Category Theory

Haskell and Cryptocurrencies

Dr. Andres Löh, Well-Typed LLP
Dr. Lars Brünjes, IOHK

2018-02-09

INPUT | OUTPUT

Goals

- Categories
- Functors
- Initial- & Final Objects
- Sums & Products
- Natural Transformations

Motivation

Sets

- Let S, T and U be sets, and let $f : S \to T$ and $g : T \to U$ be functions. Then there is a composition function $gf = g \circ f : S \to U$.
- Function composition is associative: If $h : U \to V$ is a third function, then $h(gf) = (hg)f$.
- For each set S, we have the identity function $\mathrm{id}_S = 1_S : S \to S$.
- If $f : S \to T$ is a function, then $f1_S = f$.
- If $g : R \to S$ is a function, then $1_S g = g$.
- So identity functions are neutral with respect to composition.
- Two sets X and Y are isomorphic if there are functions $f : X \to Y$ and $g : Y \to X$ with $fg = 1_X$ and $gf = 1_Y$. Functions f and g are called isomorphisms (or bijections). For all intents and purposes, isomorphic sets are “equal”.

Vector spaces

- Let k be a field, let S, T and U be k-vector spaces, and let $f : S \to T$ and $g : T \to U$ be k-linear maps. Then the composition $gf = g \circ f : S \to U$ is also a k-linear map.
- Composition of k-linear maps is associative: If $h : U \to V$ is a third k-linear map, then $h(gf) = (hg)f$.
- For each k-vector space S, the identity function $\mathrm{id}_S = 1_S : S \to S$ is k-linear.
- If $f : S \to T$ is a k-linear, then $f1_S = f$.
- If $g : R \to S$ is a k-linear, then $1_S g = g$.
- So the identity k-linear maps are neutral with respect to composition of k-linear maps.
- Two k-vector spaces X and Y are isomorphic if there are k-linear maps $f : X \to Y$ and $g : Y \to X$ with $fg = 1_X$ and $gf = 1_Y$. The maps f and g are called isomorphisms. For all intents and purposes, isomorphic k-vector spaces are "equal".

Groups

- Let S, T and U be groups, and let $f : S \to T$ and $g : T \to U$ be group homomorphisms. Then the composition $gf = g \circ f : S \to U$ is also a group homomorphism.
- Composition of group homomorphisms is associative: If $h : U \to V$ is a third group homomorphism, then $h(gf) = (hg)f$.
- For each group S, the identity function $\mathrm{id}_S = 1_S : S \to S$ is a group homomorphism.
- If $f : S \to T$ is a group homomorphism, then $f1_S = f$.
- If $g : R \to S$ is a group homomorphism, then $1_S g = g$.
- So the identity group homomorphisms are neutral with respect to composition of group homomorphisms.
- Two groups X and Y are isomorphic if there are group homomorphisms $f : X \to Y$ and $g : Y \to X$ with $fg = 1_X$ and $gf = 1_Y$. The homomorphisms f and g are called (group) isomorphisms. For all intents and purposes, isomorphic groups are “equal”.

Topological spaces

- Let S, T and U be topological spaces, and let $f : S \to T$ and $g : T \to U$ be continuous. Then the composition $gf = g \circ f : S \to U$ is also continuous.
- Composition of continuous maps is associative: If $h : U \to V$ is a third continuous map, then $h(gf) = (hg)f$.
- For each topological space S, the identity function $\mathrm{id}_S = 1_S : S \to S$ is continuous.
- If $f : S \to T$ is a continuous, then $f1_S = f$.
- If $g : R \to S$ is a continuous, then $1_S g = g$.
- So the identity continuous maps are neutral with respect to composition of continuous maps.
- Two topological spaces X and Y are isomorphic if there are continuous maps $f : X \to Y$ and $g : Y \to X$ with $fg = 1_X$ and $gf = 1_Y$. The maps f and g are called isomorphisms (or homeomorphisms). For all intents and purposes, isomorphic topological spaces are "equal".

Homotopy

- Let X be a topological space, let $s, t, u \in X$ be points, let $f : s \rightsquigarrow t$ and $g : t \rightsquigarrow u$ be *paths*, and let $[f]$ and $[g]$ be their *homotopy classes*. Then the composition $[g][f] = [gf] : s \rightsquigarrow u$ is also a (class of a) path.
- Composition is associative: If $h : u \rightsquigarrow v$ is a third path, then $h(gf) \cong (hg)f$.
- For each point $s \in X$, we have the *constant path* $\mathrm{id}_s = 1_s : s \rightsquigarrow s$.
- If $f : s \rightsquigarrow t$ is a path, then $f1_s \cong f$.
- If $g : r \to s$ is a path, then $1_s g \cong g$.
- So the constant paths are neutral with respect to composition of (classes of) paths.
- If $f : s \rightsquigarrow t$ is a path, then there is a path $f^{-1} : t \rightsquigarrow s$, such that $ff^{-1} \cong f^{-1}f \cong 1_s$. For all intent and pupose, two points connected by a path are indistinguishable from the point of view of homotopy theory.

