1

Categories

This chapter gives the definition of ‘category’ in Section 1.1, and follows that by four sections devoted entirely to examples of categories of various kinds. If you have never met the notion of a category before, you should quite quickly read through Definition 1.1.1 and then go to Section 1.2. There you will find some examples of categories that you are familiar with, although you may not have recognized the categorical structure before. In this way you will begin to see what Definition 1.1.1 is getting at. After that you can move around the chapter as you like.

Remember that it is probably better not to start at this page and read each word, sentence, paragraph, . . . , in turn. Move around a bit. If there is something you don’t understand, or don’t see the point of, then leave it for a while and come back to it later.

Life isn’t linear, but written words are.

1.1 Categories defined

This section contains the definition of ‘category’, follows that with a few bits and pieces, and concludes with a discussion of some examples. No examples are looked at in detail, that is done in the remaining four sections. Section 1.2 contains a collection of simpler examples, some of which you will know already. You might want to dip into that section as you read this section. In the first instance you should find a couple of examples that you already know. As you become familiar with the categorical ideas you should look at some of the more complicated examples given in the later sections.

The following definition doesn’t quite give all the required information. There are a couple of restrictions that are needed and which are described in detail in the paragraphs following.

1.1.1 Definition A category C consists of

- a collection Obj of entities called objects
- a collection Arw of entities called arrows
- two assignments $\quad Arw \underset{target}{\overset{source}{\rightrightarrows}} Obj$
- an assignment $\quad Obj \xrightarrow{id} Arw$
- a partial composition $\quad Arw \times Arw \longrightarrow Arw$

where this data must satisfy certain restrictions as described below. □

Before we look at the restrictions on this data let's fix some notation.

- We let $A, B, C, \ldots$ range over objects.
- We let $f, g, h, \ldots$ range over arrows.

This convention isn't always used. For instance, sometimes $a, b, c, \ldots$ range over objects, and $\alpha, \beta, \gamma, \ldots$ or $\theta, \phi, \psi, \ldots$ range over arrows. The notation used depends on what is convenient at the time and what is the custom in the topic under discussion. Here we will take the above convention as the norm, but sometimes we will use other notations.

There are two assignments

$$source \qquad target$$

each of which attaches an object to an arrow, that is each consumes an arrow and returns an object. We write

$$A \xrightarrow{f} B$$

to indicate that f is an arrow with source A and target B. This is a small example of a diagram. Later we will see some slightly bigger ones.

This terminology isn't always used. Sometimes combinations of

A	$\xrightarrow{\quad f \quad}$	B
source	arrow	target
domain	morphism	codomain
	map	

are used. Certainly morphisms (such as group morphisms) and maps (such as continuous maps) usually are examples of arrows in some category. However, it is better to use 'arrow' for the abstract notion, and so distinguish between the general and the particular.

The word 'domain' already has other meanings in mathematics. Why bother with this and 'codomain' when there are two perfectly good words that capture the idea quite neatly. You will also see

$$f : A \longrightarrow B$$

used to name the arrow above. However, as we see later, you should not think of an arrow as a function.

All three of the notations

$$A \xrightarrow{\quad \boldsymbol{id}_A \quad id_A \quad 1_A \quad} A$$

are used for the identity arrow assigned to the object A. We will tend to use id_A. Notice that the source and the target of id_A are both the parent object A. Quite often when there is not much danger of confusion id is written for id_A. You will also find in the literature that some people write 'A' for the arrow id_A. This is a notation so ridiculous that it should be laughed at in the street.

Certain pairs of arrows are compatible for composition to form another arrow. Two arrows

$$A \xrightarrow{\quad f \quad} B_1 \qquad\qquad B_2 \xrightarrow{\quad g \quad} C$$

are composible, in that order, precisely when B_1 and B_2 are the same object, and then an arrow

$$A \longrightarrow C$$

is formed. For arrows

$$A \xrightarrow{\quad f \quad} B \xrightarrow{\quad g \quad} C$$

both of the notations

$$A \xrightarrow{\quad g \circ f \quad gf \quad} C$$

are used for the composite arrow. Read this as

$$g \text{ after } f$$

and be careful with the order of composition. Here we write $g \circ f$ for the composite.

