Category theory course
Lecture 3
ITI9200, Spring 2020

Fosco Loregian
fouche@yoneda.ninja

February 14, 2020

Contents

1 Universal properties

1.1 Initial and terminal objects

Definition 1 (Initial and terminal objects)**.** *Let $\mathcal{C}$ be a category.*

- *An object $\varnothing : \mathcal{C}_o$ is called* initial *if for every other object $C : \mathcal{C}_o$ there is a single morphism $i_C : \varnothing \to C$.*
- *Dually, an object 1 is called* final *o* terminal *if for every object $C : \mathcal{C}_o$ there is a unique morphism $t_C : C \to 1$.*

(Note the substantial difference between 'there is ***at most*** *one morphism' and 'there is* ***exactly one*** *morphism'!)*

Remark 1. *As a consequence of the definition, if $\varnothing$ is an initial object in $\mathcal{C}$, then there is a single arrow $i_\varnothing : \varnothing \to \varnothing$, the identity of $\varnothing$. Similarly, if 1 is* *terminal, there is a unique arrow $t_1 : 1 \to 1$, the identity of 1; if $\mathcal{C}$ has both an initial and a terminal object, then there is a unique arrow $z : \varnothing \to 1$; we say that $\mathcal{C}$ has a* zero *object if z is an isomorphism.*

The simple proof of the following statement will enlighten the nature of the notion of universal property. An initial object $\varnothing : \mathcal{C}$ enjoys what is called a *universal property*:

Remark 2. *Let $\mathcal{C}$ be a category with an initial object $\varnothing$. If $\varnothing'$ is another initial object, then there is a unique isomorphism $\varnothing \cong \varnothing'$.*

Proof. Assume that there are two initial objects $\varnothing, \varnothing'$; then, by the respective universal properties of $\varnothing$ and $\varnothing'$, there is a unique arrow $u : \varnothing \to \varnothing'$, and similarly a unique arrow $v : \varnothing' \to \varnothing$ (as observed in remark 2, there is just one element in $\mathcal{C}(\varnothing, \varnothing)$, and just one element in $\mathcal{C}(\varnothing', \varnothing')$). The compositions $v \circ u$ and $u \circ v$ must be the identities of $\varnothing$ and $\varnothing'$ respectively, thus showing that u, v are mutually inverse isomorphisms. □

1.2 Products

Consider two sets A, B; the product $A \times B$ of the two is the set of all pairs

$$\{(a, b) \mid a : A, b : B\}.$$

It comes equipped with two functions $p_A : A \times B \to A, p_B : A \times B \to B$,

$$\begin{aligned} p_A \colon A \times B \to A, &\quad (x, y) \mapsto x \\ p_B \colon A \times B \to B, &\quad (x, y) \mapsto y \end{aligned}$$

with the property that for every pair of functions $f : X \to A, g : X \to B$ (X any set) there is a unique function $(f, g) : X \to A \times B$ such that the equations $p_A \circ (f, g) = f$ and $p_B \circ (f, g) = g$ hold; in other words, the diagram

$$\begin{tikzcd} & X \arrow[dl, "f"'] \arrow[d, dotted] \arrow[dr, "g"] & \\ A & A \times B \arrow[l, "p_A"'] \arrow[r, "p_B"] & B \end{tikzcd} \tag{1}$$

is commutative, when filled with $(f, g) : X \to A \times B$.

Of course, this is not something you don't know: you just gave it a different name until today. Given two types `a,b` you can build the *product type* `(a,b)` with the unique constructor `(,)` that coheres the pair `a,b` into a single type, and the two functions

```
-- p_A
fst :: (a,b) -> a
fst (x,_) =  x
-- p_B
snd :: (a,b) -> b
snd (_,y) = y
```

What does the above universal property really say though is that the two sets

$$\mathsf{Set}(X,A)\times\mathsf{Set}(X,B),\quad \mathsf{Set}(X,A\times B) \tag{2}$$

are in bijection:

- define a function

  $$\mathsf{Set}(X,A)\times\mathsf{Set}(X,B)\to\mathsf{Set}(X,A\times B)$$

  sending a pair (f,g) into the function $(f,g):X\to A\times B$ defined by $x\mapsto(fx,gx)$;
- define a function in the opposite direction:

  $$\mathsf{Set}(X,A\times B)\to\mathsf{Set}(X,A)\times\mathsf{Set}(X,B);$$

  each function $u:X\to A\times B$ goes to the pair of functions $(\pi_A\circ u,\pi_B\circ u)$.

It is evident that the composition of these two functions is the identity both ways, and this proves the result.

If you're more familiar with Haskell-ish notation, the result can be restated as follows: the two types `( x -> a, x -> b )` and `x -> ( a,b )` are isomorphic.[1] In `Control.Arrow` there is the function

```
(&&&) :: (a -> b) -> (a -> c) -> a -> (b, c)
```

and the isomorphism is realised just by sending `f g` to `&&& f g`.

