M \to M$ also has this property, by the uniqueness part of the UMP of M, we have $\bar{i} \circ \bar{j} = 1_M$. Exchanging the roles of M and N shows $\bar{j} \circ \bar{i} = 1_N$;

in **Mon***:*



in **Sets***:*

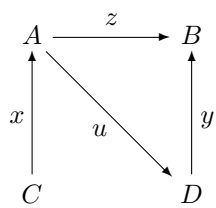


□

For example, the free monoid on any set with a single element is easily seen to be isomorphic to the monoid of natural numbers $\mathbb{N}$ under addition (the "generator" is the number 1). Thus, as a monoid, $\mathbb{N}$ is uniquely determined up to isomorphism by the UMP of free monoids.

Free category. Now, we want to do the same thing for categories in general (not just monoids). Instead of underlying sets, categories have underlying graphs, so let us review these first.

A *directed graph* consists of vertices and edges, each of which is directed, that is, each edge has a "source" and a "target" vertex.



We draw graphs just like categories, but there is no composition of edges, and there are no identities.

A graph thus consists of two sets, E (edges) and V (vertices), and two functions, $s : E \to V$ (source) and $t : E \to V$ (target). Thus, in **Sets**, a graph is just a configuration of objects and arrows of the form:

$$E \underset{t}{\overset{s}{\rightrightarrows}} V$$

Now, every graph G "generates" a category $\mathbf{C}(G)$, the *free category* on G. It is defined by taking the vertices of G as objects, and the *paths* in G as arrows, where a path is a finite sequence of edges $e_1, \ldots, e_n$ such that $t(e_i) = s(e_{i+1})$, for all $i = 1 \ldots n$. We'll write the arrows of $\mathbf{C}(G)$ in the form $e_n e_{n-1} \ldots e_1$.

$$v_0 \xrightarrow{e_1} v_1 \xrightarrow{e_2} v_2 \xrightarrow{e_3} \ldots \xrightarrow{e_n} v_n$$

Put:

$$\mathrm{dom}(e_n \ldots e_1) = s(e_1)$$
$$\mathrm{cod}(e_n \ldots e_1) = t(e_n)$$

and define composition by concatenation:

$$e_n \ldots e_1 \circ e'_m \ldots e'_1 = e_n \ldots e_1 e'_m \ldots e'_1.$$

For each vertex v, we have an "empty path" denoted 1_v, which is to be the identity arrow at v.

Note that if G has only one vertex, then $\mathbf{C}(G)$ is just the free monoid on the set of edges of G. Also note that if G has only vertices (no edges), then $\mathbf{C}(G)$ is the discrete category on the set of vertices of G.

Later on, we will have a general definition of "free." For now, let us see that $\mathbf{C}(G)$ also has a UMP. First, define a "forgetful functor"

$$U : \mathbf{Cat} \to \mathbf{Graphs}$$

in the obvious way: the underlying graph of a category $\mathbf{C}$ has as edges the arrows of $\mathbf{C}$, and as vertices the objects, with $s = \mathrm{dom}$ and $t = \mathrm{cod}$. The action of U on functors is equally clear, or at least it will be, once we have defined the arrows in **Graphs**.

A homomorphism of graphs is of course a "functor without the conditions on identities and composition," that is, a mapping of edges to edges and vertices to vertices that preserves sources and targets. We will describe this from a slightly different point of view, which will be useful later on.

First, observe that we can describe a category $\mathbf{C}$ with a diagram like this:

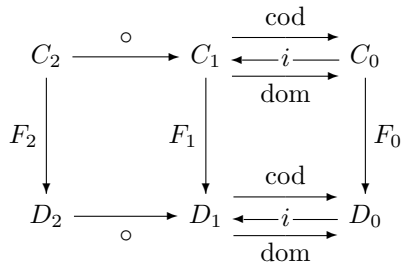
$$C_2 \xrightarrow{\circ} C_1 \underset{\mathrm{dom}}{\overset{\mathrm{cod}}{\rightleftarrows}} C_0 \quad \text{with } i : C_0 \to C_1$$

where C_0 is the collection of objects of $\mathbf{C}$, C_1 the arrows, i is the identity arrow operation, and C_2 is the collection $\{(f, g) \in C_1 \times C_1 : \mathrm{cod}(f) = \mathrm{dom}(g)\}$.