Categories

Category

A category $\mathcal{C}$ is given by the following data:

- A class/set $\mathrm{Ob}(\mathcal{C})$ of objects.
- For each pair $X, Y \in \mathrm{Ob}(\mathcal{C})$ of objects, a set $\mathrm{Mor}_{\mathcal{C}}(X, Y)$ of morphisms. A morphism $f \in \mathrm{Mor}_{\mathcal{C}}(X, Y)$ is often written as $f : X \to Y$.
- For each triple X, Y, Z of objects a map
  $$\circ : \mathrm{Mor}_{\mathcal{C}}(Y, Z) \times \mathrm{Mor}_{\mathcal{C}}(X, Y) \to \mathrm{Mor}_{\mathcal{C}}(X, Z),$$
  called composition, which must be associative (i.e. $f(gh) = (fg)h$ for composable morphisms).
- For each object $X \in \mathrm{Ob}(\mathcal{C})$, a morphism $\mathrm{id}_X = 1_X : X \to X$ in $\mathrm{Mor}_{\mathcal{C}}(X, X)$, the identity (morphism) of X, such that the identity morphisms are neutral with respect to composition of morphisms (i.e. $f1_X = f$ and $1_X g = g$ for all suitable f and g).

Isomorphism

- Let $\mathcal{C}$ be a category, let $X, Y \in \mathrm{Ob}(\mathcal{C})$ be objects, and let $f : X \to Y$ be a morphism.
- f is called an isomorphism if there is a morphism $g : Y \to X$ with $fg = 1_Y$ and $gf = 1_X$.
- X and Y are called isomorphic ($X \cong Y$) if there exists an isomorphism $f : X \to Y$.
- An isomorphism $f : X \to Y$ is often denoted by $f : X \xrightarrow{\sim} Y$.
- If $f : X \xrightarrow{\sim} Y$ is an isomorphism, then there is *exactly one* $g : Y \to X$ with $fg = 1_Y$ and $gf = 1_X$:

$$g' = g' 1_Y = g'(fg) = (g'f)g = 1_X g = g.$$

This unique g is called the inverse of f and denoted by f^{-1}.

Isomorphic versus equal

- If two objects X and Y of a category are isomorphic, then all *categorical properties* (i.e. properties formulated in the language of category theory) that hold for X also hold for Y and vice versa.
- Therefore isomorphic objects are "as good as equal" – undistinguishable from a categorical point of view.
- Often objects are only known "up to isomorphism", and that is good enough. Equality does not really make sense in a categorical setting.

Example: $\underline{\mathrm{Set}}$

- The "mother of all categories" is $\underline{\mathrm{Set}}$, the category of sets.
- Objects in $\underline{\mathrm{Set}}$ are *sets*. (Note that the there is no "set of all sets", hence in general, the objects of a category don't form a set.)
- Morphisms $\mathrm{Mor}_{\underline{\mathrm{Set}}}(X, Y)$ are (total) *functions* from X to Y.
- Composition is usual function composition.
- Identities are usual identity functions.

Translating set-theoretical concepts

- Many concepts and properties that – at first glance – seem to be intimately tied to set-theory (mentioning *elements* and so on) have an alternative categorical formulation.

Translating set-theoretical concepts

- Many concepts and properties that – at first glance – seem to be intimately tied to set-theory (mentioning *elements* and so on) have an alternative categorical formulation.
- Take the usual definition for *injectivity* as an example: A function $f : S \to T$ is injective iff

$$\forall s, s' \in S : \; f(s) = f(s') \Rightarrow s = s'.$$

Translating set-theoretical concepts

- Many concepts and properties that – at first glance – seem to be intimately tied to set-theory (mentioning *elements* and so on) have an alternative categorical formulation.
- Take the usual definition for *injectivity* as an example: A function $f : S \to T$ is injective iff

$$\forall s, s' \in S : \; f(s) = f(s') \Rightarrow s = s'.$$

- However, this is equivalent to the following property, which is formulated purely in terms of categorical notions:

$$\forall U : \mathrm{Ob}(\underline{\mathrm{Set}}) : \; \forall i, j : \mathrm{Mor}_{\underline{\mathrm{Set}}}(U, S) : \; fi = fj \Rightarrow i = j.$$

$$\begin{tikzcd} U \arrow[r, shift left, "i"] \arrow[r, shift right, "j"'] \arrow[rr, bend left, "fi"] \arrow[rr, bend right, "fj"'] & S \arrow[r, red, "f"] & T \end{tikzcd}$$

Translating set-theoretical concepts

- Many concepts and properties that – at first glance – seem to be intimately tied to set-theory (mentioning *elements* and so on) have an alternative categorical formulation.
- Take the usual definition for *injectivity* as an example: A function $f : S \to T$ is injective iff

$$\forall s, s' \in S : \; f(s) = f(s') \Rightarrow s = s'.$$

- However, this is equivalent to the following property, which is formulated purely in terms of categorical notions:

$$\forall U : \mathrm{Ob}(\underline{\mathrm{Set}}) : \;\; \forall i, j : \mathrm{Mor}_{\underline{\mathrm{Set}}}(U, S) : \; fi = fj \Rightarrow i = j.$$

- This latter definition makes sense in *any* category. A morphism f with this property is called a monomorphism.
- Dually (revert all arrows!), surjective maps can be defined in categorical terms and are called epimorphisms then.

Example: sets with structure

- A huge class of examples for categories is of the form "set plus extra structure".
- Objects are sets with some extra structure.
- Morphisms are maps that "respect" the structure.

