

Lecture Notes in Category Theory

Paul Ossientis

December 19, 2019

Contents

1	Category	2
1.1	Small Category	2
1.2	Category	4
1.3	The Category of Sets	6
2	Functor	7
3	Natural Transformation	8
4	Adjunction	9
4.1	Definition	9

Chapter 1

Category

1.1 Small Category

Before we define a category in full generality, we shall focus our attention on the notion of *small category*. This notion is interesting to us because while it essentially describes the notion of *category* itself, it remains simple enough to be compared with various other algebraic structures. For example, consider the case of a monoid: a monoid is essentially a set M together with a binary relation \circ defined on M which is associative, and an element e of M which acts as an identity element for \circ . In short a monoid is a tuple (M, \circ, e) containing some data, and which satisfy certain properties. The same is true of a *small category*: it is also a tuple containing some data, and which satisfy certain properties:

Definition 1 We call small category any tuple $(\text{Ob}, \text{Arr}, \text{dom}, \text{cod}, \text{id}, \circ)$ with:

- (1) Ob is a set
- (2) Arr is a set
- (3) $\text{dom} : \text{Arr} \rightarrow \text{Ob}$ is a function
- (4) $\text{cod} : \text{Arr} \rightarrow \text{Ob}$ is a function
- (5) $\text{id} : \text{Ob} \rightarrow \text{Arr}$ is a function
- (6) $\circ : \text{Arr} \times \text{Arr} \rightarrow \text{Arr}$ is a partial function
- (7) $g \circ f$ is defined $\Leftrightarrow \text{cod}(f) = \text{dom}(g)$
- (8) $\text{cod}(f) = \text{dom}(g) \Rightarrow \text{dom}(g \circ f) = \text{dom}(f)$
- (9) $\text{cod}(f) = \text{dom}(g) \Rightarrow \text{cod}(g \circ f) = \text{cod}(g)$
- (10) $\text{cod}(f) = \text{dom}(g) \wedge \text{cod}(g) = \text{dom}(h) \Rightarrow (h \circ g) \circ f = h \circ (g \circ f)$
- (11) $\text{dom}(\text{id}(a)) = a = \text{cod}(\text{id}(a))$
- (12) $\text{dom}(f) = a \Rightarrow f \circ \text{id}(a) = f$
- (13) $\text{cod}(f) = a \Rightarrow \text{id}(a) \circ f = f$

where (7) – (13) hold for all $f, g, h \in \text{Arr}$ and $a \in \text{Ob}$:

So if $\mathcal{C} = (\text{Ob}, \text{Arr}, \text{dom}, \text{cod}, \text{id}, \circ)$ is a small category, we have two sets Ob and Arr together with some structure defined on these sets. This feels very much like a monoid, except that we have two sets instead of one and it all looks more complicated. The set Ob is called the *set of objects* of the small category \mathcal{C} and is denoted $\text{Ob } \mathcal{C}$, while the set Arr is called the *set of arrows* of the small category \mathcal{C} and is denoted $\text{Arr } \mathcal{C}$. An element $x \in \text{Ob } \mathcal{C}$ is called an *object* of \mathcal{C} , while an element $f \in \text{Arr } \mathcal{C}$ is called an *arrow* of \mathcal{C} .

As part of the structure defined on the small category \mathcal{C} , we have two functions $\text{dom} : \text{Arr} \rightarrow \text{Ob}$ and $\text{cod} : \text{Arr} \rightarrow \text{Ob}$. Hence, given an arrow f of the small category \mathcal{C} , we have two objects $\text{dom}(f)$ and $\text{cod}(f)$ of the small category \mathcal{C} . The object $\text{dom}(f)$ is called the *domain* of f . The object $\text{cod}(f)$ is called the *codomain* of f . Note that an arrow f of the small category \mathcal{C} is simply an element of the set $\text{Arr } \mathcal{C}$. So it is itself a set but it may not be a function. The words *domain* and *codomain* are therefore overloaded as we are using them in relation to a set f which is possibly not a function. Whenever f is an arrow of the small category \mathcal{C} and a, b are objects, it is common to use the notation $f : a \rightarrow b$ as a notational shortcut for the equations $\text{dom}(f) = a$ and $\text{cod}(f) = b$. Once again, it is important to guard against the possible confusion induced by the notation $f : a \rightarrow b$ which does not mean that f is function. It simply means that f is an arrow with domain a and codomain b in the small category \mathcal{C} .