Then a functor $F : \mathbf{C} \to \mathbf{D}$ from $\mathbf{C}$ to another category $\mathbf{D}$ is a pair of functions

$$F_0 : C_0 \to D_0$$
$$F_1 : C_1 \to D_1$$

such that each similarly labeled square in the following diagram commutes:

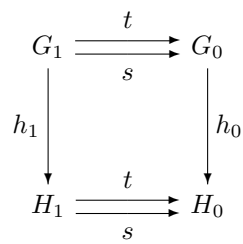


where $F_2(f, g) = (F_1(f), F_1(g))$.

Now let us describe a *homomorphism of graphs*,

$$h : G \to H.$$

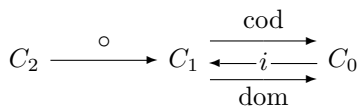
We need a pair of functions $h_0 : G_0 \to H_0$, $h_1 : G_1 \to H_1$ making the two squares (once with t's, once with s's) in the following diagram commute:



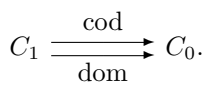
In these terms, we can easily describe the forgetful functor,

$$U : \mathbf{Cat} \to \mathbf{Graphs}$$

as sending the category



to the underlying graph



And similarly for functors, the effect of U is described by simply erasing some parts of the diagrams (which is easier to demonstrate with chalk!). Let us again write $|\mathbf{C}| = U(\mathbf{C})$, etc., for the underlying graph of a category $\mathbf{C}$, in analogy to the case of monoids above.

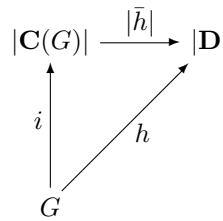
The free category on a graph now has the following UMP:

Universal Mapping Property of $\mathbf{C}(G)$
There is a graph homomorphism $i : G \to |\mathbf{C}(G)|$, and given any category $\mathbf{D}$ and any graph homomorphism $h : G \to |\mathbf{D}|$, there is a *unique* functor $\bar{h} : \mathbf{C}(G) \to \mathbf{D}$ with $|\bar{h}| \circ i = h$.

in **Cat***:*

$$\mathbf{C}(G) \overset{\bar{h}}{\dashrightarrow} \mathbf{D}$$

in **Graph***:*



The free category on a graph with just one vertex is just a free monoid on the set of edges. The free category on a graph with two vertices and one edge between them is the finite category **2**. The free category on a graph of the form:

$$A \underset{f}{\overset{e}{\rightleftarrows}} B$$

has (in addition to the identity arrows) the infinitely many arrows:

$$e, f, ef, fe, efe, fef, efef, ...$$

1.8 Foundations: large, small, and locally small

Let us begin by distinguishing between the following things:

categorical foundations for mathematics,
mathematical foundations for category theory.

As for the first: one sometimes hears it said that category theory can be used to provide "foundations for mathematics," as an alternative to set theory. That is in fact the case, but it is not what we are doing here. In set theory, one often begins with existential axioms such as "there is an infinite set" and derives further sets by axioms like "every set has a powerset," thus building up a

universe of mathematical objects (namely sets), which in principle suffice for "all of mathematics." Our axiom that every arrow has a domain and a codomain is not to be understood in the same way as set theory's axiom that every set has a powerset! The difference is that in set theory—at least as usually conceived—the axioms are to be regarded as referring to (or determining) a single universe of sets. In category theory, by contrast, the axioms are a *definition* of something, namely of categories. This is just like in group theory or topology, where the axioms serve to define the objects under investigation. These, in turn, are assumed to exist in some "background" or "foundational" system, like set theory (or type theory). That theory of sets could itself, in turn, be determined using category theory, or in some other way.

