Categories for the lazy functional programmer
Lecture notes

Thorsten Altenkirch

April 7, 2024

Abstract

The course is an introduction to category theory emphasising applications in computer science, especially functional programming, from a type theoretic perspective. We cover the basic concepts of category theory: categories, duality, functors and natural transformations, adjunctions, the Yoneda lemma, products, coproducts and exponentials, initial algebras and terminal coagebras, limits and colimits.

Contents

1 Preliminaries

We will use naive type theory as a metalanguage, this is not very different from a disciplined use of set theory. It is not necessary to have any knowledge about the formalism of type theory in as much as it is not necessary to know the axioms of set theory when using set theory naively.

However, a few notes are in place. We write $a : A$ to mean that a is an element of the type A unlike $a \in A$ in set theory this is a judgement not a proposition, i.e. it is a static property.

We write **Set** for the type of sets and **Prop** for the type of truth values, which we do not assume to be equal to Bool. We have an embedding from **Prop** to **Set** which is justified by the proposition as types translation: a proposition corresponds to a set with at most one inhabitant, it is true if the type is non-empty. For every $A : \mathbf{Set}$ we have an equality relation, that is for $a, b : A$ we have $a = b : \mathbf{Prop}$, expressing that a and b are equal. **Set** is a type but not every type is a set, and we avoid talking about equality of elements of types that are not sets[1]. On the other hand $\mathbf{Prop} : \mathbf{Set}$ and equality of propositions is logical equivalence.

We will assume that there is a hierarchy of of types, eg. $\mathbf{Type}_0 : \mathbf{Type}_1 : \dots$ and correspondingly $\mathbf{Set}_i, \mathbf{Prop}_i : \mathbf{Type}_{i+1}$ and $\mathbf{Prop}_i : \mathbf{Set}_{i+1}$. [2] We usually assume implicitly that we are working for some fixed level i and write $\mathbf{Type}, \mathbf{Set}, \mathbf{Prop}$ for $\mathbf{Type}_i, \mathbf{Set}_i, \mathbf{Prop}_i$. We say types (sets, propositions) are small if they are in $\mathbf{Set}_i$ and large if they are in $\mathbf{Set}_{i+1}$.

We also use function types (sets, propositions) $A \to B$ extensively, we view them as primitive and not defined as relations. Given $f : A \to B$ and $a : A$ we write application as juxtaposition $f\,a : B$ as usual in functional programming and type theory. We assume functional extensionality: two functions are equal, if they are pointwise equal.

We borrow Agda's convention to write mixfix operations by using _ where the arguments should go, e.g. a binary infix operator is written $\_+\_ : \mathbb{N} \to \mathbb{N} \to \mathbb{N}$. There are some notations I borrow from set theory, e.g. I am going to write finite sets as $\{c_0, c_1, \dots, c_n\}$ where c_i are some names for the constructors. I also use comprehension notation $\{x : A \mid P\,x\}$ where A is a type and $P : A \to \mathbf{Prop}$,

[1] That is I adopt an agnostic view you can view types as Zermelo-Fraenkel sets or as types in the sense of HoTT

[2] Some people like to assume that $\mathbf{Prop}_i : \mathbf{Set}_0$ which is a consequence of $\mathbf{Prop}_i = \mathrm{Bool}$.

type theoretically this can be interpreted as the type of pairs (a, p) where $a : A$ and $p : P\,a$.