[1] A word of explanation is in order: there is no truly satisfying way of proving the isomorphism in Haskell; we can however explicitly define functions

$$f:\mathsf{Set}(X,A)\times\mathsf{Set}(X,B)\leftrightarrows\mathsf{Set}(X,A\times B):g$$

and "prove" for example through `QuickCheck` that they are mutually inverse, i.e. that the two compositions `f . g` and `g . f` are the identity function `id a = a`. Do it, it'll be a good way to spend an afternoon meditating on this remark!

Definition 2. *Let $\mathcal{C}$ be a category, and $A, B : \mathcal{C}$ be objects; the* product *$A \times B : \mathcal{C}$ is a third object such that the universal property expressed in diagram (1) holds true: for every pair of morphisms $f : X \to A, g : X \to B$, there is a* unique *$(fg) : X \to A \times B$ such that the diagram commutes. In other words, there is a bijection of sets (even if X, A, B are not sets!)*

$$\mathcal{C}(X, A) \times \mathcal{C}(X, B), \quad \mathcal{C}(X, A \times B)$$

like the one in (2).

Examples of products in various categories of interest:

- in the category of sets and functions, the product of two sets is their *cartesian product*: its elements are the ordered pairs of elements (a, b), the first in A and the second in B.
- in the category of monoids and monoid homomorphisms: given two monoids M, N, we define a monoid structure on the cartesian product of their underlying sets by

  $$(a, b) \cdot (c, d) := (a \cdot_M c, b \cdot_N d)$$

  the associative and unital properties are easily checked.
- in *a* partially ordered set P, regarded as a category, the definition of product acquires the following form:

  > Given two elements $p, q : P$, there is an element $z = pq$ such that whenever $x \leq p, q$, then $x \leq pq$.

  Of course, this means that pq is the max element with the property of being less than both p, q: this means that a product for $p, q : P$ is a *meet*, or *infimum*, of p, q.
- In *the category of* partially ordered sets things are less straightforward but also more interesting. Let $f : X \to A$ and $g : X \to B$ two monotone functions; in order to show that the product $A \times B$ is the cartesian product of sets $A \times B$, endowed with the partial order relation $(a, b) \leq (a', b') \iff a \leq_A a', b \leq_B b'$, we shall show that the unique function $(f, g) : X \to A \times B$ is monotone with respect to this order. This is, however, straightforward: let $x \leq x'$; then $fx \leq_A fx'$, because f is monotone, and $gx \leq_B gx'$, because g is monotone.
- Let Set_* be the category of pointed sets, i.e. of pairs $(A, a : A)$; let $(A, a), (B, b)$ two sets. The product $(A \times B, (a, b))$ has the universal property of the product in pointed sets.

- in the category of graphs: a *graph* is a pair of sets E, V such that $E \subseteq \binom{V}{2}$ is a subset of the set of two-element subsets of V: each element $e = \{v, v'\}$ of E is an *edge* of the graph connecting the two vertices v, v'.
  - The *box product* of two graphs has the set $V_G \times V_{G'}$ as set of vertices, and there is an edge connecting (v_1, v_1') and (v_2, v_2') if and only if $v_1 = v_2$ and $\{v_1', v_2'\} : E_{G'}$ or $\{v_1, v_2\} : E_G$ and $v_1' = v_2'$.
  - The *tensor product* of two graphs has the set $V_G \times V_{G'}$ as set of vertices, and there is an edge connecting (v_1, v_1') and (v_2, v_2') if and only if there is an edge connecting v_1, v_2 and an edge connecting v_1', v_2'.

  It is easy to see that $G \,\Box\, G' \ncong G \otimes G'$: the two constructions are not isomorphic;[2] only the second construction has the universal property of the product in the category Gph of graphs and graph homomorphisms.

The following is a useful lemma:

Lemma 1. *Let $\mathcal{C}$ be a category, and let $u, v : T \to X \times Y$ two parallel morphisms between objects $T, X \times Y : \mathcal{C}$ (in particular, we assume that $X \times Y$ exists, and we let $p_X, p_Y : X \times Y \to X, Y$ be the product projections). Then $u = v$ if and only if*

$$\begin{cases} p_X \circ u = p_X \circ v \\ p_Y \circ u = p_Y \circ v \end{cases}$$

Proof. One direction of the implication is obvious: composition on the left and on the right is a *congruence*, so if $u = v$, a fortiori $p_X u = p_X v$, and $p_Y u = p_Y v$.

If the post-compositions with the projections are the same, though, then both u and v must have the universal property for the same map $\left[\begin{smallmatrix} p_X u \\ p_Y u \end{smallmatrix}\right]$. Or, to put it another way, the function $u \mapsto (p_X u, p_Y u)$ is bijective, hence injective: this means that if $p_X u = p_X v$, and $p_Y u = p_Y v$, i.e. if the pairs $(p_X u, p_Y u)$ and $(p_X v, p_Y v)$ are equal, then $u = v$. □

Theorem 1. *Let $\mathcal{C}$ be a category, $f : A \to A', g : B \to B'$ two morphisms;*

[2] Draw a picture: let $0 \text{---} 1$ be the graph having two vertices $0, 1$ connected by a single edge; show that $(0 \text{---} 1)^{\Box 2} \ncong (0 \text{---} 1)^{\otimes 2}$.

assume that $A \times B, A' \times B'$ both exist. Then the square

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

commutes.