Example: sets with structure

- A huge class of examples for categories is of the form “set plus extra structure”.
- Objects are sets with some extra structure.
- Morphisms are maps that “respect” the structure.
- The categories of (Abelian) *groups* with *group homomorphisms* ($\underline{\mathrm{Ab}}$ and $\underline{\mathrm{Grp}}$).
- For a field k, the category of *k-vector spaces* and *k-linear maps*.
- The category of (commutative) *rings* with *ring homomorphisms* $\underline{\mathrm{Ring}}$.
- The catgeory of *topological spaces* with *continuous maps* $\underline{\mathrm{Top}}$.
- The category of (real/complex) *manifolds* with *differentiable maps*.
- ...

Example: simplex category

- We define the simplex category Δ as follows:
- Objects are non-empty, finite sets $[n] := \{0, 1, \ldots, n\}$ for $n \in \mathbb{N}$.
- Morphisms $[m] \to [n]$ are *monotonically increasing* maps.
- Composition is usual function composition.
- The identity $1_{[n]}$ is the usual identity function $[n] \to [n]$.

Example: one group

- Let G be a group. We can regard G as a category $\underline{G}$ as follows:
- There is exactly one object, let's call it $*$.
- Morphisms $* \to *$ are *elements* of G.
- Composition is given by the group operation.
- The (only) identity 1_* is the neutral element.

Example: partially ordered set

- Let $(S, \leq)$ be a *partially ordered set*. We can turn S into a category $\underline{S}$ as follows:
- Objects are the elements of S.
- For $x, y \in S$, we define

$$\mathrm{Mor}(x, y) := \begin{cases} \{*\} & \text{if } x \leq y \\ \varnothing & \text{otherwise} \end{cases}$$

- Composition is given by *transitivity*.
- Identities are given by *reflexivity*.
- Note that $x, y \in S$ are *isomorphic* if $x \leq y \wedge y \leq x$, i.e. iff $x = y$ (by *antisymmetry*).

Example: opposite category

- Let $\mathcal{C}$ be a category, We define the opposite category $\mathcal{C}^{\mathrm{op}}$ as follows:
- Objects of $\mathcal{C}^{\mathrm{op}}$ are just the objects of $\mathcal{C}$.
- For objects $X, Y \in \mathrm{Ob}(\mathcal{C}^{\mathrm{op}}) = \mathrm{Ob}(\mathcal{C})$, we define $\mathrm{Mor}_{\mathcal{C}^{\mathrm{op}}}(X, Y) := \mathrm{Mor}_{\mathcal{C}}(Y, X)$.
- Composition and identities are given by composition and identities in $\mathcal{C}$.

Example: product

- Let $\mathcal{C}$ and $\mathcal{D}$ be categories. We define the product category $\mathcal{C} \times \mathcal{D}$ as follows:
- Objects of $\mathcal{C} \times \mathcal{D}$ are pairs (X, Y) with $X \in \mathrm{Ob}(\mathcal{C})$ and $Y \in \mathrm{Ob}(\mathcal{D})$.
- For objects $(X, Y), (X', Y') \in \mathrm{Ob}(\mathcal{C} \times \mathcal{D})$, morphisms $(X, Y) \to (X', Y')$ are pairs of morphisms (f, f') with $f : X \to X'$ and $g : Y \to Y'$.
- For an object $(X, Y) \in \mathrm{Ob}(\mathcal{C} \times \mathcal{D})$, $1_{(X,Y)} = (1_X, 1_Y)$, and composition is given componentwise.

Example: slice category

- Let $\mathcal{C}$ be a category, and let $X \in \mathrm{Ob}(\mathcal{C})$ be an object. We define the slice category $\mathcal{C}/X$ as follows:
- Objects are morphisms $Y \to X$ in $\mathcal{C}$.
- A morphism from $f : Y \to X$ to $g : Z \to X$ is a morphism $\alpha : Y \to Z$ in $\mathcal{C}$ such that $g\alpha = f$.

$$\begin{tikzcd} Y \arrow[rr, "\alpha"] \arrow[dr, "f"'] & & Z \arrow[dl, "g"] \\ & X & \end{tikzcd}$$

- Composition and identity are given by composition and identity in $\mathcal{C}$.

Example: $\underline{\mathrm{Hask}}$

- We define the category $\underline{\mathrm{Hask}}$ of Haskell types and functions as follows:
- Objects are Haskell types (of kind $*$).
- A morphism $f \in \mathrm{Mor}_{\underline{\mathrm{Hask}}}(a, b)$ is a (total) Haskell function `f :: a -> b`. We identify two such functions `f` and `g` if they "behave" the same, i.e. produce the same output for all inputs: `f x = g x` for all `x`.
- Composition is given by Haskell composition of functions `.`, and identities are given by Haskell's polymorphic identity function `id`, restricted to the type in question.

Functors

Functor

- Let $\mathcal{C}$ and $\mathcal{D}$ be categories. Then a Functor $F : \mathcal{C} \to \mathcal{D}$ is given by the following data:
- For each object $X \in \mathrm{Ob}(\mathcal{C})$ an object $FX \in \mathrm{Ob}(\mathcal{D})$.
- For each morphism $f : X \to Y$ in $\mathcal{C}$, a morphism $Ff : FX \to FY$ in $\mathcal{D}$, so that the following conditions hold:
- For each $X \in \mathrm{Ob}(\mathcal{C})$, we have $F1_X = 1_{FX}$, i.e. identities are mapped to identities.
- For objects $X, Y, Z \in \mathcal{C}$ and morphisms $f : X \to Y$ and $g : Y \to Z$, we have $F(gf) = Fg \circ Ff$ as morphisms in $\mathcal{D}$, i.e. the functor “respects compositions”.