2 Categories

2.1 From sets to categories

We start with the standard example of a category: the category of sets $\underline{\mathbf{Set}}$. To define a category we need a type of objects which is $\mathbf{Set}$. We write $|\underline{\mathbf{Set}}| = \mathbf{Set}$. Given two objects $A, B : \mathbf{Set}$ we define the set of morphisms which in this case is the set of functions, $\underline{\mathbf{Set}}(A, B) = A \to B$. For every object $A : \mathbf{Set}$ we have an identity function $\mathrm{id}_A : A \to A$ which is defined as $\mathrm{id}_A\,a = a$, given functions $f : B \to C$ and $g : A \to B$ we define function composition $f \circ g : A \to C$ as $(f \circ g)\,a = f\,(g\,a)$. The strange order of arguments in composition is a consequence of the fact that we write function application this way around, hence there is no change of direction in the definition of $\circ$. We observe that there are a number of laws for function composition: identity is neutral, that is given $f : A \to B$ we have that $\mathrm{id}_B \circ f = f$ and $f \circ \mathrm{id}_A = f$ and moreover it is associative, that is given three composable function $f : C \to D, g : B \to C, h : A \to B$ we have that $f \circ (g \circ h) = (f \circ g) \circ h$.

Exercise 1 *Write down the proofs that identity is neutral and composition is associative in detail.*

We abstract from this example to define what is a category: A category $\underline{\mathbf{C}}$ is given by a type of objects $|\underline{\mathbf{C}}|$ and given two objects $A, B : |\underline{\mathbf{C}}|$, a set of morphisms $\underline{\mathbf{C}}(A, B) : \mathbf{Set}$ called the *homset*. For every object $A : |\underline{\mathbf{C}}|$ we have an identity function[3] $\mathrm{id}_A : \underline{\mathbf{C}}(A, A)$, given functions $f : \underline{\mathbf{C}}(B, C)$ and $g : \underline{\mathbf{C}}(A, B)$ there is a composite $f \circ g : \underline{\mathbf{C}}(A, C)$ satisfying the following laws: identity is neutral, that is given $f : \underline{\mathbf{C}}(A, B)$ we have that $\mathrm{id}_B \circ f = f$ and $f \circ \mathrm{id}_A = f$ and moreover it is associative, that is given three composable functions $f : \underline{\mathbf{C}}(A, B), g : \underline{\mathbf{C}}(B, C), h : \underline{\mathbf{C}}(C, D)$ we have that $h \circ (g \circ h) = (h \circ g) \circ f$.

Equations in category theory are often shown as *commuting diagrams*. Composition corresponds to completing a triangle

$$\begin{tikzcd}
A \arrow[r, "f"] \arrow[dr, dashed, "f \circ g"'] & B \arrow[d, "g"] \\
 & C
\end{tikzcd}$$

I am using a dashed arrow to indicate that this is the unique arrow that makes this diagram commute. The identity laws can be drawn as follows where I use

[3] We are overloading id and $\circ$ to be precise we should annotate these symbols with the category, i.e. write $\mathrm{id}^{\underline{\mathbf{C}}}$ and $\circ^{\underline{\mathbf{C}}}$. However, it is usually clear from the context which category is meant.

a double line to indicate an identity arrow:

$$\begin{tikzcd}
A \arrow[r, "f"] \arrow[dr, "f"'] & B \arrow[d, equal] & & A \arrow[r, equal] \arrow[dr, "f"'] & A \arrow[d, "f"] \\
& B & & & B
\end{tikzcd}$$

The associativity law can be drawn as below, the arrow on the right makes both the big triangle and the small triangle commute corresponding to the two sides of the associativity law:

$$\begin{tikzcd}
& D & \\
& C \arrow[u, "h"] & \\
A \arrow[ur, "g \circ f"] \arrow[rr, "f"] \arrow[uur, dashed, "(h \circ g) \circ f", "h \circ (g \circ f)"'] & & B \arrow[ul, "g"'] \arrow[uul, "h \circ g"']
\end{tikzcd}$$

A good intuition for a commuting diagram is to consider that the inside of the diagram is filled so that a path on the one side can be continuously transformed into a path on the other side. In the diagram for associativity we use the rule that we can also complete tetraeders, hence the bottom side of the tetraeder commutes and hence because composition is unique, associativity holds.

The sort of geometric, or rather homotopic, thinking suggest a clear path to higher categories, but this is beyond the scope of this course.