Notation 1. *We define $f \times g$ to be either the $\urcorner$ side, or the $\llcorner$ side of the above square. To put it another way, the diagonal of the square above is well-defined as it does not change whatever path we choose from north-west to south-east; we call such diagonal $f \times g : A \times B \to A' \times B'$.*

Proof. In order to prove that the square commutes, according to the Lemma just stated, we just have to post-compose the $\urcorner$ side of the diagram with the $\llcorner$ side of the diagram:

$$\begin{aligned}
p_{A'} \circ (A' \times g) \circ (f \times B) &= p_{A'} \circ (f \times B) = f \\
p_{A'} \circ (f \times B') \circ (A \times g) &= f \circ p_A \circ (A \times g) = f \\
p_{B'} \circ (A' \times g) \circ (f \times B) &= g \circ p_B \circ (f \times B) = g \\
p_{B'} \circ (f \times B') \circ (A \times g) &= p_{B'} \circ (A \times g) = g.
\end{aligned}$$

(the notation is based on a slight, innocuous, abuse.) □

1.3 Coproducts

Consider now two sets A, B; their *disjoint sum* or *coproduct* is the set $A \coprod B$ made of two non-intersecting copies of A, B.[3] The set $A \coprod B$ comes equipped with canonical injections $i_A : A \to A \coprod B$ and $i_B : B \to A \coprod B$

Now, a function $f : A \coprod B \to X$ is completely determined by its compositions with the injections i_A, i_B, in the sense that for every other set X and pair of functions $f : A \to X$ and $g : B \to X$ there is a unique function $\begin{bmatrix} f \\ g \end{bmatrix} : A \coprod B \to X$ such that the diagram

$$\begin{tikzcd}
 & X & \\
A \arrow[ur, "f"] \arrow[r, "i_A"'] & A \coprod B \arrow[u, dotted] & B \arrow[ul, "g"'] \arrow[l, "i_B"]
\end{tikzcd}$$

[3] More formally, the set $A \coprod B$ is defined as the set of all pairs $(a, 陰), (陽, b)$, or (a, yin) and (yang, b), or `Left a` and `Right b`, as a, b run over the elements $(\ldots\text{terms?})$ of A, B.

is commutative when $\left[\begin{smallmatrix} f \\ g \end{smallmatrix}\right] : A \coprod B \to X$ fills the dotted arrow.

This translates into the bijection of sets

$$\mathcal{C}(A \coprod B, X) \cong \mathsf{Set}(A, X) \times \mathsf{Set}(B, X)$$

The attentive reader might have noted that this isomorphism is completely dual to the one for products proved above; sure this is not accidental: also the proof of the statement is completely analogous -but dual- to the one for products. The tight relation between the two constructions will become clear along the discussion about limits and colimits as dual constructions.

In Haskell, the isomorphism of types we have to prove is

```
( a -> x, b -> x ) ~ ( Either a b -> x )
```

Such an isomorphism is realised by the function

```
either :: (a -> x) -> (b -> x) -> Either a b -> x
either f _ (Left a) = f a
either _ g (Right b) = g b
```

and the isomorphism above can be realised as `f,g` $\mapsto$ `either f g`.

Examples of coproducts:

- in the category of sets, the coproduct of A, B is made by joining one copy of A “at level 0” and a copy of B “at level 1” (this seemingly artificial definition is instead the only way to prevent A, B from intersecting in the disjoint sum); the universal property of $A \coprod B$ has been fleshed out in the previous paragraphs.
- in many categories, like spaces and graphs, coproducts are “easy”: it is enough to join disjoint copies of object A and object B, consider the set-theoretic embeddings $A, B \hookrightarrow A \coprod B$, and this will turn out to have the correct universal property (make examples).
- in the category of vector spaces, given two vector spaces V, W their coproduct is called *direct sum*: it is the vector space whose elements are pairs (v, w) of vectors and whose operations are defined componentwise. Fun fact: this object has the same universal property as the product $V \times W$ (prove it!), so we are given not only injections $i_V, i_W : V, W \to V \oplus W$ sending v to $(v, 0)$ and w to $(0, w)$, but also projections $p_V, p_W : V \oplus W \to V, W$. Of course, $p_V i_W = 0 = p_W i_V$, etc.[4]
- the same happens in the category of abelian groups.

[4] The same is true for any *finite* sum of vector spaces, but not for infinite sums.

According to the "two theorems at the price of one" philosophy, we get dual versions of Lemma 1 and Theorem 1:

Lemma 2. *Let $\mathcal{C}$ be a category, and let $u, v : X \sqcup Y \to T$ two parallel morphisms between objects $T, X \sqcup Y : \mathcal{C}$ (in particular, we assume that $X \sqcup Y$ exists, and we let $i_X, i_Y : X, Y \to X \sqcup Y$ be the coproduct injections). Then $u = v$ if and only if*

$$\begin{cases} u \circ i_X = v \circ i_X \\ u \circ i_Y = v \circ i_Y \end{cases}$$

Theorem 2. *Let $\mathcal{C}$ be a category, $f : A \to A', g : B \to B'$ two morphisms; assume that $A \sqcup B, A' \sqcup B'$ both exist. Then the square*

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

commutes.

