3. Universality and Representability

Let $\mathcal{C}$ be a category. We now identify some special, or *universal*, objects of $\mathcal{C}$.

Definition 3.1. We say 0 is the *initial object* of $\mathcal{C}$ if for any $\mathcal{C}$-object—including itself—X, there exists a unique arrow $0 \to X$. Dually, we say 1 is the *terminal object* of $\mathcal{C}$ if for any $\mathcal{C}$-object X, there exists a unique arrow $X \to 1$.

By duality, an initial object in $\mathcal{C}$ is a terminal object in $\mathcal{C}^{\mathrm{op}}$ and vice versa. When $\mathcal{C}$ is thin, i.e. a preorder, the bottom $\bot$ is the initial object and top $\top$ is the terminal object. From this observation, we remark that not all categories have initial or terminal objects—for example, the poset $(\mathbb{Z}, \leq)$, thought of as a thin category, has neither top nor bottom and hence neither initial nor terminal object. We often denote the unique arrow guaranteed by universality via a dashed line:

$$0 \dashrightarrow S$$

The use of the symbols 0 and 1 for generic initial and terminal objects is inspired by the fact that $\mathbf{0} = \varnothing$ and $\mathbf{1} \cong \star$ are respectively the initial and terminal objects in **Set**. The latter fact is easier to see: given a set S, a map $S \to \star$ consists of a choice of element in $\star$ for every $s \in S$—but there can be only one such choice: the sole element $* \in \star$. In turn, one can conceptualize the fact that $\varnothing$ is the initial object in **Set** by considering the fact that a map $\varnothing \to S$ consists of a choice for every element in $\varnothing$—but there are no such elements, and hence no choice needs to be made. Said otherwise, each element of the domain poses a challenge: to choose a single codomain element associated with it. In the case that there are no such challenges, we say—via some combination of philosophy and convention—that there is hence precisely one map that is gifted to us by a lack of choice in the matter. This is in some sense analogous to the situation in which we define nullary operators to be units.

In the context of **Pre**, the initial and terminal objects are, just as in **Set**, $\varnothing$ and $\star$, equipped with the unique ordering structure on them. This turns out to be no coincidence: **Pre**—and similarly **Pos**—inherit much of their "universal structure" from **Set**. This will be elaborated further in the subsequent section.

In both **Mon** and $\mathbf{Vect}_k$ we have a so called *zero object*—i.e. an object that is both terminal and initial. These are respectively given by the one element monoid $\{e\}$ and the zero vector space $\{0\}$. At the risk of mild notational overload, we will denote both of these by 0. These are terminal objects for the same reason that $\star$ is a terminal object in $\mathcal{C}$—every homormorphism to the one element structure maps the entire codomain to that element. Just as above, this arises from the fact that half of the universal structure of **Set** is inherited by **Mon** and $\mathbf{Vect}_k$. In turn, however, the initial object is not the empty set! This is because both monoids and vector spaces require unit elements. The arrows from these one element structures to any other structure are unique by virtue of the fact that these arrows by definition must preserve unit elements, and hence uniquely send the sole domain element to the unit of the codomain.

We now present the first special property of universal objects.

Proposition 3.2. *Let* 0 *and* $0'$ *be initial objects of* $\mathcal{C}$*. Then* $0 \cong 0'$*.*

Proof. By initiality, there are unique arrows $0 \to 0$ and $0' \to 0'$, which must be the identity arrows $\mathbb{1}_0$ and $\mathbb{1}_{0'}$. We also have unique arrows $\varphi : 0 \to 0'$ and $\varphi' : 0' \to 0$.

This yields arrows $\varphi /\!\!/ \varphi' : 0 \to 0$ and $\varphi' /\!\!/ \varphi : 0' \to 0$, which, by uniqeueness, must be the respective identity arrows. □

By duality, the same fact holds for terminal objects. In fact, we will see throughout the section that any universal construction can be cast as an initial object in some category, and hence enjoys the result of this theorem.

As the title suggests, we will also consider a second approach, called representability, towards the study of categories. In particular, we say a functor $F : \mathcal{C} \to \mathbf{Set}$ is *representable* if there is a natural isomorphism $F \cong \mathcal{C}(c, -)$ for some $\mathcal{C}$-object c. Dually, we say a functor $F : \mathcal{C}^{\mathrm{op}} \to \mathbf{Set}$ is representable if there is a natural isomorphism $F \cong \mathcal{C}(-, c)$ for some $\mathcal{C}$-object c. In both of these cases, we say that c *represents* the relevant functor. In turn, the functor that an object represents will often provide a semantic interpretation for the object. A particularly important example of reprsentable functor is the identity functor $\mathbb{1}_{\mathbf{Set}} : \mathbf{Set} \to \mathbf{Set}$. Recalling that an element $s \in S$ is equivalent to the arrow $\lambda * .s : \star \to S$, we have that $S \cong \mathbf{Set}(\star, S)$, and hence that $\mathbb{1}_{\mathbf{Set}} \cong \mathbf{Set}(\star, -)$.

Universal constructions always have an interpretation in the language of representability. In the case of initial and terminal objects, this may seem a little dull. In particular, consider the constant functor $\mathcal{C} \to \mathbf{Set}$ that sends every object to $\star$ and every arrow to $\mathbb{1}_\star$. If $\mathcal{C}$ has an initial object 0, then this functor is naturally isomorphic to $\mathcal{C}(0, -)$ since by definition the set of arrows $0 \to X$ for any X is a singleton. Dually, the constant functor $\mathcal{C}^{\mathrm{op}} \to \mathbf{Set}$ with the same behavior is naturally isomorphic to $\mathcal{C}(-, 1)$. The contravariance, although not necessary to the well definedness of this particular functor, is stiuplated to respect the fact that $\mathcal{C}(-, c)$ is generically contravariant.