One of the main ingredients of the structure defining a small category \mathcal{C} is the partial function $\circ : \text{Arr} \times \text{Arr} \rightarrow \text{Arr}$, called the *composition operator* in the small category \mathcal{C} . The domain of this partial function is made of all ordered pairs (g, f) of arrows in \mathcal{C} for which $\text{cod}(f) = \text{dom}(g)$. As already indicated in definition (1), we use the infix notation $g \circ f$ rather than $\circ(g, f)$ and the arrow $g \circ f$ is called the *composition* of g and f . Once again, we should remember that the notation $g \circ f$ does not mean that g or f are functions. They are simply arrows in the small category \mathcal{C} . One key property of the composition operator \circ is the associativity postulated by (10) of definition (1). Note that if $f : a \rightarrow b$ and $g : b \rightarrow c$, then from properties (8) and (9) of definition (1) we obtain $g \circ f : a \rightarrow c$. Furthermore, if $h : c \rightarrow d$ we have $h \circ g : b \rightarrow d$ and the arrows $(h \circ g) \circ f$ and $h \circ (g \circ f)$ are therefore well-defined arrows with domain a and codomain d . This shows that the expression involved in the associativity condition (10) of definition (1) is always meaningful, involving terms which are well-defined provided $g \circ f$ and $h \circ g$ are themselves well-defined, i.e. provided $\text{cod}(f) = \text{dom}(g)$ and $\text{cod}(g) = \text{dom}(h)$.

Finally, as part of the structure defining the small category \mathcal{C} , we have a function $\text{id} : \text{Ob} \rightarrow \text{Arr}$ called the *identity operator* on the small category \mathcal{C} . Hence, for every object a of \mathcal{C} we have an arrow $\text{id}(a)$ called the *identity at a* . Looking at property (11) of definition (1) we see that $\text{id}(a) : a \rightarrow a$. In other words, the arrow $\text{id}(a)$ has domain a and codomain a . Furthermore, looking at properties (12) and (13) of definition (1), for every arrow $f : a \rightarrow b$, the composition arrows $\text{id}(b) \circ f$ and $f \circ \text{id}(a)$ are well-defined and both equal to f .

1.2 Category

The notion of *small category* defined in definition (1) is similar to that of any other algebraic structure the reader may be familiar with. It can safely be encoded in set theory as a tuple (which is a set) containing data (which are other sets) which satisfies certain properties. In set theory, everything is a set. A small category \mathcal{C} is a set, its collection of objects $\text{Ob } \mathcal{C}$ is a set, its arrows $\text{Arr } \mathcal{C}$ form a set, the functions dom , cod , id and the composition operator \circ are all sets (functions are typically encoded as sets of ordered pairs).

Category theory falls outside of set theory. While the definition of a *category* we provide below is formally identical to that of a small category, the object we are defining can no longer be encoded in general as an object of set theory. For example, say we want to speak about the *universe of all sets* or the *universe of all monoids*. These *universes* which are known as *classes* cannot be represented as sets. They are not objects of set theory. Or say we are working within the formal framework of a proof assistant such as Coq, Agda or Lean. These tools are based on type theory and do not fall within the scope of set theory. When defining a *category*, we assume some form of meta-theoretic context, some form of logic, some way of reasoning about objects which may not be sets, where some meaning is attached to the words *tuple*, *collection*, *equality* and *map*. This may sound all very fuzzy, yet we cannot be more formal at this stage.