Functor (contd.)

$$
\begin{tikzcd}
X \arrow[d, "f"] \arrow[dd, bend right, "gf"'] \\
Y \arrow[d, "g"] \\
Z
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
FX \arrow[d, "Ff"'] \arrow[dd, bend left, "F(gf)"] \\
FY \arrow[d, "Fg"'] \\
FZ
\end{tikzcd}
$$

Example: identity functor

- Let $\mathcal{C}$ be a category. Then the identity functor $1_{\mathcal{C}} : \mathcal{C} \to \mathcal{C}$ is defined as follows:
- For each object $X \in \mathrm{Ob}(\mathcal{C})$, we have $1_{\mathcal{C}} X = X$.
- For each morphism $f : X \to Y$, we have $1_{\mathcal{C}} f = f$.
- The functor laws are obviously satisfied!

Example: composition of functors

- Let $\mathcal{C}$, $\mathcal{D}$ and $\mathcal{E}$ be categories, and let $F : \mathcal{C} \to \mathcal{D}$ and $G : \mathcal{D} \to \mathcal{E}$ be functors. Then we define the composition $GF : \mathcal{C} \to \mathcal{E}$ as follows:
- For each object $X \in \mathrm{Ob}(\mathcal{C})$, we have $(GF)X = G(FX)$.
- For each morphism $f : X \to Y$, we have $(GF)f = G(Ff)$.
- The functor laws for GF follow trivially from the functor laws for F and G.

Example: composition of functors

- Let $\mathcal{C}$, $\mathcal{D}$ and $\mathcal{E}$ be categories, and let $F : \mathcal{C} \to \mathcal{D}$ and $G : \mathcal{D} \to \mathcal{E}$ be functors. Then we define the composition $GF : \mathcal{C} \to \mathcal{E}$ as follows:
- For each object $X \in \mathrm{Ob}(\mathcal{C})$, we have $(GF)X = G(FX)$.
- For each morphism $f : X \to Y$, we have $(GF)f = G(Ff)$.
- The functor laws for GF follow trivially from the functor laws for F and G.
- We get yet another example for categories, the category of categories $\underline{\mathrm{Cat}}$: Objects are categories, and morphisms are functors!

Example: "Hom"-functors

- Let $\mathcal{C}$ be a category, and let $X \in \mathrm{Ob}(\mathcal{C})$ be an object. Then we have a functor $\mathrm{Mor}_{\mathcal{C}}(X, \cdot) : \mathcal{C} \to \underline{\mathrm{Set}}$ defined as follows:
- An object $Y \in \mathrm{Ob}(C)$ is sent to the set $\mathrm{Mor}_{\mathcal{C}}(X, Y)$.
- A morphism $f : Y \to Z$ is sent to the function $\mathrm{Mor}_{\mathcal{C}}(X, Y) \to \mathrm{Mor}_{\mathcal{C}}(X, Z)$ given by composition with f: $g \mapsto fg$.
- Such functors are called "Hom-functors", because homomorphism is often used as a synonym for "morphism", especially in "algebraic" categories.

Example: forgetful functors

- Let $\mathcal{C}$ be a category of "sets with extra structure", for example the category $\underline{\mathrm{Grp}}$ of groups.
- Then we have the so-called forgetful functor $\underline{\mathrm{Grp}} \to \underline{\mathrm{Set}}$ which sends a group to its "underlying" set ("forgetting" the group structure) and a group homomorphism to itself, considered as a simple map between sets.
- The functor laws hold trivially.

Example: discrete topology

- According to the previous slide, we have a forgetful functor $\underline{\mathrm{Top}} \to \underline{\mathrm{Set}}$.
- We also have a functor in the opposite direction $\underline{\mathrm{Set}} \to \underline{\mathrm{Top}}$:
- This functor sends a set S to S equipped with the *discrete topology* (i.e. every subset is open) and a map $S \to T$ to itself (which is continuous due to our choice of topology).

Example: polynomial ring

- According to the previous slide, we have a forgetful functor $\underline{\mathrm{Ring}} \to \underline{\mathrm{Set}}$.
- We also have a functor in the opposite direction $\underline{\mathrm{Set}} \to \underline{\mathrm{Ring}}$:
- This functor sends a set S to the *polynomial ring* $\mathbb{Z}[S]$ and a map $f : S \to T$ to the ring homomorphism $\mathbb{Z}[S] \to \mathbb{Z}[T]$ given by

$$\Big(\dots + n s_1 s_2 \dots s_k + \dots\Big) \mapsto \Big(\dots + n f(s_1) f(s_2) \dots f(s_k) + \dots\Big)$$

Example: connected components

- Apart from the forgetful functor, we can define a more interesting functor $\pi_0 : \underline{\mathrm{Top}} \to \underline{\mathrm{Set}}$ as follows:
- Send a topological space to the set of its *connected components* $\pi_0(X)$.
- Due to the fact that a continuous map $f : X \to Y$ maps connected subsets to connected subsets, we get an induced map $\pi_0 : \pi_0(X) \to \pi_0(Y)$.

Example: group homomorphism

- Let $\varphi : G \to H$ be a group homomorphism, and let $\underline{G}$ and $\underline{H}$ be the categories associated to G and H.
- Then φ induces a functor $\underline{\varphi} : \underline{G} \to \underline{H}$ by sending $*$ to $*$ and a morphism g (which is just an element of G!) to $\varphi(g)$.
- The functor laws follow immediately from the properties of a group homomorphism.