Representable functors are special because of the *Yoneda Lemma*, the most famous result in category theory.

Theorem 3.3. *Let $\mathcal{C}$ be a category, $c \in |\mathcal{C}|$, and $F : \mathcal{C} \to \mathbf{Set}$. Then there is a natural bijection*

$$\mathbf{Cat}(\mathcal{C}(c, -), F) \cong Fc$$

given by mapping a natural transformation $T : \mathcal{C}(c, -) \to F$ to $T_c(\mathbb{1}_c) \in Fc$.

Proof. Let $\Phi : \mathbf{Cat}(\mathcal{C}(c, -), F) \to Fc :: T \mapsto T(\mathbb{1}_c)$. We will now construct a map $\Psi : F(c) \to \mathbf{Cat}(\mathcal{C}(c, -), F)$ that satisfies $\Phi \circ \Psi = \mathbb{1}_{Fc}$ and $\Psi \circ \Phi = \mathbb{1}_{\mathbf{Cat}(\mathcal{C}(c,-),F)}$.

We define Ψ to be a valid inverse by mapping an element x of Fc to a natural transformation T for which $T_c(\mathbb{1}_c) = x$. We now show that this uniquely defines a natural transformation. Since T consists of component maps $T_{c'} : \mathcal{C}(c, c') \to Fc$ for all c', it suffices to define $T_{c'}(f)$ for all $f \in \mathcal{C}(c, c')$. Then, letting $f \in \mathcal{C}(c, c')$, the naturality condition states that the following square must commute.

$$\begin{array}{ccc} \mathcal{C}(c,c) & \xrightarrow{T_c} & Fc \\ {\scriptstyle -/\!\!/ f}\downarrow & & \downarrow{\scriptstyle Ff} \\ \mathcal{C}(c,c') & \xrightarrow[T_{c'}]{} & Fc' \end{array}$$

Tracing the two paths of the identity arrow $\mathbb{1}_c \in \mathcal{C}(c, c)$, we have:

$$\to\downarrow : \mathbb{1}_c \mapsto T_c(\mathbb{1}_c) = x \mapsto (Ff)x$$
$$\downarrow\to : \mathbb{1}_c \mapsto f \mapsto T_{c'}(f)$$

This then forces us to define $T_{c'}(f) = (Ff)x$. □

The reader is encouraged to check that this bijection is natural in c. Furthermore, this bijection is natural in $\mathcal{C}$ and F, but this involves slightly more mental and notational contortion to articulate. Note that, were we to apply the Yoneda Lemma to $\mathcal{C}^{\mathrm{op}}$, we would have that for any $F : \mathcal{C}^{\mathrm{op}} \to \mathbf{Set}$ and $c \in |\mathcal{C}|$:

$$\mathbf{Cat}(\mathcal{C}^{\mathrm{op}}(c, -), F) = \mathbf{Cat}(\mathcal{C}(-, c), F) \cong Fc.$$

The Yoneda Lemma can be used to answer questions like "what are the natural maps $X \to \mathbf{Set}(S, X)$?" Said both more suggestively and formally: what are the natural transformations $\mathbb{1}_{\mathbf{Set}} \cong \mathbf{Set}(\star, -) \to \mathbf{Set}(S, -)$? The Yoneda Lemma states that these would be in natural bijection with elements of the set $\mathbf{Set}(S, \star)$, which, by the terminality of $\star$, form a singleton given by the constant map. Hence the only natural transformation $T : \mathbb{1}_{\mathbf{Set}} \to \mathbf{Set}(S, -)$ would be defined by extending $T_\star : \star \to \mathbf{Set}(S, \star) :: \mathbb{1}_\star \mapsto \lambda s.* : S$. Hence, applying the bijection given by the Yoneda Lemma, we have, for a set X and X-element x, i.e. arrow $x : \star \to X$, that:

$$T_X(x) = \mathbf{Set}(S, x) : \mathbf{Set}(S, \star) \to \mathbf{Set}(S, X) :: f \mapsto x.$$

Said more plainly, a natural map $\mathbf{Set}(\star, X) \to \mathbf{Set}(S, X)$ is just a precomposition with the map $x : \star \to S$, i.e. a map that sends the element $x \in X$ to the constant map $\lambda s.x : S \to X$. This then allows us to conclude that the only natural map—i.e. the only map that can be defined generically, with no reference to the sets involved—$X \to \mathbf{Set}(S, X)$ is the map that takes elements $x \in X$ to constant maps $S \to X$ that send each S-element to x.