This brings us to the second point: we assume that our categories are comprised of sets and functions, in one way or another, like most mathematical objects, and taking into account the remarks just made about the possibility of categorical (or other) foundations. But in category theory, we sometimes run into difficulties with set theory as usually practiced. Mostly these are questions of size; some categories are "too big" to be handled comfortably in conventional set theory.We already encountered this issue when we considered the Cayley representation in Section 1.5. There we had to require that the category under consideration had (no more than) a set of arrows. We would certainly not want to impose this restriction in general, however (as one usually does for, say, groups); for then even the "category" **Sets** would fail to be a proper category, as would many other categories that we definitely want to study.

There are various formal devices for addressing these issues, and they are discussed in the book by Mac Lane. For our immediate purposes, the following distinction will be useful:

Definition 1.11. A category $\mathbf{C}$ is called *small* if both the collection $\mathbf{C}_0$ of objects of $\mathbf{C}$ and the collection $\mathbf{C}_1$ of arrows of $\mathbf{C}$ are sets. Otherwise, $\mathbf{C}$ is called *large*.

For example, all finite categories are clearly small, as is the category $\mathbf{Sets}_{\text{fin}}$ of finite sets and functions. (Actually, one should stipulate that the sets are only built from other finite sets, all the way down, i.e. that they are "hereditarily finite".) On the other hand, the category **Pos** of posets, the category **Groups** of groups, and the category **Sets** of sets are all large. We let **Cat** be the category of all *small categories*, which itself is a large category. In particular, then, **Cat** is not an object of itself, which may come as a relief to some readers.

This does not really solve all of our difficulties. Even for large categories like **Groups** and **Sets** we will want to also consider constructions like the category of all functors from one to the other (we will define this "functor category" later). But if these are not small, conventional set theory does not provide the means to do this directly (these categories would be "too large"). So, one needs a more elaborate theory of "classes" to handle such constructions. We will not worry about this when it is just a matter of technical foundations (Mac Lane I.6

addresses this issue). However, one very useful notion in this connection is the following:

Definition 1.12. A category $\mathbf{C}$ is called *locally small* if for all objects X, Y in $\mathbf{C}$, the collection $\mathrm{Hom}_{\mathbf{C}}(X,Y) = \{f \in \mathbf{C}_1 \mid f : X \to Y\}$ is a *set* (called a *hom-set*).

Many of the large categories we want to consider are in fact locally small. **Sets** is locally small since $\mathrm{Hom}_{\mathbf{Sets}}(X,Y) = Y^X$, the *set* of all functions from X to Y. Similarly, **Pos**, **Top**, and **Group** are all locally small (is **Cat**?), and, of course, any small category is locally small.

Warning 1.13. Don't confuse the notions *concrete* and *small*. To say that a category is concrete is to say that the *objects* of the category are (structured) sets, and the arrows of the category are (certain) functions. To say that a category is small is to say that the *collection of all objects* of the category is a set, as is the collection of all arrows. The real numbers $\mathbb{R}$, regarded as a poset category, is small but not concrete. The category **Pos** of all posets is concrete but not small.

1.9 Exercises

1. The objects of **Rel** are sets, and an arrow $f : A \to B$ is a relation from A to B, that is, a subset $f \subseteq A \times B$. The equality relation $\{\langle a, a\rangle \in A \times A \mid a \in A\}$ is the identity arrow on a set A. Composition in **Rel** is to be given by

$$g \circ f = \{\langle a, c\rangle \in A \times C \mid \exists b\,(\langle a, b\rangle \in f \;\&\; \langle b, c\rangle \in g)\}$$

   for $f \subseteq A \times B$ and $g \subseteq B \times C$.
   Show that **Rel** is a category.
2. Consider the following isomorphisms of categories and determine which hold.
   (a) $\mathbf{Rel} \cong \mathbf{Rel}^{\mathrm{op}}$
   (b) $\mathbf{Sets} \cong \mathbf{Sets}^{\mathrm{op}}$
   (c) For a fixed set X with powerset $P(X)$, as poset categories $P(X) \cong P(X)^{\mathrm{op}}$ (the arrows in $P(X)$ are subset inclusions $A \subseteq B$ for all $A, B \subseteq X$).
3. (a) Show that in **Sets**, the isomorphisms are exactly the bijections.
   (b) Show that in **Monoids**, the isomorphisms are exactly the bijective homomorphisms.
   (c) Show that in **Posets**, the isomorphisms are *not* the same as the bijective homomorphisms.
4. Let X be a topological space and preorder the points by *specialization*: $x \leq y$ iff y is contained in every open set that contains x. Show that this

is a preorder, and that it is a poset if X is T_0 (for any two distinct points, there is some open set containing one but not the other). Show that the ordering is trivial if X is T_1 (for any two distinct points, each is contained in an open set not containing the other).