We need to understand how to manipulate composition, sometimes involving many arrows.

Composition of arrows is associative as far as it can be. For arrows

$$A \xrightarrow{f} B \xrightarrow{g} C \xrightarrow{h} D$$

various composites are possible, as follows.

$$\begin{array}{l}
A \xrightarrow{(h \circ g) \circ f} D \\
A \xrightarrow{f} B \xrightarrow{h \circ g} D \\
A \xrightarrow{f} B \xrightarrow{g} C \xrightarrow{h} D \\
A \xrightarrow{g \circ f} C \xrightarrow{h} D \\
A \xrightarrow[h \circ (g \circ f)]{} D
\end{array}$$

It is required that the two extreme arrows are equal

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

and we usually write

$$h \circ g \circ f$$

for this composite. This is the first of the axioms restricting the data.

The second axiom says that identity arrows are just that. Consider

$$A \xrightarrow{id_A} A \xrightarrow{f} B \xrightarrow{id_B} B$$

an arbitrary arrow and the two compatible identity arrows. Then

$$id_B \circ f = f = f \circ id_A$$

must hold.

Given two objects A and B in an arbitrary category $\boldsymbol{C}$, there may be no arrows from A to B, or there may be many. We write

$$\boldsymbol{C}[A, B] \quad \text{or} \quad \boldsymbol{C}(A, B)$$

for the collection of all such arrows. For historical reasons this is usually called the

hom-set

from A to B, although

arrow-class

would be better. Some people insist that $\boldsymbol{C}[A, B]$ should be a set, not a class. As usual, there are some variants of this notation. We often write

$$[A, B] \quad \text{for} \quad \boldsymbol{C}[A, B]$$

especially when it is clear which category $\boldsymbol{C}$ is intended. Sometimes

$$\mathrm{Hom}_{\boldsymbol{C}}[A, B]$$

is used for this hom-set.

We have seen above one very small diagram. Composition gives us a slightly larger one. Consider three arrows

$$\begin{tikzcd} & \bullet \arrow[dr, "g"] & \\ \bullet \arrow[ur, "f"] \arrow[rr, "h"'] & & \bullet \end{tikzcd}$$

arranged in a triangle, as shown. Here we haven't given each object a name, because we don't need to. However, the notation does *not* mean that the three objects are the same. For this small diagram, the triangle, the composite $g \circ f$ exists to give us a parallel pair

$$\begin{tikzcd} \bullet \arrow[r, shift left, "g \circ f"] \arrow[r, shift right, "h"'] & \bullet \end{tikzcd}$$

of arrows across the bottom of the triangle. These two arrows may or may not be the same. When they are

$$h = g \circ f$$

we say the triangle commutes. We look at some more commuting diagrams in Section 2.1, and other examples occur throughout the book.

Examples of categories

In the remaining sections of this chapter we look at a selection of examples of categories. Roughly speaking these are of four kinds.

The first collection is listed in Table 1.1 on page 6. These all have a similar nature and are examples of the most common kind of category we meet in practice. In each an object is a structured set, a set furnished, or equipped, with some extra gadgetry, the furnishings of the object. An arrow between two objects is a function between the carrying sets where the function 'respects' the carried structure. Arrow composition is then function composition. We look at some of these categories in Section 1.2.