2.2 Terminal and initial objects

Here is an example how we can use the language of category theory to define concepts in the category of sets. We are interested in the concept of a set with exactly one element, but there are lots of choices for this element: some people write () and in Agda it is called `tt` other people write $*$. In category theory we avoid this confusion and say that a *terminal object* $\mathbf{1}$ is an object such that there is exactly one morphisms from any other object, let's say

$$\begin{tikzcd}
A \arrow[r, dashed, "!_A"] & \mathbf{1}
\end{tikzcd}$$

Clearly one element sets are terminal objects in the category of sets. Now terminal objects are unique *up to isomorphism.*

We say that two objects A, B are *isomorphic* if there are morphisms $f : A \to B$ and $g : B \to A$ such that there compositions are identities: $f \circ g = \mathrm{id}_B$ and $g \circ f = \mathrm{id}_A$.

$$\begin{tikzcd}
A \arrow[loop left, "\mathrm{id}_A"] \arrow[r, bend left, "f"] & B \arrow[l, bend left, "g"] \arrow[loop right, "\mathrm{id}_B"]
\end{tikzcd}$$

We call f and g *isomorphisms.* We write $f : A \cong B$ or omitting the witness we write $A \cong B$ to express that A and B are isomorphic.

Exercise 2 *Show that inverses are unique, i.e. if (f, g) and (f, g') are an isomorphisms then $g = g'$.*

Now given two terminal objects $\mathbf{1}, \mathbf{1}'$ we have $!'_{\mathbf{1}} : \mathbf{1} \to \mathbf{1}'$ and $!_{\mathbf{1}'} : \mathbf{1} \to \mathbf{1}'$. There compositions are the identity because for example $!_{\mathbf{1}'} \circ !_{\mathbf{1}'} : \mathbf{1} \to \mathbf{1}$ but there is also the identity $\mathrm{id}_{\mathbf{1}} : \mathbf{1} \to \mathbf{1}$ and since we said that there is exactly one morphism from any object to the terminal object it must be that $!_{\mathbf{1}'} \circ !_{\mathbf{1}'} = !_{\mathbf{1}} = \mathrm{id}_{\mathbf{1}}$. The same reasoning shows that the other composition is also the identity.

In category theory we are only interested in concepts *up to isomorphism* and all constructions are preserved by isomorphism. In the category of sets isomorphic objects have the same elements up to a renaming given by the isomorphism. Hence we ignore the irrelevant detail how the elements are called.

There is also a dual concept: the empty set is an *initial object* $\mathbf{0}$, that is there is a unique morphism from the empty set into any set,

$$\begin{tikzcd} \mathbf{0} \arrow[r, dashed, "?_A"] & A \end{tikzcd}$$

this exists trivially because we don't have to say what the functions returns since there are no elements in its domain. In this case we are lucky there is only one way to write the initial object. We can make the symmetry precise by introducing the concept of a dual category: given a category $\underline{\mathbf{C}}$ the opposite category $\underline{\mathbf{C}}^{\mathrm{op}}$ has the same objects as $\underline{\mathbf{C}}$ but the homsets are given by reversing the order $\underline{\mathbf{C}}^{\mathrm{op}}(A, B) = \underline{\mathbf{C}}(B, A)$ and consequently composition also reverses order $f \circ^{\underline{\mathbf{C}}^{\mathrm{op}}} g = g \circ^{\underline{\mathbf{C}}} f$. We can now say that an initial object is a terminal object in the opposite category, or vice versa a terminal object is an initial object in the opposite category. We use the syllable *co-* in this situation, we could say that an initial object is a co-terminal object or that a terminal object is a co-initial object (but nobody says this).

Exercise 3 *Prove explicitly that initial objects are unique upto isomorphism.*

When I say that a category has a terminal or initial object this also indicates that I assume that we have chosen one which I indicate by 1 or 0 respectively. The same principle applies to all categorical constructs we are going to introduce. We will later discuss the concept of univalent categories which entails that categorical constructions are unique.