1.4 Natural numbers

Consider the following type declaration:

```
data N = Z
       | Suc N
       deriving (Eq, Show)
```

that defines the type of natural numbers; consider also the `plus` function

```
plus :: N -> N -> N
plus Z n = n
plus (Suc m) n = Suc (plus m n)
```

If you fire up `ghci`, and type `(Suc Z) `plus` (Suc Z)`, you pray the Haskell gods and you wait some time, you shall see

```
Prelude> (Suc Z) `plus` (Suc Z)
      => Suc (Suc Z)
```

So far so good. But what happened *exactly*? It turns out that the function `Suc :: N -> N` is "defined inductively" in such a way that from a base definition we can go back and back until we hit the base case of a recurrence.

The same happens for the function `_+n :: N -> N`: we can pattern-match from the base case `Z + n` to define

```
(Suc Z) + n = Suc (Z + n) = Suc n
(Suc (Suc Z)) + n = Suc ((Suc Z) + n) = Suc (Suc n)
```

etc. Of course, this is nothing more nothing less than *induction principle* à la Peano: here we want to characterise this property using the language of category theory; we will come to a pretty neat description of what natural numbers, and inductive definition, really are.

1.4.1 Dynamical systems, and N

Define a category Dyn whose objects are triples $(X, t : X \to X, x : X)$, i.e. diagrams

$$\begin{tikzcd} 1 \arrow[r, "x"] & X \arrow[r, "t"] & X \end{tikzcd}$$

and morphisms between (X, t, x) and (Y, g, y) consist of functions $u : X \to Y$ such that $u(x) = y$ and $u \circ t = g \circ u$, i.e. of commutative diagrams

$$\begin{tikzcd} 1 \arrow[r, "x"] \arrow[d, equal] & X \arrow[r, "t"] \arrow[d, "u"'] & X \arrow[d, "u"] \\ 1 \arrow[r, "y"'] & Y \arrow[r, "g"'] & Y \end{tikzcd}$$

The object $\mathbf{N} = (\mathbb{N}, \mathrm{s} : \mathbb{N} \to \mathbb{N}, 0 : \mathbb{N})$, or rather the type `N` of the type declaration above, belongs to this category if we let s be the function `Suc :: N -> N` sending a natural number to its successor.

Let $\mathbf{X} = (X, t, x)$ be any object of Dyn; then, there is an arrow $u : \mathbf{N} \to \mathbf{X}$ in Dyn such that[5]

$$\begin{tikzcd} 1 \arrow[r, "z"] \arrow[d, equal] & \mathbb{N} \arrow[r, "\mathrm{s}"] \arrow[d, "u"] & \mathbb{N} \arrow[d, "u"] \\ 1 \arrow[r, "x"'] & X \arrow[r, "t"'] & X \end{tikzcd}$$

[5] This means that given an initial value $u(0) = x$ and any endomorphism of the set X, there is a unique possible way to define a sequence of element $u(n)$ of X recursively, by setting $u(0) = x$ and $u(n + 1) = t(u(n))$.

and such a function u is *unique* with respect to this property;[6] this means precisely that $\mathbf{N}$ is an *initial object* of the category Dyn.

The category Dyn models the notion of a *dynamical system*: a set X and an initial state $x : X$ are given, together with a function $t : X \to X$ mapping *evolution* of the system in discrete time, according to the function t.[7]

Also, as you might expect, examples of such systems abound:

- The *product* function,

  ```
  prod :: N -> N -> N
  prod Z n = Z
  prod (Suc m) n = n `plus` (m `prod` n)
  ```

  that employs recursion to define $(m + 1) \times n = m \times n + n$, and the power function

  ```
  powe :: N -> N -> N
  powe Z _ = Z
  powe m Z = Suc Z
  powe m (Suc n) = prod (powe m n) m
  ```

- The *Ackermann function*:

  ```
  ackr :: N -> N -> N
  ackr Z n = plus (Suc Z) n
  ackr (Suc m) Z = ackr m (Suc Z)
  ackr (Suc m) (Suc n) = ackr m (ackr (Suc m) n)
  ```

  (do not compute `ackr 4 4` unless you *really* need it!).

  The request that in Dyn there is a morphism $u : \mathbf{N} \to \mathbf{X}$ means that

[6] If there is another sequence $v : \mathbb{N} \to X$ with the same property, the equality of u, v can be assessed "by induction" using t: $u(0) = v(0) = x$, and $u(n+1) = t(u(n)) = t(v_n) = v(n+1)$.