Example: monotonic maps

- Let $f : (S, \leq) \to (T, \leq)$ be a monotonic (i.e. order preserving) map between partially ordered sets.
- Then f induces a functor between the associated categories $\underline{f} : \underline{S} \to \underline{T}$ by sending objects $s \in S$ to $f(s) \in T$.
- Seeing as morphism sets in these categories have at most one element, we have no choice for morphisms.
- The fact that f is monotonic implies that we get a well-defined functor in this way.

Example: Haskell functors

- Let `f :: * -> *` be a *Haskell functor* that obeys the Haskell functor laws.
- Then `f` defines a functor (in the categorical sense) $f : \underline{\mathrm{Hask}} \to \underline{\mathrm{Hask}}$ by sending a type `a` to `f a` and a Haskell function `g :: a -> b` to `fmap g :: f a -> f b`.
- The Haskell functor laws imply the categorical functor laws!

Functors respect isomorphisms

Lemma

Let $\mathcal{C}$ and $\mathcal{D}$ be categories, let $F : \mathcal{C} \to \mathcal{D}$ be a functor, and let $f : X \xrightarrow{\sim} Y$ be an isomorphism in $\mathcal{C}$. Then $Ff : FX \to FY$ is an isomorphism in $\mathcal{D}$.

Proof.

We claim that $F(f^-1)$ is an/the inverse of Ff:

$$Ff \circ F(f^{-1}) = F(ff^{-1}) = F1_Y = 1_{FY}$$

and

$$Ff^{-1} \circ Ff = F(f^{-1}f) = F1_X = 1_{FX}.$$

□

Functors respect isomorphisms

Lemma

Let $\mathcal{C}$ and $\mathcal{D}$ be categories, let $F : \mathcal{C} \to \mathcal{D}$ be a functor, and let $f : X \xrightarrow{\sim} Y$ be an isomorphism in $\mathcal{C}$. Then $Ff : FX \to FY$ is an isomorphism in $\mathcal{D}$.

Corollary

Let $\mathcal{C}$ and $\mathcal{D}$ be categories, let $F : \mathcal{C} \to \mathcal{D}$ be a functor, and let X and Y be isomorphic objects of $\mathcal{C}$. Then FX and FY are isomorphic in $\mathcal{D}$.

Importance of functors

- We have seen examples of functors between categories of quite different branches of mathematics.
- So once we have a functor, we can often transfer problems from one area of mathematics to another.

Example

Are the topological spaces $\mathbb{R}$ and $\mathbb{R} \setminus \{0\}$ homeomorphic? No, because $\pi_0(\mathbb{R}) = \{*\}$, but $\pi_0(\mathbb{R} \setminus \{0\}) = \{-, +\}$. By the previous corollary, the two spaces cannot be isomorphic, because the two sets clearly are not. We have reduced a difficult topological problem to simple counting of elements!

Importance of functors

- We have seen examples of functors between categories of quite different branches of mathematics.
- So once we have a functor, we can often transfer problems from one area of mathematics to another.

Outlook

This example is only a first glimpse at the power of functors. Using functors to translate geometric problems into algebraic ones has revolutionized 20th century mathematics – with concepts like (co-)homology and (higher) homotopy groups.

Contravariant functors

- Let $\mathcal{C}$ and $\mathcal{D}$ be categories.
- A contravariant functor $\mathcal{C} \to \mathcal{D}$ is a functor $\mathcal{C} \to \mathcal{D}^{\mathrm{op}}$.
- In concrete terms, that means that a contravariant functor F sends objects $X \in \mathrm{Ob}(\mathcal{C})$ to objects $FX \in \mathrm{Ob}(\mathcal{D})$ and morphisms $f : X \to Y$ to morphisms $Ff : FY \to FX$ in $\mathcal{D}$.
- In case of possible confusion, "normal" functors are also called covariant functors.

Example: contravariant "Hom"-functors

- Let $\mathcal{C}$ be a category, and let $X \in \mathrm{Ob}(\mathcal{C})$ be an object. Then we have a contravariant functor $\mathrm{Mor}_{\mathcal{C}}(\cdot, X) : \mathcal{C} \to \underline{\mathrm{Set}}$ defined as follows:
- An object $Y \in \mathrm{Ob}(C)$ is sent to the set $\mathrm{Mor}_{\mathcal{C}}(Y, X)$.
- A morphism $f : Y \to Z$ is sent to the function $\mathrm{Mor}_{\mathcal{C}}(Z, X) \to \mathrm{Mor}_{\mathcal{C}}(Y, X)$ given by composition with f: $g \mapsto gf$.

Example: contravariant "Hom"-functors

- Let $\mathcal{C}$ be a category, and let $X \in \mathrm{Ob}(\mathcal{C})$ be an object. Then we have a contravariant functor $\mathrm{Mor}_{\mathcal{C}}(\cdot, X) : \mathcal{C} \to \underline{\mathrm{Set}}$ defined as follows:
- An object $Y \in \mathrm{Ob}(C)$ is sent to the set $\mathrm{Mor}_{\mathcal{C}}(Y, X)$.
- A morphism $f : Y \to Z$ is sent to the function $\mathrm{Mor}_{\mathcal{C}}(Z, X) \to \mathrm{Mor}_{\mathcal{C}}(Y, X)$ given by composition with f: $g \mapsto gf$.
- A special – hopefully familiar – case is given by *dual vector spaces*: If k is a field, then we have a contravariant functor $*$ from k-vector spaces to k-vector spaces, which sends a k-vector space V to its dual $V^* := \mathrm{Hom}_k(V, k)$ and a k-linear map $\varphi : V \to W$ to the dual map $\varphi^* : W^* \to V^*$.