2.3 Preorders and monoids

Let us look at some other examples of categories. We can use the natural numbers to define some categories: The category $\underline{\omega}$ has as objects the natural numbers and morphisms are given as $\underline{\omega}(m, n) = m \leq n$. This is a category because $\leq$ is reflexive and transitive and those properties give rise to identity and composition. The laws hold trivial because there is at most one element of a proposition. This category is an example of a degenerate case of a category: it is a *preorder*, i.e. a relation that is reflexive and transitive.

Question 4 *Has this category an initial or a terminal object?*

Another category we can be build from natural numbers has only one object and its morphisms are the natural numbers. The identity morphism is 0 and composition is given by addition. The laws follow from the fact that 0 is neutral $(n + 0 = n = 0 + n)$ and associative $(l + m) + n = l + (m + n)$. Here we exploit the algebraic properties of $+$, namely that it is a *monoid*; hence categories with one object correspond to monoids.

Question 5 *Has this category an initial or a terminal object?*

In Mathematics we are also interested in equivalence relations (preorders that are symmetric) and groups (monoids that have a inverses), like the integers $\mathbb{Z}$ with addition on negation). We can also do this for categories by saying that a category is a *groupoid*, if for every morphism $f : A \to B$ there is an inverse $f^{-1} : B \to A$ such that $f \circ f^{-1} = \mathrm{id}$ and $f^{-1} \circ f = \mathrm{id}$. A groupoid whose homsets are propositions is an equivalence relation and a groupoid with one object is a group.

Exercise 6 *Show that sets with isomorphisms are a groupoid. Here the objects are sets and the homsets are isomorphisms. First you need to show that this is indeed a category.*

A rich source of categories are sets with structure. For example we define the category of preorders $\underline{\mathbf{Pre}}$: Its objects are preorders that is a set A together with a relation $R : A \to A \to \mathbf{Prop}$ which is reflexive and transitive. Given preorders (A, R) and (B, S) a morphism is a function $f : A \to B$ which preserves the relation that is if $R\,a\,a'$ then $S\,(f\,a)\,(f\,a')$. We use identity and composition as in $\underline{\mathbf{Set}}$, we need to check that identity is a preorder morphism and that the composition of preorder morphisms is a preorder morphism.

Another example is the category of monoids $\underline{\mathbf{Mon}}$: Its objects are monoids, that is a set A with a neutral element $e : A$ and a binary operation $\_ * \_$ such that the laws of a monoid hold $(x * e = x, e * x = x, x * (y * z) = (x * y) * z)$. A morphism between monoids $(A, e, \_ * \_)$ and $(B, e', \_ *' \_)$ is a function $f : A \to B$ which preserves the structure, i.e. $f\,e = e'$ and $f\,(x * y) = (f\,x) *' (f\,y)$. Again we only have to show that identity is a morphism and that morphisms are closed under composition to see that this is a category.

Question 7 *Do the categories* $\underline{\mathbf{Pre}}$ *and* $\underline{\mathbf{Mon}}$ *have initial and terminal objects?*

2.4 Monos and epis

A function $f : A \to B$ is injective if every element appears at most once in the image, i.e. $\forall x, y : A. f\,x = f\,y \implies x = y$, it is surjective if every element of the codomain is in the image: $\forall y : B. \exists x : A. f\,x = y$, it is bijective if it is both injective and surjective.

So for example the function $\mathrm{double} : \mathbb{N} \to \mathbb{N}$ which doubles its input is injective (but not surjective), while the function $\mathrm{half} : \mathbb{N} \to \mathbb{N}$ which divides by

2 ignoring remainders (eg. $\mathrm{half}\,5 = 2$) is surjective but not injective. On the other hand the function $\mathrm{swap} : \mathbb{N} \to \mathbb{N}$ which sends every even number to its successor and every odd number to its predecessor (e.g. $\mathrm{swap}\,5 = 4, \mathrm{swap}\,4 = 5$ is both injective and surjective and hence it is a bijection.