Some categories listed in Table 1.1 are not defined in this chapter. Some are used later to illustrate various aspects of category theory, in which case each

Table 1.1 *Categories of structured sets and structure preserving functions*

Category	Objects	Arrows
Set	sets	total functions
Pfn	sets	partial functions
$\boldsymbol{Set}_{\perp}$	pointed sets	point preserving functions
RelH	sets with a relation	relation respecting functions
Sgp	semigroups	morphisms
Mon	monoids	morphisms
CMon	commutative monoids	morphisms
Grp	groups	morphisms
AGrp	abelian groups	morphisms
Rng	rings	morphisms
CRng	commutative rings	morphisms
Pre	pre-ordered sets	monotone maps
Pos	posets	monotone maps
Sup	complete posets	$\bigvee$-preserving monotone functions
Join	posets with all finitary joins	$\vee$-preserving monotone functions
Inf	complete posets	$\bigwedge$-preserving monotone functions
Meet	posets with all finitary meets	$\wedge$-preserving monotone functions
Top	topological spaces	continuous maps
$\boldsymbol{Top}_{\star}$	pointed topological spaces	point preserving continuous maps
$\boldsymbol{Top}^{\text{open}}$	topological spaces	continuous open maps
$\boldsymbol{Vect}_K$	vector spaces over a given field K	linear transformations
$\boldsymbol{Set}$-R	sets with a right action from a given monoid R	action preserving functions
R-$\boldsymbol{Set}$	sets with a left action from a given monoid R	action preserving functions
$\boldsymbol{Mod}$-R	right R-modules over a ring R	morphisms
R-$\boldsymbol{Mod}$	left R-modules over a ring R	morphisms

Table 1.2 *More complicated categories*

Category	Objects	Arrows
$\boldsymbol{RelA}$	sets	binary relations
$\boldsymbol{Pos}^{\dashv}$	posets	poset adjunctions
$\boldsymbol{Pos}^{pp}$	posets	projection embedding pairs
$\widehat{S}$	presheaves on a given poset S	natural transformations
$\widehat{\boldsymbol{C}}$	presheaves on a given category $\boldsymbol{C}$	natural transformations
$\boldsymbol{Ch}(\boldsymbol{Mod}\text{-}R)$	chain complexes	

is defined when it first appears. Some categories are listed but not used in this book, but you should be able to fill in the details when you need to.

These simple examples tend to give the impression that in any category an object is a structured set and an arrow is a function of a certain kind. This is a false impression, and in Section 1.3 we look at some examples to illustrate this. In particular, these examples show that an arrow need not be a function (of the kind you first thought of).

An important message of category theory is that the more important part of a category is not its objects but the way these are compared, its arrows. Given this we might expect a category to be named after its arrows. For historical reasons this often doesn't happen.

Section 1.4 contains some examples to show that the objects of a category can have a rather complicated internal structure, and the arrows are just as complicated. These examples are important in various parts of mathematics, but you shouldn't worry if you cannot understand them immediately.

Table 1.2 lists some of these more complicated examples looked at in Sections 1.3 and 1.4.

Finally in Section 1.5 we look at two very simple kinds of categories. These examples could be given now, but in some ways it is better if we leave them for a while.

Exercises

1.1.1 Observe that sets and functions do form a category $\boldsymbol{Set}$.

1.1.2 Can you see that each poset is a category, and each monoid is a category? Read that again.

1.2 Categories of structured sets

The categories we first meet usually have a rather simple nature. Each object is a structured set

$$(A, \cdots)$$

a set furnished with some extra gadgetry, its furnishings, and each arrow

$$(A, \cdots) \longrightarrow (B, \cdots)$$

is a (total) function

$$f : A \longrightarrow B$$

between the two carrying sets which respects the carried structure in some appropriate sense. More often than not these structured sets are 'algebras'. Thus the furnishings carried by A are a selection of nominated elements, and a selection of nominated operations on A. These operations are usually binary or singulary, but other arities do occur.

You have already met

$$\boldsymbol{Grp} \qquad \boldsymbol{Rng} \qquad \boldsymbol{Vect}_K$$

as given in Table 1.1, but you may not have realized that each of these is a category. You should make sure that you understand the workings of each of these as a category of 'algebras'. You may have to puzzle a bit over $\boldsymbol{Vect}_K$, but later we look at some more general examples of this nature, and that should help you.

