Lecture Notes in Category Theory

Paul Ossientis

December 23, 2019

Contents

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* $(\mathrm{Ob}, \mathrm{Arr}, \mathrm{dom}, \mathrm{cod}, \mathrm{id}, \circ)$ *with:*

(1) Ob *is a set*

(2) Arr *is a set*

(3) $\mathrm{dom} : \mathrm{Arr} \to \mathrm{Ob}$ *is a function*

(4) $\mathrm{cod} : \mathrm{Arr} \to \mathrm{Ob}$ *is a function*

(5) $\mathrm{id} : \mathrm{Ob} \to \mathrm{Arr}$ *is a function*

(6) $\circ : \mathrm{Arr} \times \mathrm{Arr} \to \mathrm{Arr}$ *is a partial function*

(7) $g \circ f$ *is defined* $\Leftrightarrow \ \mathrm{cod}(f) = \mathrm{dom}(g)$

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

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

(10) $\mathrm{cod}(f) = \mathrm{dom}(g) \wedge \mathrm{cod}(g) = \mathrm{dom}(h) \ \Rightarrow \ (h \circ g) \circ f = h \circ (g \circ f)$

(11) $\mathrm{dom}\,(\,\mathrm{id}(a)\,) = a = \mathrm{cod}\,(\,\mathrm{id}(a)\,)$

(12) $\mathrm{dom}(f) = a \ \Rightarrow \ f \circ \mathrm{id}(a) = f$

(13) $\mathrm{cod}(f) = a \ \Rightarrow \ \mathrm{id}(a) \circ f = f$

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

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 $\text{dom} : \text{Arr} \to \text{Ob}$ and $\text{cod} : \text{Arr} \to \text{Ob}$. 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 objets, it is common to use the notation $f : a \to b$ 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 $f : a \to b$ 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 $\circ : \text{Arr} \times \text{Arr} \to \text{Arr}$, 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 $f : a \to b$ and $g : b \to c$, then from properties (8) and (9) of definition (1) we obtain $g \circ f : a \to c$. Furthermore, if $h : c \to d$ we have $h \circ g : b \to d$ 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 $\text{id} : \text{Ob} \to \text{Arr}$ 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 $\text{id}(a) : a \to a$. 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 $f : a \to b$, 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 $\mathrm{Ob}\ \mathcal{C}$ is a set, its arrows $\mathrm{Arr}\ \mathcal{C}$ 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* $(\mathrm{Ob}, \mathrm{Arr}, \mathrm{dom}, \mathrm{cod}, \mathrm{id}, \circ)$ *such that:*

(1) Ob *is a collection with equality*

(2) Arr *is a collection with equality*

(3) $\mathrm{dom} : \mathrm{Arr} \to \mathrm{Ob}$ *is a map*

(4) $\mathrm{cod} : \mathrm{Arr} \to \mathrm{Ob}$ *is a map*

(5) $\mathrm{id} : \mathrm{Ob} \to \mathrm{Arr}$ *is a map*

(6) $\circ : \mathrm{Arr} \times \mathrm{Arr} \to \mathrm{Arr}$ *is a partial map*

(7) $g \circ f$ *is defined* $\Leftrightarrow\ \mathrm{cod}(f) = \mathrm{dom}(g)$

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

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

(10) $\mathrm{cod}(f) = \mathrm{dom}(g) \wedge \mathrm{cod}(g) = \mathrm{dom}(h)\ \Rightarrow\ (h \circ g) \circ f = h \circ (g \circ f)$

(11) $\mathrm{dom}\,(\,\mathrm{id}(a)\,) = a = \mathrm{cod}\,(\,\mathrm{id}(a)\,)$

(12) $\mathrm{dom}(f) = a\ \Rightarrow\ f \circ \mathrm{id}(a) = f$

(13) $\mathrm{cod}(f) = a\ \Rightarrow\ \mathrm{id}(a) \circ f = f$

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

So let $\mathcal{C} = (\mathrm{Ob}, \mathrm{Arr}, \mathrm{dom}, \mathrm{cod}, \mathrm{id}, \circ)$ 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 $\mathrm{Ob}\ \mathcal{C}$ 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 '$a \in \mathrm{Ob}\ \mathcal{C}$' 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* $\mathrm{Ob}\ \mathcal{C}$. However, properties $(7) - (13)$ of definition (2) are all referring to equalities between objects such that $\mathrm{cod}(f) = \mathrm{dom}(g)$. So it must be the case that the notion of *equality* be meaningful on the collection $\mathrm{Ob}\ \mathcal{C}$. 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 $\mathrm{Arr}\ \mathcal{C}$, 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 '$f \in \mathrm{Arr}\ \mathcal{C}$' simply to indicate that f is a *member* of the *collection* $\mathrm{Arr}\ \mathcal{C}$. Properties (10), (12) and (13) of definition (2) are all referring to equalities between arrows so the *collection* $\mathrm{Arr}\ \mathcal{C}$ must have some notion of *equality* defined on it.