This example involved the case that the functor F was itself representable. In this context, the Yoneda Lemma has another meaningful interpretation. Define the functor $\mathcal{Y}$, called the *Yoneda embedding*, of type $\mathcal{C}^{\mathrm{op}} \to \mathbf{Cat}(\mathcal{C}, \mathbf{Set})$ as follows. For an object c, define $\mathcal{Y}c = \mathcal{C}(c, -)$. For an arrow $\varphi : c \to c'$, define the natural transformation $\mathcal{Y}\varphi : \mathcal{C}(c', -) \to \mathcal{C}(c, -)$ via the components

$$[\mathcal{Y}\varphi]_X : \mathcal{C}(c', X) \to \mathcal{C}(c, X) :: f \mapsto \varphi \mathbin{/\!\!/} f.$$

Then, the Yoneda lemma states that

$$\mathbf{Cat}(\mathcal{Y}c', \mathcal{Y}c) = \mathbf{Cat}(\mathcal{C}(c', -), \mathcal{C}(c, -)) \cong \mathcal{C}(c, c').$$

This can be interpreted as the statement that the Yoneda embedding $\mathcal{Y}$ is fully faithful; i.e. that $\mathcal{Y}_{c',c} : \mathcal{C}^{\mathrm{op}}(c', c) = \mathcal{C}(c, c') \to \mathbf{Cat}(\mathcal{Y}c', \mathcal{Y}c)$ is always a bijection. Recall that this means that $\mathcal{Y}$ has the property that if $\mathcal{Y}c \cong \mathcal{Y}c'$, then $c \cong c'$. This has the deep implication that an object is determined up to isomorphism by the functor it represents. This truth is in some sense why category theory contributes insight to mathematics beyond mere bookkeeping: rather, it allows one to speak of mathematical structures purely in terms of their relationships to all other structures of their kind, as opposed to relying on what they're "made of." As an example of this principle, consider the case of initial objects: since these always represent the constant set-valued functor of value $\star$, they must all be isomorphic.

Let's now consider a more interesting dual pair of universal constructions. We motivate these with a case study in the context of $\mathbf{Set}$. Consider a map $\varphi : S \to X \times Y$; for example, perhaps S consists of the set of points in some island, X and Y respectively denote the sets of possible values for temperature and pressure, and φ assigns to each point the ordered pair of air temperature and pressure at the altitude of an average human height. We can easily extract from this map two maps φ_X and φ_Y, which respectively report the temperature and the pressure. Intuitively, φ_X merely forgets the pressure data and φ_Y merely forgets the temperature data.

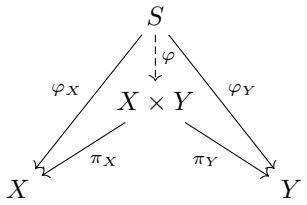
More formally, we can define $\varphi_X = \varphi /\!\!/ \pi_X$, where we call π_X the *projection* map, and define it by the rule $(x, y) \mapsto x$. Doing the same for Y, we note the following:

$$\varphi s = (\varphi_X(s), \varphi_Y(s))$$

Said otherwise, from a map $\varphi \in \mathbf{Set}(S, X \times Y)$, we can extract the pair of maps $(\varphi /\!\!/ \pi_X, \varphi /\!\!/ \pi_Y) \in \mathbf{Set}(S, X) \times \mathbf{Set}(S, Y)$, and, from a pair of maps $(\varphi_X, \varphi_Y) \in \mathbf{Set}(S, X) \times \mathbf{Set}(S, Y)$, we can extract the map $\lambda s.(\varphi_X s, \varphi_Y s) \in \mathbf{Set}(S, X \times Y)$. Furthermore, this correspondence yields a bijection

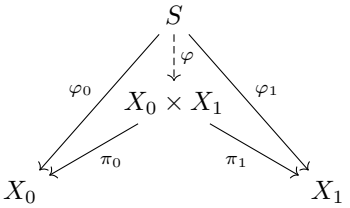
$$\mathbf{Set}(S, X \times Y) \cong \mathbf{Set}(S, X) \times \mathbf{Set}(S, Y).$$

Diagrammatically, this is often expressed as follows:



The dashed arrow is read as follows: given a pair $(\varphi_X : S \to X, \varphi_Y : S \to Y)$, there exists a unique arrow $\varphi : S \to X \times Y$ making the diagram commute. This diagram also states that—following the existence of compositions—that, given a φ, composing it with the projections yields a pair $(\varphi_X : S \to X, \varphi_Y : S \to Y)$. The bijection and the diagram respectively characterize the universality and representability features of the Cartesian Product.

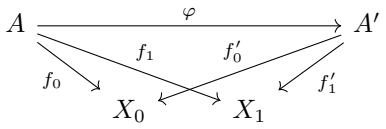
More generally, given a category $\mathcal{C}$ and two objects X_i for $i = 0, 1$, the *product* $X_0 \times X_1$, if it exists, is an object equipped with projection arrows $\pi_i : X_0 \times X_1 \to X_i$ for $i = 0, 1$, such that, given another object S and a pair of arrows $\varphi_i : S \to X_i$ for $i = 0, 1$, there exists a unique arrow $\varphi : S \to X_0 \times X_1$ making the following diagram in $\mathcal{C}$ commute:



The product $X_0 \times X_1$ then satisfies the natural bijection:

$$\mathcal{C}(-, X_0 \times X_1) \cong \mathcal{C}(-, X_0) \times \mathcal{C}(-, X_1).$$

We can encode this even more tightly as a terminal object or a representable functor. Letting X_0, X_1 be objects of $\mathcal{C}$, define the category $\mathcal{C}/(X_0, X_1)$ as having objects *cones*, i.e. triples (A, f_0, f_1) where A is a $\mathcal{C}$-object and $f_i : A \to X_i$ for $i = 0, 1$ are $\mathcal{C}$-arrows. We call A the *apex* and the f_i's the *legs*. Arrows $(A, f_0, f_1) \to (A', f_0', f_1')$ are simply $\mathcal{C}$-arrows $\varphi : A \to A'$ for which the following diagram commutes:



The triple $(X_0 \times X_1, \pi_0, \pi_1)$ is then the terminal object in $\mathcal{C}/(X_0, X_1)$. This merely reformulates the diagram we used to define products since there $(S, \varphi_0, \varphi_1)$ was a cone and we stipulated that it had a unique cone arrow to $(X_0 \times X_1, \pi_0, \pi_1)$—this is precisely what it meant to be a terminal object! Since they are terminal objects, we automatically have that products are unique up to isomorphism—where this isomorphism is itself the unique one that consists of arrows that commute with projections. The cone constructions also allows us to codify the representability perspective of the product. More precisely, let $\mathrm{Cone}_{(X_0,X_1)} : \mathcal{C}^{\mathrm{op}} \to \mathbf{Set}$ map a $\mathcal{C}$-object A to the set of cones, i.e. objects of $\mathcal{C}/(X_0, X_1)$, with apex A. Furthermore, it maps a $\mathcal{C}$-arrow $\varphi : A \to A'$ to the set map $(A', f_0, f_1) \mapsto (A, \varphi /\!\!/ f_0, \varphi /\!\!/ f_1)$. We then have that this functor is represented by the product $X_0 \times X_1$; i.e. we have the following natural isomorphism:

$$\mathrm{Cone}_{(X_0,X_1)} \cong \mathcal{C}(-, X_0 \times X_1)$$

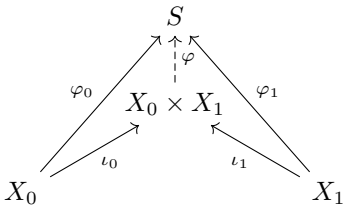
This is the case, since, as our definition established: a cone over X_0, X_1 is naturally equivalent to an arrow to the product $X_0 \times X_1$.

We now consider the dual case: consider a set map $\psi : R + S \to X$, e.g. if R and S represent the set of points on two different islands—and hence their disjoint union $R + S$ represents the set of points across both islands, X the set of possible temperatures, and ψ a map assigning a point its temperature. Dually to the previous example, we can split this up into a pair $(\psi_R : R \to X, \psi_S : S \to X)$, corresponding to the temperature maps for each individual island. Just as before, this yields a natural bijection:

$$\mathbf{Set}(R + S, X) \cong \mathbf{Set}(R, X) \times \mathbf{Set}(S, X).$$

Just as composing with the projection maps allows for decomposing maps into a product, precomposing with so called *inclusion maps* allows for decomposing maps out of a disjoint union. More specifically the inclusion map $\iota_R : R \to R + S$ is given by the inert rule $r \mapsto r$, which merely changes the ambient setting within which the element is considered. Then we have that $\psi_R = \iota_R /\!\!/ \psi$, and similarly for S.

More generally, given a category $\mathcal{C}$ and two objects X_i for $i = 0, 1$, the *coproduct* or *sum* $X_0 + X_1$, if it exists, is merely the product in $\mathcal{C}^{\mathrm{op}}$. Thus it is an object equipped with inclusion arrows $\iota_i : X_i \to X_0 + X_1$ for $i = 0, 1$, such that, given another object S and a pair of arrows $\varphi_i : X_i \to S$ for $i = 0, 1$, there exists a unique arrow $\varphi : X_0 + X_1 \to S$ making the following diagram in $\mathcal{C}$ commute:

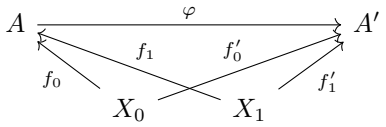


The coproduct $X_0 + X_1$ then satisfies the natural bijection:

$$\mathcal{C}(X_0 + X_1, -) \cong \mathcal{C}(X_0, -) \times \mathcal{C}(X_1, -).$$

We can encode this even more tightly as an initial object or a representable functor. Letting X_0, X_1 be objects of $\mathcal{C}$, define the category $(X_0, X_1)/\mathcal{C}$ as having objects *cocones*, i.e. triples (A, f_0, f_1) where A is a $\mathcal{C}$-object and $f_i : X_i \to A$ for $i = 0, 1$ are $\mathcal{C}$-arrows. We still call A the apex (but, if you're a stickler for symmetry, feel free to

call it the *nadir*) and the ι_i's the legs (or *arms*). Arrows $(A, f_0, f_1) \to (A', f_0', f_1')$ are simply $\mathcal{C}$-arrows $\varphi : A \to A'$ for which the following diagram commutes:



The triple $(X_0 + X_1, \iota_0, \iota_1)$ is then the initial object in $(X_0, X_1)/\mathcal{C}$. This merely reformulates the diagram we used to define coproducts since there $(S, \varphi_0, \varphi_1)$ was a cocone and we stipulated that it had a unique cocone arrow from $(X_0 + X_1, \iota_0, \iota_1)$—this is precisely what it meant to be an initial object! Again, we automatically have that coproducts are unique up to canonical isomorphism. The cocone constructions also allows us to codify the representability perspective of the coproduct. More precisely, let $\mathrm{Cocone}_{(X_0,X_1)} : \mathcal{C} \to \mathbf{Set}$ map a $\mathcal{C}$-object A to the set of cocones, i.e. objects of $(X_0, X_1)/\mathcal{C}$, with apex A. Furthermore, it maps a $\mathcal{C}$-arrow $\varphi : A \to A'$ to the set map $(A, f_0, f_1) \mapsto (A', f_0 /\!\!/ \varphi, f_1 /\!\!/ \varphi)$. We then have that this functor is represented by the coproduct $X_0 + X_1$; i.e. we have the following isomorphism:

$$\mathrm{Cocone}_{(X_0,X_1)} \cong \mathcal{C}(X_0 + X_1, -)$$

This is the case, since, as our definition established: a cocone under X_0, X_1 is naturally equivalent to an arrow from the coproduct $X_0 + X_1$.