[7] The dynamical system determined by $t : X \to X$ consists of the iterates

$$u_n(x) = \{t^n(x) \mid n : \mathbb{N}\}.$$

The system lends itself to all sorts of questions: is there a fixed point for $u_n(x)$? Does the limit of $u_n(x)$ belong to X (not obvious: consider $X = [0, 1[$ and $u_n(x) \equiv 1 - \frac{1}{n}$)? Is $u_n(x)$ continuous in x? Etc.

we can find a function $u\ :\mathbb{N}\to X$ such that

$$\begin{tikzcd}
1 \arrow[r, "z"] \arrow[d, equal] & \mathbb{N} \arrow[r, "\mathrm{s}"] \arrow[d, "u"] & \mathbb{N} \arrow[d, "u"] \\
1 \arrow[r, "x"'] & X \arrow[r, "t"'] & X
\end{tikzcd}$$

This condition says that if u exists, it is unique because it is recursively defined using t and its iterates: if $u(\mathrm{s}(n)) = t(u(n))$, this means that u_{n+1} is forced to be $= t(u_n)$, and considering u as a sequence $\{u_n\}_{n:\mathbb{N}}$ of elements of X it suffices to specify its value at zero to get that

$$\begin{cases} u_1 & = t(u_0) \\ u_2 & = t(u_1) = t(t(u_0)) \\ u_3 & = t(t(t(u_0))) \\ & \vdots \end{cases}$$

Is there an implementation for the iterates of t to the element $x : X$? But of course there is! The `iterate` function.

```
iterate :: (a -> a) -> a -> [a]
iterate t x = x : iterate t (t x)
```

whose output is exactly the (infinite) list `[x, t x, t t x, t t t x, ...]`.

1.5 Diagram chasing

The definition of products and coproducts allows for more than a finite number of factors/summands: a tuple $A_1, \dots, A_n$ of sets, or objects of $\mathcal{C}$, can be arranged in a *product*

$$A_1 \times \cdots \times A_n$$

It turns out that all the sets obtained from a parenthesisation $\mathfrak{P}$ of an n-tuple of objects $[A_1, \dots, A_n]_{\mathfrak{P}}$ are isomorphic: the function $[A_1, \dots, A_n]_{\mathfrak{P}} \to [A_1, \dots, A_n]_{\mathfrak{Q}}$ that sends $(a_1, \dots, a_n)_{\mathfrak{P}}$ to $(a_1, \dots, a_n)_{\mathfrak{Q}}$ is a bijection, whose inverse just "rearranges" brackets. This just means, for example, that the element

$$(a, (b, c), d)$$

and the element

$$(a, (b, (c, d)))$$

correspond each other in a bijection $A \times (B \times C) \times D \to A \times (B \times (C \times D))$.

In a category-theoretic perspective, it is however quite unpleasant to refer to the "elements" of an object: oftentimes, there is no such concept available (for example: let P be a poset regarded as a category; what are the "elements" of $p : P$? And yet, $p \wedge q = q \wedge p$).

Establishing such a result uses a form of induction on the length of the tuple, and it turns out that in order to prove that the operation of categorical product and coproduct are "coherently associative", i.e. that every two different bracketing are canonically isomorphic, it is enough to prove the case $n = 3$, i.e. that the objects $A \times (B \times C)$ and $(A \times B) \times C$, formally defined in two different ways, satisfy the same universal property: joining the universal property that define $B \times C$ and $A \times B$ we will conclude.

So, we are in the following situation: $A, B, C : \mathcal{C}$ are three objects of a category; the products $(A \times B) \times C, A \times (B \times C)$ all exist. We want to show that $A \times (B \times C)$ and $(A \times B) \times C$ have the same universal property. In order to do so, we consider a diagram like

$$\begin{tikzcd}
(A\times B)\times C \arrow[rrrd, bend left] \arrow[dd] & & & \\
& A\times(B\times C) \arrow[ul, dotted] \arrow[r, "p_{B\times C}"] \arrow[dl, dotted, "(p_A{,}p'_B p_{B\times C})"'] \arrow[dd, "p_A"] & B\times C \arrow[r, "p'_C"'] \arrow[dd, "p'_B"] & C \\
A\times B \arrow[dr] \arrow[drr, bend left] & & & \\
& A & B &
\end{tikzcd}$$

Each of these functions is a projection on a factor; now, p_A and $p'_B \circ p_{B\times C}$ give a unique function $q_{A\times B} = (p_A, p'_B \circ p_{B\times C}) : A \times (B \times C) \to A \times B$ that paired with the composition $q_C : A \times (B \times C) \to B \times C \to C$ yields a unique function $u = (q_{A\times B}, q_C) : A \times (B \times C) \to (A \times B) \times C$ commuting with the projections on the factors.

A similar argument yields a unique $v : A \times (B \times C) \leftarrow (A \times B) \times C$, commuting with the projection on the factors; uniqueness now implies that there is no other choice for the functions

$$(A \times B) \times C \xrightarrow{v} A \times (B \times C) \xrightarrow{u} (A \times B) \times C$$

and

$$A \times (B \times C) \xrightarrow{u} (A \times B) \times C \xrightarrow{v} A \times (B \times C)$$

other than the identity of $(A \times B) \times C$ and of $A \times (B \times C)$.