Initial- & Final Objects

Initial Object

- Let $\mathcal{C}$ be a category. An object I in $\mathcal{C}$ is called initial object if it has the following universal property: For all objects X in $\mathcal{C}$, there is exactly one morphism $i_X : I \to X$.
- If such an initial object exists, it is *unique up to isomorphism*: Let I' be another initial object. Then

$$i'_I\, i_{I'},\, 1_I \,:\, I \to I \overset{I \text{ initial}}{\Longrightarrow} i'_I\, i_{I'} = 1_I,$$

and similarly $i_{I'}\, i'_I = 1_{I'}$, so $i_{I'} : I \xrightarrow{\sim} I'$.

Initial Object

- Let $\mathcal{C}$ be a category. An object I in $\mathcal{C}$ is called initial object if it has the following universal property: For all objects X in $\mathcal{C}$, there is exactly one morphism $i_X : I \to X$.
- If such an initial object exists, it is *unique up to isomorphism*: Let I' be another initial object. Then

$$i'_I \, i_{I'}, 1_I \;:\; I \to I \overset{I \text{ initial}}{\Longrightarrow} i'_I \, i_{I'} = 1_I,$$

and similarly $i_{I'} \, i'_I = 1_{I'}$, so $i_{I'} : I \xrightarrow{\sim} I'$.

Universal Property

Many objects in category theory are defined via a “universal property” like this. If such an object exists, it is always unique up to (unique) isomorphism.

Final Object

- Let $\mathcal{C}$ be a category. An object T in $\mathcal{C}$ is called final object or terminal object if it is an initial object in $\mathcal{C}^{\mathrm{op}}$.
- Explicitly, T is a final object if is has the *universal property* that for all objects X in $\mathcal{C}$, there is exactly one morphism $t_X : X \to T$.
- If such a final object exists, it is *unique up to isomorphism*.

Example: initial- & final object in $\underline{\mathrm{Set}}$

- The category $\underline{\mathrm{Set}}$ has both an initial and a final object.

Example: initial- & final object in $\underline{\mathrm{Set}}$

- The category $\underline{\mathrm{Set}}$ has both an initial and a final object.
- The initial object is the *empty set* $\varnothing$.

Example: initial- & final object in $\underline{\mathrm{Set}}$

- The category $\underline{\mathrm{Set}}$ has both an initial and a final object.
- The initial object is the *empty set* $\varnothing$.
- The final object is the *singleton set* $\{*\}$.

Example: initial- & final object in $\underline{\mathrm{Set}}$

- The category $\underline{\mathrm{Set}}$ has both an initial and a final object.
- The initial object is the *empty set* $\varnothing$.
- The final object is the *singleton set* $\{*\}$.

Note

Note that while there is only one empty set, there are indeed infinitely many "different" singletons – but they are all isomorphic, and we don't distinguish between them.

Example: initial- & final object in $\underline{\mathrm{Set}}$

- The category $\underline{\mathrm{Set}}$ has both an initial and a final object.
- The initial object is the *empty set* $\varnothing$.
- The final object is the *singleton set* $\{*\}$.

Note

Note that while there is only one empty set, there are indeed infinitely many ”different” singletons – but they are all isomorphic, and we don’t distinguish between them.

Note

Also note that $\varnothing$ is *not* isomorphic to $\{*\}$.

Example: initial- & final object in $\underline{\mathrm{Grp}}$

- The category $\underline{\mathrm{Grp}}$ has both an initial and a final object.

Example: initial- & final object in $\underline{\mathrm{Grp}}$

- The category $\underline{\mathrm{Grp}}$ has both an initial and a final object.
- The initial object is the *one-element group* $\{1\}$.

Example: initial- & final object in $\underline{\mathrm{Grp}}$

- The category $\underline{\mathrm{Grp}}$ has both an initial and a final object.
- The initial object is the *one-element group* $\{1\}$.
- The final object is *also* $\{1\}$.

Example: initial- & final object in $\underline{\mathrm{Grp}}$

- The category $\underline{\mathrm{Grp}}$ has both an initial and a final object.
- The initial object is the *one-element group* $\{1\}$.
- The final object is *also* $\{1\}$.

Note

Note that in $\underline{\mathrm{Grp}}$, the initial object is isomorphic to the final object!

Example: final object in Δ

- The simplex-category Δ has a final object, but *no initial object.*

Example: final object in Δ

- The simplex-category Δ has a final object, but *no initial object*.
- The final object is $[0]$.

Example: initial- & final object in a slice category

- Let $\mathcal{C}$ be a category and $X \in \mathrm{Ob}(\mathcal{C})$ an object. The slice category $\mathcal{C}/X$ has a final object, and if $\mathcal{C}$ has an initial object, then so does $\mathcal{C}/X$.

Example: initial- & final object in a slice category

- Let $\mathcal{C}$ be a category and $X \in \mathrm{Ob}(\mathcal{C})$ an object. The slice category $\mathcal{C}/X$ has a final object, and if $\mathcal{C}$ has an initial object, then so does $\mathcal{C}/X$.
- If I is the initial object of $\mathcal{C}$, then $i_X : I \to X$ is the initial object of $\mathcal{C}/X$.