We note that, in the context of a thin category, i.e. preorder, $\overline{P}$ we had that the meet $x \wedge y$ satisfied, in arrow notation:

$$t \to x \wedge y \Leftrightarrow t \to x, t \to y,$$

I.e. we have the natural bijection

$$\overline{P}(t, x \wedge y) \cong \overline{P}(t, x) \times \overline{P}(t, y)$$

Hence the meet is a just a product, and, by duality, a join is just a coproduct.

Now, zooming out, let's consider products and coproducts in **Pre**. Recall that **Pre** inherited its initial and terminal object from **Set** for deep reasons we put off until the next section. For the same reason, this will also turn out to hold for products and coproducts! Our work isn't necessarily finished, though: it may be the case, as we'll show, that the underlying set of the product of two preorders is the Cartesian product of their underlying sets, but we still need to describe what the ordering is on this set—and the same for coproducts. This wasn't necessary before, simply because the initial object $\varnothing$ and terminal object $\star$ had a unique ordering. We start with the coproduct because it is more simple. More precisely, given preorders $(P_i, \preceq_i)$ for $i = 0, 1$, we define their coproduct $(P_0 + P_1, \preceq)$ by

$$\preceq (p, q) = \begin{cases} \preceq_i (p, q) & p, q \in P_i \\ \bot & \text{otherwise} \end{cases}$$

Said otherwise, if p and q belong to the same summand, then we merely relate them as we did before; otherwise, we do not relate them. This automatically makes the set inclusion maps $\iota_i : P_i \to P_0 + P_1$ monotonic since $p \preceq_i q \Rightarrow \iota_i(p) \preceq \iota_i(q)$. Furthermore, given a pair of monotone maps $\varphi_i : (P_i, \preceq_i) \to (Q, \sqsubseteq)$, this automatically assembled into a monotone map

$$\varphi_0 + \varphi_1 : (P_0 + P_1, \preceq) \to (Q, \sqsubseteq)$$

defined by $p \mapsto \varphi_i(p)$ for $p \in P_i$. This is automatically monotone since we needn't concern ourselves with preserving cross-summand relations! This vacuousness of satisfying property is at the heart of a universal construction. We now define the product to satisfy the dual condition: i.e. component maps should be vacuously combinable into a monotone map to the product. Towards this end, given preorders $(P_i, \preceq_i)$ for $i = 0, 1$, we define their product $(P_0 \times P_1, \preceq)$ as

$$\preceq ((p_0, q_0), (p_1, q_1)) = \begin{cases} \top & [p_0 \preceq_0 q_0] \wedge [p_1 \preceq_1 q_1] \\ \bot & \text{otherwise} \end{cases}$$

Said otherwise, two ordered pairs (p_i, q_i) are related if and only if both of their respective components had been related. This allows projections to be vacuously monotone since $(p_0, p_1) \preceq (q_0, q_1) \Rightarrow p_i \preceq_i q_i$; and, furthermore, given monotone maps $\varphi_i : (Q, \sqsubseteq) \to (P_i, \preceq_i)$, this automatically assembles into a monotone map

$$\varphi_0 \times \varphi_1 : (Q, \sqsubseteq) \to (P_0 \times P_1, \preceq) :: q \mapsto (\varphi_0 q, \varphi_1 q)$$

since $q \sqsubseteq q' \Rightarrow \forall i : \varphi_i(q) \preceq_i \varphi_i(q') \Rightarrow \varphi(q) \preceq \varphi(q')$.

We now use these constructions to investigate the products and coproducts in the subcategories **Pos** and **Tot**. Before exploring them, we note that, if these categories have products or coproducts, then these must be the same as those constructed in **Pre**. This property is often referred to as *reflection* of products and coproducts, and is due to the following result.

Proposition 3.4. *Fully faithful functors reflect products. Formally, let $F : \mathcal{C} \to \mathcal{D}$ be a fully faithful functor, d_0, d_1 be $\mathcal{D}$-objects with product $d_0 \times d_1$, and c_0, c_1 be $\mathcal{C}$-objects for which $Fc_i = d_i$. Then, if $c_0 \times c_1$ exists, $F(c_0 \times c_1) = d_0 \times d_1$.*

Proof. Let $c \in |\mathcal{C}|$. Then consider the sequence of isomorphisms

$$\begin{aligned} \mathcal{D}(Fc, d_0 \times d_1) &\cong \mathcal{D}(Fc, d_0) \times \mathcal{D}(Fc, d_1) \\ &\cong \mathcal{C}(c, c_0) \times \mathcal{C}(c, c_1) \\ &\cong \mathcal{C}(c, c_0 \times c_1) \\ &\cong \mathcal{D}(Fc, F(c_0 \times c_1)) \end{aligned}$$

Since this did not rely on c, this gives the isomorphism of representable functors:

$$\mathcal{D}(-, d_0 \times d_1) \cong \mathcal{D}(-, F(c_0 \times c_1))$$

The full-faithfuness of the Yoneda embedding then implies the desired isomorphism

$$F(c_0 \times c_1) \cong d_0 \times d_1. \qquad \square$$

Since full-faithfulness does not rely on arrow orientation, duality gives us that fully faithful functors also reflect coproducts. Thus, to study products and coproducts in **Pos** and **Tot**, it suffices to check that the products and coproducts defined in **Pre** preserve posetality and totality of their components.