What about commutativity of cartesian products? This is proved in the exact same way, but with a shorter construction for the two mutual inverses maps: $A \times B \xrightarrow{\left[\begin{smallmatrix} p_B \\ p_A \end{smallmatrix}\right]} B \times A$ yields a symmetry function, whose inverse is $B \times A \xrightarrow{\left[\begin{smallmatrix} p_A \\ p_B \end{smallmatrix}\right]} A \times B$. So, $A \times B$ and $B \times A$ are naturally isomorphic.

1.6 Free objects

Let X be any set; let us consider the set of all almost-everywhere-zero[8] functions $f : X \to \mathbb{Z}$ endowed with the sum operation

$$f + g : X \to \mathbb{Z} : x \mapsto f(x) + g(x).$$

It is easily seen that this set is an abelian group, with identity element the function constantly zero.

This set is called the *free group* on the set X. The free group on a set X s endowed with a canonical map $\delta : X \to \mathbb{Z}(X) : x \mapsto \delta_x$: $\delta_x(y) = 1$ if $x = y$, and 0 otherwise.

The free group on a set X enjoys the following universal property:

> For every function of sets $h : X \to G$, where $\mathbb{G} = (G, +, 0)$ is an abelian group, there is a unique extension of h to a group homomorphism $\bar{h} : \mathbb{Z}(X) \to \mathbb{G}$ such that the triangle
>
> $$\begin{tikzcd} & X \arrow[dl, "\delta"'] \arrow[dr, "h"] & \\ \mathbb{Z}(X) \arrow[rr, dotted, "\bar{h}"'] & & \mathbb{G} \end{tikzcd}$$
>
> is commutative.

In order to prove the property, let us argue this way: let $h : X \to G$ be a function of sets; by the request that $\bar{h} \circ \delta = h$ it is evident that we have to define $\bar{h}(\delta_x) = h(x)$. But then, $\bar{h}$ is uniquely determined by this condition: observe that every element in $\mathbb{Z}(X)$ results as a finite sum $f = \sum_{x:\mathsf{sup}(f)} f(x)\delta_x$, and since $\bar{h}$ has to be a group homomorphisms,

$$\bar{h}(f) = \bar{h}\left(\sum_{x:\mathsf{sup}(f)} f(x)\delta_x\right) = \sum_{x:\mathsf{sup}(f)} f(x)h(x).$$

A similar property can be stated to obtain *free objects* of all sorts:

[8] This means that the *support* of f, i.e. the set $\mathsf{sup}(f) = \{x : X \mid f(x) \neq 0\}$ is finite.

- For every set X there is a *free monoid* on X obtained as the set of all finite strings over the set X,[9] with string concatenation $x_n : X^{\times n}, y_m : X^{\times m} \mapsto (x.y)_{n+m}$ as operation, the empty string as identity element. The free monoid on a set X enjoys the following universal property:

  > For every function of sets $h : X \to M$, where $\mathbb{M} = (M, \cdot, 1)$ is a monoid, there is a unique extension of h to a monoid homomorphism $\bar{h} : FX \to \mathbb{M}$ such that the triangle
  >
  > $$\begin{tikzcd} & X \arrow[dl, "\delta"'] \arrow[dr, "h"] & \\ FX \arrow[rr, dotted, "\bar{h}"'] & & \mathbb{M} \end{tikzcd}$$
  >
  > is commutative.

- For every set X there is a *free group* on X obtained as the free monoid over $X \sqcup \bar{X}$ ($\bar{X}$ is a copy of X containing all formal inverses of elements of X), where words have been further reduced according to the rule $x\bar{x} \rightsquigarrow (), \bar{x}x \rightsquigarrow ()$. The free monoid on a set X enjoys the following universal property:

  > For every function of sets $h : X \to G$, where $\mathbb{G} = (G, \cdot, 1)$ is a group, there is a unique extension of h to a group homomorphism $\bar{h} : FX \to \mathbb{G}$ such that the triangle
  >
  > $$\begin{tikzcd} & X \arrow[dl, "\delta"'] \arrow[dr, "h"] & \\ FX \arrow[rr, dotted, "\bar{h}"'] & & \mathbb{G} \end{tikzcd}$$
  >
  > is commutative.

- Let G be a directed graph (it means that the set $E(G)$ of edges of G is an *ordered* pair). The *free category* CG on G consists of the category having the set of vertices of G as objects, and where the morphisms are freely generated by the set of edges. This means that we recursively define a category as follows:

  - There is a morphism $()_u : u \to u$;

[9] This means that we consider the set $X^* := \coprod_{n \geq 0} X^{\times n}$, i.e. the disjoint union $\{()\} \sqcup X \sqcup (X \times X) \sqcup \cdots \sqcup (X \times \cdots \times X) \sqcup \cdots$.

  - There is a morphism $e : u \to v$ for every $e = (u, v) : E(G)$;
  - Whenever there is a morphism $f : u \to v$ and a morphism $g : v \to w$, there is a morphism $g \odot f : u \to w$, (wicked souls also write $f; g : u \to w$).