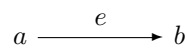
5. For any category $\mathbf{C}$, define a functor $U : \mathbf{C}/C \to \mathbf{C}$ from the slice category over an object C that "forgets about C". Find a functor $F : \mathbf{C}/C \to \mathbf{C}^{\to}$ to the arrow category such that $\mathbf{dom} \circ F = U$.
6. Construct the "coslice category" $C/\mathbf{C}$ of a category $\mathbf{C}$ under an object C from the slice category $\mathbf{C}/C$ and the "dual category" operation $-^{\mathrm{op}}$.
7. Let $2 = \{a, b\}$ be any set with exactly 2 elements a and b. Define a functor $F : \mathbf{Sets}/2 \to \mathbf{Sets} \times \mathbf{Sets}$ with $F(f : X \to 2) = (f^{-1}(a), f^{-1}(b))$. Is this an isomorphism of categories? What about the analogous situation with a one element set $1 = \{a\}$ instead of 2?
8. Any category $\mathbf{C}$ determines a preorder $P(\mathbf{C})$ by defining a binary relation $\leq$ on the objects by:

$$A \leq B \text{ if and only if there is an arrow } A \to B$$

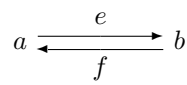
Show that P determines a functor from categories to preorders, by defining its effect on functors between categories and checking the required conditions. Show that P is a (one-sided) inverse to the evident inclusion functor of preorders into categories.

9. Describe the free categories on the following graphs by determining their objects, arrows, and composition operations.

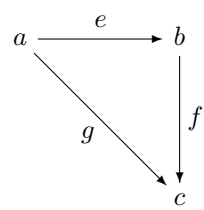
(a)



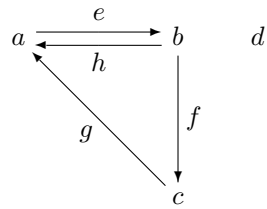
(b)



(c)



(d)



10. How many free categories on graphs are there which have exactly six arrows? Draw the graphs that generate these categories.
11. Show that the free monoid functor

$$M : \mathbf{Sets} \to \mathbf{Mon}$$

exists, in two different ways:

(a) Assume the particular choice $M(X) = X^*$ and define its effect

$$M(f) : M(A) \to M(B)$$

on a function $f : A \to B$ to be

$$M(f)(a_1 \dots a_k) = f(a_1) \dots f(a_k), \qquad a_1, \dots a_k \in A.$$

(b) Assume only the UMP of the free monoid and use it to determine M on functions, showing the result to be a functor.

Reflect on how these two approaches are related.

12. Verify the UMP for free categories on graphs, defined as above with arrows being sequences of edges. Specifically, let $\mathbf{C}(G)$ be the free category on the graph G, so defined, and $i : G \to U(\mathbf{C}(G))$ the graph homomorphism taking vertices and edges to themselves, regarded as objects and arrows in $\mathbf{C}(G)$. Show that for any category $\mathbf{D}$ and graph homomorphism $f : G \to U(\mathbf{D})$, there is a unique functor

$$\bar{h} : \mathbf{C}(G) \to \mathbf{D}$$

with

$$U(\bar{h}) \circ i = h,$$

where $U : \mathbf{Cat} \to \mathbf{Graph}$ is the underlying graph functor.

13. Use the Cayley representation to show that every small category is isomorphic to a "concrete" one, i.e. one in which the objects are sets and the arrows are functions between them.
14. The notion of a category can also be defined with just one sort (arrows) rather than two (arrows and objects); the domains and codomains are taken to be certain *arrows* that act as units under composition, which is

partially defined. Read about this definition in section I.1 of Mac Lane's *Categories for the Working Mathematician*, and do the exercise mentioned there, showing that it is equivalent to the usual definition.