In the case of **Pos** both products and coproducts defined in **Pre** turn out to be products and coproducts in **Pos**. To see this, we check that if $(P_i, \preceq_i)$ are both posetal, then their product and coproduct is also posetal. Recall that posetality merely requires the antisymmetry condition: $x \preceq y, y \preceq x \Leftrightarrow x = y$. Since coproducts do not add any cross-summand relations, we receive no candidate pairs $p_i \in P_i$ for which $p_0 \preceq p_1$ or vice versa, and hence we automatically have that the preorder coproduct is also a poset. Now consider the product $(P_0 \times P_1, \preceq)$ and suppose

$(p_0, p_1) \preceq (q_0, q_1)$ and $(q_0, q_1) \preceq (p_0, p_1)$ and hence that $p_i \preceq_i q_i$ and $q_i \preceq_i p_i$, thus, by the posetality of the factors, forcing $p_i = q_i$. Therefore **Pos** inherits both its product and coproduct from **Pre**.

It is easy to see, in contrast, that the same doesn't hold for **Tot**. Since total orders require that for any pair p, q either $p \preceq q$ or $q \preceq p$, the coproduct of non-empty total orders is never a total order since we have no relations across summands. The same goes for products, and this is perhaps best exemplified with the example of $(\mathbb{Z} \times \mathbb{Z}, \leq)$—there's no way to compare $(3, 5)$ and $(5, 3)$, and hence products of total orders are typically not total.

Just as it did its terminal object, the category **Mon** inherits products from **Set**. More precisely, given monoids $(M_i, \odot_i, e_i)$ for $i = 0, 1$, we define their product as $(M_0 \times M_1, \odot, (e_0, e_1))$, where $\odot$ is defined component-wise:

$$(m_0, m_1) \odot (n_0, n_1) = (m_0 \odot_0 n_0, m_1 \odot_1 n_1).$$

It is a straightforward exercise to check that this satisfies the monoid axioms. The projection maps preserve the operation since

$$\pi_i((m_0, m_1) \odot (n_0, n_1)) = \pi_i(m_0 \odot_0 n_0, m_1 \odot_1 n_1) = m_i \odot_i n_i.$$

Similarly, two monoid homomorphisms $\varphi_i : N \to M_i$ vacuously assemble into a monoid homormorphism φ defined by $n \mapsto (\varphi_0 n, \varphi_1 n)$ since

$$\begin{aligned}
\varphi(n \odot n') &= (\varphi_0(n \odot n'), \varphi_1(n \odot n')) \\
&= (\varphi_0 n \odot_0 \varphi_0 n', \varphi_1 n \odot_1 \varphi_1 n') \\
&= (\varphi_0 n, \varphi_1 n) \odot (\varphi_0 n', \varphi_1 n') \\
&= \varphi(n) \odot \varphi(n').
\end{aligned}$$

The analogous identity can be checked for unit elements. The coproduct of monoids, however, looks different from what we've seen thus far, but is related to the free monoid construction. More explicitly, given monoids $(M_i, \odot_i, e_i)$, we define their *free product* $(M_0 * M_1, \odot, e)$ as follows. Let $M_0 * M_1$ be the set of all words of alternating M_0 and M_1 non-unit elements with $\varnothing$ the empty word. Using superscripts to denote which monoid each element comes from, what follows are some $M_0 * M_1$ elements.

$$x^0 y^1 z^0, x^1 y^0 x^1 z^0, \varnothing$$

We note that these can start or end with a term from either M_0 or M_1, can include repeated elements, and can be of any finite length with the unique zero length word serving as the unit element. Since the unit element of any monoid is unique, we have to fuse the two monoids' units into a single unit. Since this is the case, we had to exclude units from our words since these could be absorbed into the element to their right or left. We will later see a more elegant characterization of this set.

Finally, we define the product $w \odot w'$ of two words w, w' as follows. If the final term of w and the first term of w' come from distinct monoids, then this product is merely the concatenation of the two words. If, in contrast, these two terms come from the same monoid, then we first concatenate the two lists, but then, so as to preserve alternation, replace these two adjacent elements with their product in the respective monoid to which they both belong. Towards greater formality, let $h(w)$ and $t(w)$ denote the first and last letter of w, respectively, and let $\overline{h}(w)$ and $\overline{t}(w)$ respectively denote *all but* the first and last term of w. Letting ; denote

concatenation, we have $w = h(w); \overline{h}(w) = \overline{t}(w); t(w)$, and hence define $\odot$ by

$$w \odot w' = \begin{cases} \overline{t}(w); t(w) \odot_i h(w'); \overline{h}(w') & t(w), h(w') \in M_i \\ w; w' & \text{otherwise} \end{cases}$$