Example: initial- & final object in a slice category

- Let $\mathcal{C}$ be a category and $X \in \mathrm{Ob}(\mathcal{C})$ an object. The slice category $\mathcal{C}/X$ has a final object, and if $\mathcal{C}$ has an initial object, then so does $\mathcal{C}/X$.
- If I is the initial object of $\mathcal{C}$, then $i_X : I \to X$ is the initial object of $\mathcal{C}/X$.
- The final object of $\mathcal{C}/X$ is $1_X : X \to X$.

Example: initial- & final object in $\underline{\mathrm{Hask}}$

- The category $\underline{\mathrm{Hask}}$ has an initial and a final object.

Example: initial- & final object in $\underline{\mathrm{Hask}}$

- The category $\underline{\mathrm{Hask}}$ has an initial and a final object.
- The initial object is `Void`.

Example: initial- & final object in $\underline{\mathrm{Hask}}$

- The category $\underline{\mathrm{Hask}}$ has an initial and a final object.
- The initial object is `Void`.
- The final object is `()`.

Sums & Products

Sum

- Let $\mathcal{C}$ be a category, and let $X, Y \in \mathrm{Ob}(\mathcal{C})$ be objects. We say that X and Y have a sum (or coproduct) if there is an object $X + Y$ and morphisms $i_X : X \to X + Y$ and $i_Y : Y \to X + Y$ with the following universal property: For all objects T and morphisms $f_X : X \to T$ and $f_Y : Y \to T$, there is exactly one morphism $f : X + Y \to T$ such that the following diagram commutes ($fi_X = f_X \wedge fi_Y = f_Y$):

$$
\begin{tikzcd}
X \arrow[dr, "i_X"] \arrow[drr, bend left, "f_X"] & & \\
& X + Y \arrow[r, red, "\exists! f"] & T \\
Y \arrow[ur, "i_Y"'] \arrow[urr, bend right, "f_Y"'] & &
\end{tikzcd}
$$

- If all pairs of objects in $\mathcal{C}$ have a sum, we say that $\mathcal{C}$ has (finite) sums.

Product

- Let $\mathcal{C}$ be a category, and let $X, Y \in \mathrm{Ob}(\mathcal{C})$ be objects. We say that X and Y have a product if there is an object $X * Y$ and morphisms $p_X : X * Y \to X$ and $p_Y : X * Y \to Y$ with the following universal property: For all objects T and morphisms $f_X : T \to X$ and $f_Y : T \to Y$, there is exactly one morphism $f : T \to X * Y$ such that the following diagram commutes ($p_X f = f_X \wedge p_Y f = f_Y$):

$$
\begin{tikzcd}
 & & X \\
T \arrow[r, "\exists! f"] \arrow[urr, bend left, "f_X"] \arrow[drr, bend right, "f_Y"'] & X * Y \arrow[ur, "p_X"] \arrow[dr, "p_Y"'] & \\
 & & Y
\end{tikzcd}
$$

- If all pairs of objects in $\mathcal{C}$ have a product, we say that $\mathcal{C}$ has (finite) products.

Example: sums and products in $\underline{\mathrm{Set}}$

- The category $\underline{\mathrm{Set}}$ has both sums and products.

Example: sums and products in $\underline{\mathrm{Set}}$

- The category $\underline{\mathrm{Set}}$ has both sums and products.
- The sum of two sets S and T is the *disjoint union* $S \uplus T$.

Example: sums and products in $\underline{\mathrm{Set}}$

- The category $\underline{\mathrm{Set}}$ has both sums and products.
- The sum of two sets S and T is the *disjoint union* $S \uplus T$.
- The product of two sets S and T is the *cartesian product* $S \times T$.

Example: sums and products in $\underline{\mathrm{Set}}$

- The category $\underline{\mathrm{Set}}$ has both sums and products.
- The sum of two sets S and T is the *disjoint union* $S \uplus T$.
- The product of two sets S and T is the *cartesian product* $S \times T$.

Note

Note that for most pairs of sets S and T, $S \uplus T$ and $S \times T$ are *not* isomorphic.

Example: sums & products in $\underline{\mathrm{Ab}}$

- The category $\underline{\mathrm{Ab}}$ of abelian groups has both sums and products.

Example: sums & products in $\underline{\mathrm{Ab}}$

- The category $\underline{\mathrm{Ab}}$ of abelian groups has both sums and products.
- The sum of two groups A and B is the *direct sum* $A \oplus B$ (with injections $a \mapsto (a, 0)$ and $b \mapsto (0, b)$).

Example: sums & products in $\underline{\mathrm{Ab}}$

- The category $\underline{\mathrm{Ab}}$ of abelian groups has both sums and products.
- The sum of two groups A and B is the *direct sum* $A \oplus B$ (with injections $a \mapsto (a, 0)$ and $b \mapsto (0, b)$).
- The product of two groups A and B is *also* $A \oplus B$ (with projections $(a, b) \mapsto a$ and $(a, b) \mapsto b$).

Example: sums & products for paritally ordered sets

- Let $(S, \leq)$ be a partially ordered sets, and let $x, y \in S$.

Example: sums & products for paritally ordered sets

- Let $(S, \leq)$ be a partially ordered sets, and let $x, y \in S$.
- The sum of x and y in $\underline{S}$ – if it exists – is the *least upper bound* of x and y.