  Of course, $()_u$ plays the rôle of the identity arrow, and all this is just a convoluted way to assert that the morphisms of CG are the free partial[10] monoid on the set of edges of G; try to formalize properly this assertion. (Recall how you define `String a = [a]`:

  ```
  [] :: [a]
  (:) :: a -> [a] -> [a]
  : x ys :: [a]
  ```

  This is an inductive definition for the free monoid on the set/type X, i.e. for the monoid X^* above.)

- The free graph on a set X has X as set of vertices, and te empty set as set of edges.

1.7 Universal property as initiality

The above heuristics suggest that we can informally state the property of "being universal" for an object of a category $\mathcal{A}$ as the property of being initial or terminal in some category $\tilde{\mathcal{A}}$ obtained starting from $\mathcal{A}$ (often, there is an explicit procedure to build $\tilde{\mathcal{A}}$ out of $\mathcal{A}$):

- the product $A \times B =$ `(a,b)` of two types `a`, `b` is the terminal object in the category of all pairs ${}_{A\langle}\mathcal{A}_{\rangle B}$ whose objects are "spans" $A \overset{a}{\leftarrow} X \overset{b}{\to} B$, and whose morphisms $u : (a, b) \mapsto (a', b')$ are $u : X \to X'$ such that $a'u = a, b'u = b$.
- the coproduct $A \coprod B =$ `Either a b` of two types `a`, `b` is the initial object in the category of all pairs ${}_{A\rangle}\mathcal{A}_{\langle B}$ whose objects are "cospans" $A \overset{a}{\to} X \overset{b}{\leftarrow} B$, and whose morphisms $u : (a, b) \mapsto (a', b')$ are $u : X \to X'$ such that $ua = a', ub = b'$.
- the set of natural numbers is the initial object of the category Dyn defined as above;

[10] This just means that not all elements of the "monoid" can be composed: the product function $\odot : M \times M \to M$ is only partially defined on a domain $(M \times M)_\circ \subseteq M \times M$ of "composable" elements/arrows.

- the free monoid on a set X is the initial object in the category whose objects are functions $u : X \to M$, where M is the underlying set of a monoid $\mathbb{M} = (M, \cdot, 1)$, and morphisms are the monoid homomorphisms $f : N \to M$ such that

$$\begin{tikzcd} & X \arrow[dl, "u"'] \arrow[dr, "v"] & \\ UN \arrow[rr, "Uf"'] & & UM \end{tikzcd}$$

  is a commutative triangle.

- the free (abelian) group on a set, the free category on a set, etc. all enjoy similar universal properties: the free thingummy on a set X is the initial object in the category whose objects are functions $u : X \to T$, where $T = U\mathbb{T}$ is the underlying set of a thingummy $\mathbb{T} = (T, \#, \rhd, \dots, \text{ঌ})$, and morphisms $\begin{bmatrix} X \\ \downarrow \\ U\mathbb{S} \end{bmatrix} \to \begin{bmatrix} X \\ \downarrow \\ U\mathbb{T} \end{bmatrix}$ are the thingummy homomorphisms $f : S \to T$ such that

$$\begin{tikzcd} & X \arrow[dl, "u"'] \arrow[dr, "v"] & \\ U\mathbb{S} \arrow[rr, "Uf"'] & & U\mathbb{T} \end{tikzcd}$$

If we call a structure *free* when it exhibits no relations, i.e. it can be presented by a set of formal operations of some sort (addition, concatenation, composition of arrows...) without the request that these operations satisfy any kind of binding equation, we see that the notion of freeness admits a neat categorical formalisation: whenever a set can be endowed with one or more operations, we can build a correspondence

$$\begin{tikzcd}[row sep=tiny] U : \mathsf{Thng} \arrow[r] & \mathsf{Set} \\ \mathbb{T} \arrow[r, mapsto] & U\mathbb{T} \end{tikzcd}$$

that forgets the presence of operations on a thingummy $\mathbb{T} = (T, \dots)$; think of a thingummy as a Christmas tree with bells and whistles. The correspondence U removes all the decorations from a tree $\mathbb{T}$, and leaves the bare **u**nderlying set T.

In the opposite direction, we can think of a constructive algorithm that exhibits the free thingummy on a set X; this is a correspondence

$$\begin{tikzcd}[row sep=tiny] F : \mathsf{Set} \arrow[r] & \mathsf{Thng} \\ X \arrow[r, mapsto] & FX \end{tikzcd}$$

exhibiting the **f**ree thingummy generated by recursively considering elements $x, y : X$, formal composites $x \rhd y$ for each operation $\rhd$ that defines the thingummy, and formal composites $u \rhd w$ for each operation $\rhd$ that defines the thingummy structure, and each element obtained at the preceding step.

Such constructions are examples of *functors*. We will investigate their properties in the following lecture.

1.7.1 Exercises

Exercises denoted with a star symbol are supposed to be difficult. Don't be put off, and enjoy!