Note that, in the case of singleton words coming from the same monoid, this is merely their product in that monoid. We now define the inclusion maps $\iota_i : M_i \to M$ as sending non-unit elements to their corresponding singleton words, and sending the unit element to the empty word. These are homomorphisms since we do not alter the elements, merely recast them as words. Finally, given two monoid homomorphisms $\varphi_i : M_i \to N$, we show that these vacuously assemble into a monoid homomorphism φ defined term-wise. More precisely, $\varphi(a; b) = \varphi(a); \varphi(b)$, where, so as to respect the inclusion maps, we define $\varphi(a) = \varphi_i(a)$ for $a \in M_i$. This map automatically preserves products $w \odot w'$ when these equal $w; w'$. In the other case, i.e. when $t(w), h(w') \in M_i$, we have:

$$\begin{aligned} \varphi(w \odot w') &= \varphi(\overline{t}(w); t(w) \odot_i h(w'); \overline{h}(w')) \\ &= \varphi\overline{t}(w); \varphi_i[t(w) \odot_i h(w')]; \varphi\overline{h}(w') \\ &= \varphi\overline{t}(w); \varphi_i t(w) \odot_i \varphi_i h(w'); \varphi\overline{h}(w') \\ &= \varphi\overline{t}(w); \varphi t(w) \odot_i \varphi h(w'); \varphi\overline{h}(w') \\ &= \varphi(\overline{t}(w); t(w)) \odot \varphi(h(w')\overline{h}(w)) \\ &= \varphi w \odot \varphi w' \end{aligned}$$

Finally, we define $\varphi\varnothing = e_N$. The reader can check that this plays well with the computation above.

The product and coproduct of vector spaces is particularly interesting, in that it illuminates much of the underlying theory of linear algebra. It turns out, that in $\mathbf{Vect}_k$, the product and coproduct are both given by the direct sum $\oplus$, where the direct sum of two k-vector spaces V, W is defined as having vectors:

$$V \oplus W = \{(v, w) \mid v \in V, w \in W\},$$

with addition and scalar multiplication given component-wise:

$$c \cdot (v, w) = (c \cdot v, c \cdot w) \qquad (v, w) + (v', w') = (v + v', w + w').$$

When the product and coproduct in $\mathcal{C}$ are isomorphic, we call it a *biproduct*. The fact that the direct sum is a product follows from an identical argument as the monoid product: in both cases, we have component-wise operations. We hence leave this as an exercise to the reader. The fact that the direct sum is also a coproduct, however, is more interesting, and is made possible by the existence of 0 and $+$. More precisely, given V_i for $i = 0, 1$, we need inclusion maps $\iota_i : V_i \to V$. These are given by $v_0 \mapsto (v_0, 0)$ and $v_1 \mapsto (0, v_1)$. For sets, we did not have a canonical arrow into a Cartesian product from a factor—but because vector spaces are equipped with the special element 0, we can just define the other component of an included factor as 0. Similarly, we now need a way to assemble two maps $\varphi_i : V_i \to W$. We now wish to define the combined map $\varphi : V_0 \oplus V_1 \to W$ so as to satisfy, for the sake of a bijective correspondence, $\iota_i \mathbin{/\!\!/} \varphi = \varphi_i$. We do so by exploiting linearity:

$$\varphi(v_0, v_1) = \varphi(v_0, 0) + \varphi(0, v_1) = \varphi_0(v_0) + \varphi_1(v_1).$$

This cannot be done in the case of typical Cartesian products of sets since there is no way to break up a generic set map $f(u, v)$ into its u and v domain influences. Linearity, however, forces these to be independent!

If $\mathcal{C}$ has products and coproducts, then if we have an arrow $f : X_0 + X_1 \to Y_0 \times Y_1$, then this can be studied via an array of 4 component maps $f_i^j : X_i \to Y_j$ for $i, j = 0, 1$. When $\mathcal{C}$ has biproducts, this then becomes true for maps $\varphi : X_0 \oplus X_1 \to Y_0 \oplus Y_1$. In the context of vector spaces, this is precisely the idea of a block matrix! More precisely, for a linear map

$$L : V_0 \oplus V_1 \to W_0 \oplus W_1$$

we have a 2×2 array, or *block matrix* whose ij^{th} entry is a linear map $V_j \to V_i$. In addition to having biproducts, The category of k-vector spaces enjoys even more structure: all of its objects are isomorphic to *bipowers*, i.e. repeated biproducts, of isomorphic copies of a single object: the base field k. Recall that, for any vector space V, there exists a tuple $\mathcal{B} = (b_i)_{i=1}^n \subset V$, called a *basis*, such that, for any vector $v \in V$, there are unique $c_i \in k$ for which $v = \sum_i c_i b_i$. This can be recast as a direct sum as follows. Given a vector $v \in V$, let $kv = \{cv \mid c \in k\}$ be the one dimensional vector space of scalar multiples of v. Note that $kv \cong k$ via $v \mapsto 1$. Then, given a basis $\mathcal{B} = (b_i)_{i=1}^n \subset V$, we have the following sequence of isomorphisms, instantiating the direct sum decomposotion of V.