We can translate these notions into the language of category theory by noting that an injective function can be cancelled on the left side of a composition that is if $i : A \to B$ is injective then for all $f, g : C \to A$ it is the case that if $i \circ f = i \circ g$ then $f = g$. We call such a function a *monomorphism* or short *mono*. We draw monos with a tail at the end of the arrow:

$$\begin{tikzcd} A \arrow[r, tail, "i"] & B \end{tikzcd}$$

On the other hand a surjective function can be cancelled on the right hand side. That is given a surjective function $e : A \to B$ and $f, g : B \to C$ then if $f \circ e = g \circ e$ then $f = g$. We call a function with this property an *epimorphism* or short an *epi*. We draw epis with a double arrowhead:

$$\begin{tikzcd} A \arrow[r, two heads, "e"] & B \end{tikzcd}$$

Exercise 8 *Show that the monos in* **Set** *are exactly the injective functions and the epis are exactly the surjective functions.*

[4]

If a function $f : A \to B$ has a left inverse $l : B \to A$ with $l \circ f = \mathrm{id}$ then it is mono, because

$$\begin{aligned} f \circ g = f \circ h \implies & \quad l \circ (f \circ g) = l \circ (f \circ h) \\ \implies & \quad (l \circ f) \circ g = (l \circ f) \circ h \\ \implies & \quad \mathrm{id} \circ g = \mathrm{id} \circ h \\ \implies & \quad g = h \end{aligned}$$

Exercise 9 *Explicitly prove the dual construction: if $f : A \to B$ has a left inverse $r : B \to A$ with $f \circ r = \mathrm{id}$ then it is an epi.*

Since $\mathrm{half} \circ \mathrm{double} = \mathrm{id}$ we can conclude that double is a mono (because it has a left inverse) and half is an epi (because it has a right inverse). On the other hand swap is self inverse that is $\mathrm{swap} \circ \mathrm{swap} = \mathrm{id}$ hence it has both a left and a right inverse (itself) and hence it is both a mono and an epi. Indeed it is isomorphism and from the above it is easy to see that isomorphisms are always monos and epis.

In **Set** the inverse direction is also true:

[4]To prove that an epimorphism is a surjection, you need to use (effective) quotients: If $A : \mathbf{Set}$ and $R : A \to A \to \mathbf{Prop}$ is an equivalence relation then $A/R : \mathbf{Set}$ with $[\_] : A \to A/R$ and $R\,a\,a' \iff [a] = [a']$ and given $f : A \to B$ such that $R\,a\,a'$ then we have $\hat{f} : A/R \to B$ with $\hat{f}\,[a] = f\,a$.

Exercise 10 *Show that in* $\underline{\mathbf{Set}}$ *a morphism that is both mono and epi (i.e. bijections) is an isomorphism.* [5]

However, this is not true in general: consider the embedding $i : \mathbb{N} \to \mathbb{Z}$, this is a monoid morphism $i : \underline{\mathbf{Mon}}((\mathbb{N}, \_ + \_, 0), (\mathbb{Z}, \_ + \_, 0))$.

Exercise 11 *Show that i is both an epi and a mono. But it cannot be an iso. Why?*

However, if we know that the function is a mono and an epi because of inverses (we say it is a split mono and a split epi) that it is always an isomorphism:

Exercise 12 *Show that in any category a function that has both an left and a right inverse is always an isomorphism.*

[5] One direction uses the principle of unique choice, that is for $R : A \to B \to \mathbf{Prop}$

$$(\forall x : A.\exists! y : B.R\,x\,y) \to \exists f : A \to B.\forall x : A.R\,x\,(f\,x)$$

where $\exists!$ means there exists unique, i.e. $\exists! x : A.\phi\,x = \exists x : A.\phi\,x \wedge \forall y : A.\phi\,y \to x = y$. This principle is provable from univalence in Homotopy Type Theory.