Definition 2 We call category any tuple $(\text{Ob}, \text{Arr}, \text{dom}, \text{cod}, \text{id}, \circ)$ such that:

- (1) Ob is a collection with equality
- (2) Arr is a collection with equality
- (3) $\text{dom} : \text{Arr} \rightarrow \text{Ob}$ is a map
- (4) $\text{cod} : \text{Arr} \rightarrow \text{Ob}$ is a map
- (5) $\text{id} : \text{Ob} \rightarrow \text{Arr}$ is a map
- (6) $\circ : \text{Arr} \times \text{Arr} \rightarrow \text{Arr}$ is a partial map
- (7) $g \circ f$ is defined $\Leftrightarrow \text{cod}(f) = \text{dom}(g)$
- (8) $\text{cod}(f) = \text{dom}(g) \Rightarrow \text{dom}(g \circ f) = \text{dom}(f)$
- (9) $\text{cod}(f) = \text{dom}(g) \Rightarrow \text{cod}(g \circ f) = \text{cod}(g)$
- (10) $\text{cod}(f) = \text{dom}(g) \wedge \text{cod}(g) = \text{dom}(h) \Rightarrow (h \circ g) \circ f = h \circ (g \circ f)$
- (11) $\text{dom}(\text{id}(a)) = a = \text{cod}(\text{id}(a))$
- (12) $\text{dom}(f) = a \Rightarrow f \circ \text{id}(a) = f$
- (13) $\text{cod}(f) = a \Rightarrow \text{id}(a) \circ f = f$

where (7) – (13) hold for all $f, g, h \in \text{Arr}$ and $a \in \text{Ob}$:

So let $\mathcal{C} = (\text{Ob}, \text{Arr}, \text{dom}, \text{cod}, \text{id}, \circ)$ be a category: then \mathcal{C} is a *tuple* but it is no longer a tuple in a set-theoretic sense. We assume given some logical framework where the notion of *tuple* is clear, even if not formally defined. Furthermore, We are no longer imposing that Ob should be a set, but are instead using the phrase *collection with equality*, whatever this may mean in our given logical context. So we shall still make use of the notation $\text{Ob } \mathcal{C}$ but this will now refer to the *collection* of all *objects* of the category \mathcal{C} . In fact, if a is an object of the category \mathcal{C} , we shall abuse notations somewhat by writing ' $a \in \text{Ob } \mathcal{C}$ ' or even simply ' $a \in \mathcal{C}$ ' to express the fact that a is an object of \mathcal{C} , being understood that this use of the set membership symbol ' \in ' does not mean anything is a set. Since we are stepping out of set theory, the objects of the category \mathcal{C} may not be sets themselves. They are simply members of the *collection* $\text{Ob } \mathcal{C}$. However, properties (7) – (13) of definition (2) are all referring to equalities between objects such that $\text{cod}(f) = \text{dom}(g)$. So it must be the case that the notion of *equality* be meaningful on the collection $\text{Ob } \mathcal{C}$. This explains our use of the phrase *collection with equality*: given $a, b \in \mathcal{C}$, the statement $a = b$ while not a set-theoretic equality is nonetheless assumed to be defined.

Similarly, the *collection* of *arrows* of the category \mathcal{C} shall still be denoted $\text{Arr } \mathcal{C}$, but is no longer required to be a set. If f is an arrow of the category \mathcal{C} then f itself may not be a set and we may still write ' $f \in \text{Arr } \mathcal{C}$ ' simply to indicate that f is a *member* of the *collection* $\text{Arr } \mathcal{C}$. Properties (10), (12) and (13) of definition (2) are all referring to equalities between arrows so the *collection* $\text{Arr } \mathcal{C}$ must have some notion of *equality* defined on it.

Since Ob and Arr are no longer sets in general, the *maps* $\text{dom} : \text{Arr} \rightarrow \text{Ob}$, $\text{cod} : \text{Arr} \rightarrow \text{Ob}$, $\text{id} : \text{Ob} \rightarrow \text{Arr}$ and the partial map $\circ : \text{Arr} \times \text{Arr} \rightarrow \text{Arr}$ cannot possibly be *functions* in the set-theoretic sense. So there must be some meaning to the word *map* (from one *collection* to another) in whatever logical framework we are working in. The *collection* $\text{Arr} \times \text{Arr}$ is not a set, and is simply the *collection* of all 2-dimensional *tuples* made from Arr . Our using the word *map* rather than *function* in definition (2) is simply an attempt at reminding ourselves of the fact these are not set-theoretic functions, even though the words *map* and *function* are perfectly interchangeable in standard (set-theoretic) mathematics.