Since Ob and Arr are no longer sets in general, the *maps* $\mathrm{dom} : \mathrm{Arr} \to \mathrm{Ob}$, $\mathrm{cod} : \mathrm{Arr} \to \mathrm{Ob}$, $\mathrm{id} : \mathrm{Ob} \to \mathrm{Arr}$ and the partial map $\circ : \mathrm{Arr} \times \mathrm{Arr} \to \mathrm{Arr}$ 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* $\mathrm{Arr} \times \mathrm{Arr}$ 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, eventhough the words *map* and *function* are perfectly interchangeable in standard (set-theoretic) mathematics.

Given $f \in \mathrm{Arr}\ \mathcal{C}$, we shall still call the object $\mathrm{dom}(f)$ the *domain* of f and the object $\mathrm{cod}(f)$ the *codomain* of f. Given $a, b \in \mathcal{C}$, we shall still use the notation $f : a \to b$ as a notational shortcut for $\mathrm{dom}(f) = a$ and $\mathrm{cod}(f) = b$. 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 $\mathrm{id} : \mathrm{Ob} \to \mathrm{Arr}$ is still the *identity operator* on the category $\mathcal{C}$, and for all $a \in \mathcal{C}$, the arrow $id(a) : a \to a$ is known as the *identity at* a. For all arrows $f : a \to b$, it is still the case that the arrows $\mathrm{id}(b) \circ f$ and $f \circ \mathrm{id}(a)$ are well-defined and both equal to f. Just as in the case of a small category, whenever $f : a \to b$, $g : b \to c$ and $h : c \to d$, 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* $\mathbf{Set} = (\mathrm{Ob}, \mathrm{Arr}, \mathrm{dom}, \mathrm{cod}, \mathrm{id}, \circ)$ *where*

$$
\begin{aligned}
&(1) && \mathrm{Ob} = \{\ x \mid x \textit{ is a set}\ \} \\
&(2) && \mathrm{Arr} = \{\ (a,b,f) \mid f \textit{ is a function } f : a \to b\ \} \\
&(3) && \mathrm{dom}\,(a,b,f) = a \\
&(4) && \mathrm{cod}\,(a,b,f) = b \\
&(5) && \mathrm{id}(a) = (a,a,i(a)) \\
&(6) && (b,c,g) \circ (a,b,f) = (a,c,g \circ f)
\end{aligned}
$$

where $(3) - (6)$ *hold for all sets* a, b, c *and functions* $f : a \to b$, $g : b \to c$, $i(a) : a \to a$ *denotes the usual identity function on* a, *and* $g \circ f$ *denotes the usual function composition defined by* $(g \circ f)(x) = g(f(x))$, *for all* $x \in a$.

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

The collection of arrows of the category **Set** is defined to be the class of triples (a, b, f) where a, b are sets and f is a function $f : a \to b$. This last notation is a common set-theoretic shortcut to express that fact that f is a *function* with *domain* a and *range* **which is a subset of** b. A *function* is any set f whose elements are ordered pairs (x, y) and which is functional, i.e. for which the following implication holds for all sets x, y, y':

$$(x,y) \in f \ \wedge\ (x,y') \in f \ \Rightarrow\ y = y'$$

The *domain* of a function f is the set of all sets x for which there exists a set y with $(x, y) \in f$. The *range* of a function f is the set of all sets y for which there exists a set x with $(x, y) \in f$. If x belongs to the domain of a function f, the notation '$f(x)$' commonly refers to the unique set y with $(x, y) \in f$.