To help with the general idea, in the first part of this section we look at the category $\boldsymbol{Mon}$ of monoids. This has all the typical properties of an 'algebraic' category. You may not have met monoids before, so this example will serve as an introduction, and it is quite easy to understand. Monoids are quite important in category theory. They can tell us quite a lot about the structure of a particular category. Also, they can be used to illustrate many aspects of category theory.

The exercises for the first part of this section look at several other categories of structured sets, some of which are not 'algebraic' in this intuitive sense. One of these

$$\boldsymbol{Top}$$

is particularly important, and you should make sure you understand it. It is important here and in many other parts of mathematics.

1.2.1 Example A monoid is a structure

$$(R,\ \star,\ 1)$$

where R is a set, $\star$ is a binary operation on R (usually written as an infix), 1 is a nominated element of R, and where

$$(r \star s) \star t = r \star (s \star t) \qquad 1 \star r = r = r \star 1$$

for all $r, s, t \in R$. In other words, the operation is associative and the nominated element is a unit for the operation. Monoids are sometimes referred to as unital semigroups, or even semigroups. However, sometimes a 'semigroup' need not have a unit.

Usually we omit the operation symbol and write

$$rs \quad \text{for} \quad r \star s$$

but for the time being we will stick to the official notation.

A monoid morphism

$$R \xrightarrow{\ \phi\ } S$$

between two monoids is a function that respects the furnishings, that is

$$\phi(r \star s) = \phi(r) \star \phi(s) \qquad \phi(1) = 1$$

for all $r, s \in R$. (Notice that we have overloaded the operation symbol and the unit symbol. That shouldn't cause a problem here, but every now and then it is a good idea to distinguish between the source and target furnishings.)

It is routine to check that for two morphisms

$$R \xrightarrow{\ \phi\ } S \xrightarrow{\ \psi\ } T$$

between monoids the function composite

$$R \xrightarrow{\ \psi \circ \phi\ } T$$

is a morphism.

This gives us the category ***Mon*** of monoids (as objects) and monoid morphisms (as arrows). The verification of the axioms is almost trivial. Given a monoid R the identity arrow

$$R \xrightarrow{\ id_R\ } R$$

is just the identity function on R viewed as a morphism. □

As suggested above, many categories fit into this 'algebraic' form. Each object is a structured set, and each arrow (usually called a morphism or a map) is a structure respecting function. Almost all of the categories in Table 1.1 fit into this kind, but one or two don't.

In a sense the study of monoids is the study of composition in the miniature. There is a corresponding study of comparison in the miniature. That is the topic of the next example.

1.2.2 Example A pre-order $\leq$ on a set S is a binary relation that is both reflexive and transitive. (Sometimes a pre-order is called a quasi-order.) A partial order is a pre-order that is also anti-symmetric.

A

preset poset

is a set S furnished with a

pre-order partial order

respectively. Thus each poset is a preset, but not conversely.

When comparing two such structures

$$(R, \leq_R) \qquad (S, \leq_S)$$

we use the carrying sets R and S to refer to the structures and write $\leq$ for both the carried comparisons. Rarely does this cause any confusion, but when it does we are a bit more careful with the notation.

Given a pair R, S of presets a monotone map

$$R \xrightarrow{f} S$$

is a function, as indicated, such that

$$x \leq y \Longrightarrow f(x) \leq f(y)$$

for all $x, y \in R$. Note that this condition is an implication, not an equivalence. It is routine to check that for two monotone maps

$$R \xrightarrow{f} S \xrightarrow{g} T$$

between presets the function composition $g \circ f$ is also monotone.

This gives us two categories

$$\boldsymbol{Pre} \qquad \boldsymbol{Pos}$$

where the objects are

presets posets

respectively, and in both cases the arrows are the monotone maps. Each identity arrow is the corresponding identity function viewed as a monotone map. □