Given $f \in \text{Arr } \mathcal{C}$, we shall still call the object $\text{dom}(f)$ the *domain* of f and the object $\text{cod}(f)$ the *codomain* of f . Given $a, b \in \mathcal{C}$, we shall still use the notation $f : a \rightarrow b$ as a notational shortcut for $\text{dom}(f) = a$ and $\text{cod}(f) = b$. The partial map \circ is still the *composition operator* and the arrow $g \circ f$ shall still be called the *composition* of g and f , provided it is defined. The map $\text{id} : \text{Ob} \rightarrow \text{Arr}$ is still the *identity operator* on the category \mathcal{C} , and for all $a \in \mathcal{C}$, the arrow $\text{id}(a) : a \rightarrow a$ is known as the *identity at a*. For all arrows $f : a \rightarrow b$, it is still the case that the arrows $\text{id}(b) \circ f$ and $f \circ \text{id}(a)$ are well-defined and both equal to f . Just as in the case of a small category, whenever $f : a \rightarrow b$, $g : b \rightarrow c$ and $h : c \rightarrow d$, all the terms involved in the associativity condition $(h \circ g) \circ f = h \circ (g \circ f)$ of definition (2) are well defined.

1.3 The Category of Sets

Definition 3 We call **Set** the category $\mathbf{Set} = (\mathbf{Ob}, \mathbf{Arr}, \text{dom}, \text{cod}, \circ)$ where

- (1) $\mathbf{Ob} = \{ x \mid x \text{ is a set} \}$
- (2) $\mathbf{Arr} = \{ (a, b, f) \mid f \text{ is a function } f : a \rightarrow b \}$
- (3) $\text{dom}(a, b, f) = a$
- (4) $\text{cod}(a, b, f) = b$
- (5) $(b, c, g) \circ (a, b, f) = (a, c, g \circ f)$

where (3), (4), (5) hold for all sets a, b, c and functions $f : a \rightarrow b$, $g : b \rightarrow c$, and $g \circ f$ denotes the usual function composition with $(g \circ f)(x) = g(f(x))$.

The collection of objects of the category **Set** is defined to be the class of all sets. We are using the set comprehension notation $\{ x \mid x \text{ is a set} \}$ to denote this class, but this is an abuse of notation as \mathbf{Ob} is not a set but a proper class. One could think of a class as a predicate $P(x)$ of first order logic with one free variable. From this point of view \mathbf{Ob} becomes the predicate $\mathbf{Ob}(x) = \top$, i.e. the predicate which returns true for all x . Every set satisfies the predicate \mathbf{Ob} , so every set is a member of the class \mathbf{Ob} . The class \mathbf{Ob} is not a set because the set-theoretic statement $\exists y, \forall z, z \in y \Leftrightarrow \mathbf{Ob}(z)$ can be proven false. In other words, there exists no set y whose elements z are exactly the sets which satisfy the predicate \mathbf{Ob} . There exists no set y which contains all sets.

Chapter 2

Functor

Chapter 3

Natural Transformation

Chapter 4

Adjunction

4.1 Definition

Definition 4 We call adjunction an ordered pair (F, G) where F is a functor $F : \mathcal{C} \rightarrow \mathcal{D}$ and G is a functor $G : \mathcal{D} \rightarrow \mathcal{C}$ while \mathcal{C} and \mathcal{D} are two locally-small categories for which there exists a natural isomorphism:

$$\alpha : \mathcal{D} \circ (F \times I_{\mathcal{D}}) \Rightarrow \mathcal{C} \circ (I_{\mathcal{C}^{op}} \times G)$$

in the functor category $[\mathcal{C}^{op} \times \mathcal{D}, \mathbf{Set}]$, where F also denotes $F : \mathcal{C}^{op} \rightarrow \mathcal{D}^{op}$.