1. Let $\mathcal{C} = (\mathbb{N}, \geq)$ be the category having objects the natural numbers $0, 1, 2, \dots$ and where there is a morphism $m \to n$ if and only if $n \leq m$. Does $\mathcal{C}$ have a terminal object? Does it have an initial object? Does it have products $n \times m$? Does it have coproducts $n \sqcup m$? Answer the same four questions for the category $\mathcal{D} = (\mathbb{N}, \_ \mid \_)$, that has the same objects and where there is a morphism $n \to m$ if and only if $m = kn$ for some $k \in \mathbb{N}$ (the relation $n \mid m$ reads as "n divides m").

2. Let 1 be the set with a single element; consider the correspondence that sends a set A into $A \sqcup 1$, i.e. to the coproduct of A and 1. Show that the following diagrams commute:

$$\begin{tikzcd}
1 \sqcup 1 \sqcup 1 \arrow[r, "1 \sqcup \mathsf{fold}"] \arrow[d, "\mathsf{fold} \sqcup 1"'] & 1 \sqcup 1 \arrow[d, "\mathsf{fold}"] \\
1 \sqcup 1 \arrow[r, "\mathsf{fold}"'] & 1
\end{tikzcd}$$

where $\mathsf{fold} : 1 \sqcup 1 \to 1$ is the function $\begin{bmatrix} \mathrm{id}_1 \\ \mathrm{id}_1 \end{bmatrix}$. As a consequence, the diagram

$$\begin{tikzcd}
A \sqcup 1 \sqcup 1 \sqcup 1 \arrow[r, "1 \sqcup \mathsf{fold}"] \arrow[d, "\mathsf{fold} \sqcup 1"'] & A \sqcup 1 \sqcup 1 \arrow[d, "\mathsf{fold}"] \\
A \sqcup 1 \sqcup 1 \arrow[r, "\mathsf{fold}"'] & A \sqcup 1
\end{tikzcd}$$

remains commutative.

Show that the set of functions $a : A \sqcup 1 \to A$ such that the following diagrams commute

$$\begin{tikzcd}
A \arrow[r, "i_A"] \arrow[dr, equal] & A \sqcup 1 \arrow[d, "a"] \\
& A
\end{tikzcd}
\qquad
\begin{tikzcd}
A \sqcup 1 \sqcup 1 \arrow[r, "A \sqcup \mathsf{fold}"] \arrow[d, "a \sqcup 1"'] & A \sqcup 1 \arrow[d, "a"] \\
A \sqcup 1 \arrow[r, "a"'] & A
\end{tikzcd}$$

is in bijection with the elements of A; namely, each such $a : A \sqcup 1 \to A$ determines a unique element $a_0 : A$.

3. You have defined the free monoid FX and the free category CX on a set X. Prove that the two constructions agree when a monoid M is regarded as a category with

a single object. [Hint (highlight the following lines on a pdf viewer):

]

4. Describe the *free magma* on a set X, recalling that a *magma* is just a set with a function $f : M \times M \to M$ with no further property whatsoever. [Hint (highlight the following lines on a pdf viewer):

   ] Why the following snippet of code,

```
-- prints the different parenthesisations of a tuple of n elements
fMag :: Integer -> [String]
fMag 1 = ["*"]
fMag 2 = ["(**)"]
fMag n = concat
  [ concat [ map (x ++) (fMag (n-i))
  | x <- fMag i]
  | i <- [1..(n-1)]
  ]
```

   solves the problem?

5. ($\star$) Describe coproducts in the category of groups:

   - Given two groups G, H, define $G \coprod H$ as follows (and denote it $G * H$): the set of elements of $G * H$ is the set of all words $x_1 \cdots x_n$ of elements of $G \cup H$, that have been suitably *reduced* in such a way that (a) every occurrence of the identity $1_G, 1_H$ is removed, and (b) every pair of contiguous elements $x_i x_{i+1}$ that belong to the same set have been replaced by their product in G or H. Show that every element of the resulting set can be expressed as a reduced word
   $$g_1 h_1 g_2 h_2 \cdots g_\ell h_\ell$$
   where $g_i : G, h_i : H$ and $\ell \geq 0$. The group operation on $G * H$ is defined by concatenation of words, followed by reduction; the identity element is the empty sequence, i.e. the reduction of either the word (1_G) or the word (1_H).
   - Show that if $G = \langle S \mid R \rangle$ and $H = \langle S' \mid R' \rangle$ are presentations by generator/relations of G, H respectively, then $G * H = \langle S \cup S' \mid R \cup R' \rangle$.
   - Let $C_2 = \langle x \mid x^2 \rangle$ be the cyclic group of order 2. Show that the group $C_2 * C_2$ is infinite. Can a free product of groups $G * H$ be finite?

6. ($\star$) The *mathematical induction principle* says that
   $$\left(Q0 \wedge \textstyle\bigwedge_{i \leq n} Qi \Rightarrow Q(i+1)\right) \Rightarrow \textstyle\bigwedge_{n:\mathbb{N}} Qn$$
   (in words, if $Q : \mathbb{N} \to \{0,1\}$ is a proposition, $Q0$ is true, and $Qn \Rightarrow Q(n+1)$, then Qn is true for all $n : \mathbb{N}$. Is there a way to state the induction principle in terms of the universal property of $\mathbb{N}$? [Hint (highlight the following lines on a pdf viewer):

   ]