Now, as already pointed out the notation $f : a \to b$ only requires that the range of f should be a subset of b. There is no requirement that the range of f should be equal to b. So if $f : a \to b$ and $b \subseteq c$ then $f : a \to c$. This explains why the collection of arrows Arr is defined as a class of triples (a, b, f) rather than a class of functions f. Knowing the function f does not tell you which *codomain* it should have. Any set b which is a superset of its range is a possible codomain. So we keep the set b together with the function f in the triple (a, b, f) so as to remember which codomain is intended for this particular arrow of the

category **Set**. Incidentally, we also keep the range a of the function f in the triple (a, b, f) but this is not necessary, as the knowledge of f does allow us to recover its domain a. However, the triple (a, b, f) is convenient, allowing us to treat *domain* and *codomain* uniformly. Once again, it should be remembered that the collection of arrows Arr is not a set but a proper class, corresponding to the predicate $\mathrm{Arr}(x)$ defined as follows:

$$\mathrm{Arr}(x) = \exists a\, \exists b\, \exists f\,,\ x = (a, b, f)\ \wedge\ f : a \to b$$

The maps $\mathrm{dom} : \mathrm{Arr} \to \mathrm{Ob}$ and $\mathrm{cod} : \mathrm{Arr} \to \mathrm{Ob}$ for the category **Set** are defined respectively by $\mathrm{dom}(a, b, f) = a$ and $\mathrm{cod}(a, b, f) = b$. This looks simple enough, but for those who worry about foundational issues, we should just note that these are also proper classes which can be encoded as predicates. For example:

$$\mathrm{dom}(x) = \exists u\, \exists v\,,\ x = (u, v)\ \wedge\ \mathrm{Arr}(u)\ \wedge\ (\ \exists a\, \exists b\, \exists f\,,\ u = (a, b, f)\ \wedge\ v = a\ )$$

In other words, any set x satisfies the predicate $\mathrm{dom}(x)$ if and only if it is an ordered pair (u, v) where u satisfies the predicate $\mathrm{Arr}(u)$ and for which there exist sets a, b, f with $u = (a, b, f)$ and $v = a$. In short, (u, v) satisfies the predicate dom if and only if u is an arrow $u = (a, b, f)$ and $v = a$.

We defined the identity operator id by $\mathrm{id}(a) = (a, a, i(a))$ and the composition operator $\circ$ by $(b, c, g) \circ (a, b, f) = (a, c, g \circ f)$ where $g \circ f$ is the usual function composition and $i(a) : a \to a$ is the usual identity function. As before, these defined maps are not functional sets of ordered pairs but rather proper classes which we could also encode as precicates of first order logic. One important point to note is the fact that (6) of definition (3) only defines the composition arrow $(b, c, g) \circ (a, b, f)$ where $f : a \to b$ and $g : b \to c$. In other words, the composition $(d, c, g) \circ (a, b, f)$ with $f : a \to b$ and $g : d \to c$ is only defined when $b = d$. Furthermore, the usual function composition $g \circ f$ is a function $g \circ f : a \to c$ which from (2) of definition (3) means that the composed arrow $(b, c, g) \circ (a, b, f) = (a, c, g \circ f)$ is indeed a member of the collection Arr, and the partial map $\circ$ thus defined is indeed a partial map $\circ : \mathrm{Arr} \times \mathrm{Arr} \to \mathrm{Arr}$.

Lemma 1 *The category* **Set** *of definition (3) is a category.*

Proof
Now that we have defined the data $(\mathrm{Ob}, \mathrm{Arr}, \mathrm{dom}, \mathrm{cod}, \mathrm{id}, \circ)$ of the category **Set**, it is time to check this data actually forms a category. We need to check that conditions $(7) - (13)$ of definition (2) are satisfied.

(7): suppose f^* and g^* are two members of the collection Arr. We need to check that $g^* \circ f^*$ is defined if and only if $\mathrm{cod}(f^*) = \mathrm{dom}(g^*)$. Using (2) of definition (3), f^* can be written $f^* = (a, b, f)$ for some function $f : a \to b$ and g^* can be written $g^* = (d, c, g)$ for some function $g : d \to c$. However, from (6) of definition (3), the arrrow $(d, c, g) \circ (a, b, f)$ is only defined in the case when $b = d$. Furthermore, from (4) of definition (3) we have $\mathrm{cod}(f^*) = b$ and from (3) of definition (3) we have $\mathrm{dom}(g^*) = d$. We conclude that $g^* \circ f^*$ is defined if and only if $\mathrm{cod}(f^*) = \mathrm{dom}(g^*)$ as required.

(8): Let $f^*, g^* \in \mathrm{Arr}$ such that $\mathrm{cod}(f^*) = \mathrm{dom}(g^*)$. We need to show that $\mathrm{dom}(g^* \circ f^*) = \mathrm{dom}(f^*)$. **.**

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* $F : \mathcal{C} \to \mathcal{D}$ *and* G *is a functor* $G : \mathcal{D} \to \mathcal{C}$ *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* $F : \mathcal{C}^{op} \to \mathcal{D}^{op}$*.*