Example: sums & products for paritally ordered sets

- Let $(S, \leq)$ be a partially ordered sets, and let $x, y \in S$.
- The sum of x and y in $\underline{S}$ – if it exists – is the *least upper bound* of x and y.
- The product of x and y in $\underline{S}$ – if it exists – is the *greatest lower bound* of x and y.

Example: sums & products in $\underline{\mathrm{Hask}}$

- The category $\underline{\mathrm{Hask}}$ has sums and products.

Example: sums & products in $\underline{\mathrm{Hask}}$

- The category $\underline{\mathrm{Hask}}$ has sums and products.
- The sum of types `a` and `b` is `Either a b` (with injections `Left` and `Right`).

Example: sums & products in $\underline{\mathrm{Hask}}$

- The category $\underline{\mathrm{Hask}}$ has sums and products.
- The sum of types `a` and `b` is `Either a b` (with injections `Left` and `Right`).
- The product of types `a` and `b` is `(a, b)` (with projections `fst` and `snd`).

Natural Transformations

Natural transformation

- Let $F, G : \mathcal{C} \to \mathcal{D}$ be functors. A natural transformation $\varphi : F \to G$ is given by the following data:
- For each object $X \in \mathrm{Ob}(\mathcal{C})$, a morphism $\varphi_X : FX \to GX$ in $\mathcal{D}$.
- For each morphism $f : X \to Y$ in $\mathcal{C}$, the following diagram must commute (i.e. $\varphi_Y\, Ff = Gf\, \varphi_X$):

$$\begin{tikzcd} X \arrow[d, dashed, "f"'] \\ Y \end{tikzcd} \qquad \begin{tikzcd} FX \arrow[r, "\varphi_X"] \arrow[d, "Ff"'] & GX \arrow[d, "Gf"] \\ FY \arrow[r, "\varphi_Y"'] & GY \end{tikzcd}$$

Example: Hom-functors

Let $\mathcal{C}$ be a category, and let $g : U \to V$ be a morphism. Then g induces a natural transformation $f^* : \mathrm{Mor}_{\mathcal{C}}(V, \cdot) \to \mathrm{Mor}_{\mathcal{C}}(U, \cdot)$, given by composition with g:

$$\begin{tikzcd}
X \arrow[d, dashed, "f"'] & \mathrm{Mor}_{\mathcal{C}}(V, X) \arrow[r, "f_X^*"] \arrow[d, "\mathrm{Mor}_{\mathcal{C}}(V,\cdot)f"'] & \mathrm{Mor}_{\mathcal{C}}(U, X) \arrow[d, "\mathrm{Mor}_{\mathcal{C}}(U,\cdot)f"] \\
Y & \mathrm{Mor}_{\mathcal{C}}(V, Y) \arrow[r, "f_Y^*"'] & \mathrm{Mor}_{\mathcal{C}}(U, Y)
\end{tikzcd}$$

Example: determinant

- Let n be a natural number. Sending a commutative ring R to the group $\mathrm{GL}_n(R)$ of invertible $n \times n$-matrices defines a functor $\mathrm{GL}_n : \underline{\mathrm{Ring}} \to \underline{\mathrm{Grp}}$ (for a ring homomorphism $\alpha : R \to S$, we get an induced group homomorphism $\mathrm{GL}_n(\alpha) : \mathrm{GL}_n(R) \to \mathrm{GL}_n(S)$ by applying α to each matrix element).
- We get another functor $* : \underline{\mathrm{Ring}} \to \underline{\mathrm{Grp}}$ by sending a ring R to its *group of units* (i.e. invertible elements) R^*.
- Then the *determinant* $\mathsf{det} : \mathrm{GL}_n(R) \to R^*$ defines a natural transformation $\mathsf{det} : \mathrm{GL}_n \to *$:

$$
\begin{tikzcd}
R \arrow[d, dashed, "\alpha"'] & \mathrm{GL}_n(R) \arrow[r, "\mathsf{det}"] \arrow[d, "\mathrm{GL}_n(\alpha)"'] & R^* \arrow[d, "\alpha|_{R^*}"] \\
S & \mathrm{GL}_n(S) \arrow[r, "\mathsf{det}"'] & S^*
\end{tikzcd}
$$

Example: polymorphic Haskell functions

Let `F` and `G` be Haskell functors, and let `g :: F a -> G a` be a polymorphic function. Then `g` defines a natural transformation $F \to G$ in $\underline{\mathbf{Hask}}$ by type specialization.

For example, consider `F = Maybe`, `G = []` and

```
g :: Maybe a -> [a]
g Nothing = []
g (Just x) = [x]
```

```
GHCi> g $ fmap show $ Just True
["True"]
GHCi> fmap show $ g $ Just True
["True"]
```

Example: polymorphic Haskell functions

Let `F` and `G` be Haskell functors, and let `g :: F a -> G a` be a polymorphic function. Then `g` defines a natural transformation $F \to G$ in $\underline{\mathrm{Hask}}$ by type specialization.

$$\begin{tikzcd} \texttt{Bool} \arrow[d, dashed, "\texttt{show}"'] \\ \texttt{String} \end{tikzcd}
\qquad
\begin{tikzcd} \texttt{Maybe Bool} \arrow[r, "\texttt{g}"] \arrow[d, "\texttt{fmap show}"'] & \texttt{[Bool]} \arrow[d, "\texttt{fmap show}"] \\ \texttt{Maybe String} \arrow[r, "\texttt{g}"'] & \texttt{[String]} \end{tikzcd}$$