$$V \cong \bigoplus_{i=1}^{n} kb_i \cong k^n$$

The isomorphism ${}_{\mathcal{B}}[-]$ and its inverse $[-]_{\mathcal{B}}$ is given by the following composition of maps. Given a vector $v \in V$, we can uniquely write $v = \sum_i c_i b_i$. We then map v to the tuple $(c_1 b_1, \ldots, c_n b_n)$ and then to $(c_1, \ldots, c_n)$. In turn, given a tuple $(c_1, \ldots, c_n) \in k^n$ to the tuple $(c_1 b_1, \ldots, c_n b_n)$ and then to the sum $\sum_i c_i b_i$. The existence and uniqueness of a linear expression in terms of the basis for any vector is precisely what allows these maps to be mutual inverses. Now, given *any* linear map $L : V \to W$, and a basis $\mathcal{A}$ of V and $\mathcal{B}$ of W, we have the following composition:

$$\begin{array}{ccc} V & \xrightarrow{L} & W \\ {\scriptstyle [-]_{\mathcal{A}}}\uparrow & & \downarrow{\scriptstyle {}_{\mathcal{B}}[-]} \\ k^m & \xrightarrow[{}_{\mathcal{B}}[L]_{\mathcal{A}}]{} & k^n \end{array}$$

Note that we used the shorthand ${}_{\mathcal{B}}[L]_{\mathcal{A}} = [-]_{\mathcal{A}} /\!\!/ L /\!\!/ {}_{\mathcal{B}}[-]$. The map ${}_{\mathcal{B}}[L]_{\mathcal{A}}$ is the matrix corresponding to L written in terms of the domain basis $\mathcal{A}$ and codomain basis $\mathcal{B}$. Recall that the ij^{th} entry ${}_i[L]_j$ of this matrix is a linear map $k \to k$, which is just a scalar; more precisely, it is the coefficient of the codomain basis b_i of the image of the domain basis a_j under the linear map L. In diagrammatic form, this entry can be encoded as follows.

$$\begin{array}{ccc} V & \xrightarrow{L} & W \\ \uparrow & & \downarrow \\ ka_i & \xrightarrow[{}_i[L]_j]{} & kb_i \end{array}$$

Note that we chose the convention that the domain basis goes on the right so as to match the classic matrix composition ordering:

$$_{\mathcal{B}_2}[L]_{\mathcal{B}_1}\,{}_{\mathcal{B}_1}[L']_{\mathcal{B}_0} = {}_{\mathcal{B}_2}[LL']_{\mathcal{B}_0}.$$

Products and coproducts in $\mathcal{C}$ not only act on objects, but, when they exist, can be seen as functors $\mathcal{C} \times \mathcal{C} \to \mathcal{C}$. More precisely, we map $\mathcal{C} \times \mathcal{C}$-objects as pairs (X, Y), that the functors send to $X \times Y$ or $X + Y$. Arrows in $\mathcal{C} \times \mathcal{C}$ are pairs $(f, g) : (X, Y) \to (X', Y')$, and these should be sent to morphisms of types $X \times Y \to X' \times Y'$ and $X + Y \to X' + Y'$. In the case of **Set**, these are defined in the intuitive way: $(x, y) \mapsto (fx, gy)$ and $x \mapsto fx, g \mapsto gy$ respectively. What about in general?

Thus the direct sum of linear maps $\oplus_i^n L_i : V_i \to W_i$ is the map $L : \oplus_i V_i \to \oplus_i W_i$ given by the following *block diagonal* matrix.

$$\begin{bmatrix} L_1 & 0 & \dots & 0 \\ 0 & L_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & L_n \end{bmatrix}$$

Thus, we can define a diagonal matrix as a direct sum of scalar maps $\oplus_i (c_i : k \to k)$.

Note that, although the direct sum of vector spaces involves the product of their unerlying sets, its dimension is the sum of their respective dimensions—more precisely, the basis of the direct sum is the disjoint union of the bases of each summand. This is true for a deep reason to be explained in the next section. What about a vector space whose basis is the product of the bases of some pair of vector spaces? It turns out that there is a universal construction that achieves this. To motivate this construction, let V and V' be vector spaces with bases $\mathcal{A} = (a_j)_{j \in J}$ and $\mathcal{A}' = (a_i)_{i \in I}$. Let L be a linear map, with domain the *set* of vectors $V \times V'$, and R a *bilinear* map with domain $V \times V'$. We then have the following scenarios.

$$\begin{aligned} L(v, w) &= L(\textstyle\sum_j c_j a_j, \sum_i c_i a_i) \\ &= L(\textstyle\sum_j c_j a_j, 0) + L(0, \sum_i c_i a_i) \\ &= \textstyle\sum_j c_j L(a_j, 0) + \sum_i c_i L(0, a_i) \\ &= \textstyle\sum_{k \in I + J} c_k L\hat{a}_k \end{aligned}$$

where we let $\hat{a}_k$ be $(a_j, 0)$ for $k \in J$ and $(0, a_i)$ for $k \in I$. This notation is made to transparently imply the idea that a linear map of two arguments is determined by the disjoint union of the bases for these two arguments' spaces. Now consider the case of bilinearity.

$$\begin{aligned} R(v, w) &= R(\textstyle\sum_j c_j a_j, \sum_i c_i a_i) \\ &= \textstyle\sum_i \sum_j c_j c_i R(a_j, a_i) \\ &= \textstyle\sum_{(i,j) \in I \times J} c_i c_j R(a_j \otimes a_i) \end{aligned}$$

where the *tensor* $a_j \otimes a_i$ is a formal symbol that represents the ordered pair (a_j, a_i), but now conceived as a generic element of a new domain vector space that we define in order to reconceptualize R as a linear map instead of a bilinear one. Note that R is determined by pairs of the two arguments, and hence will have as basis the Cartesian product of the bases of its factors, or *tensorands*. The tensor product will thus be multiplicative in dimension. We may construct the *tensor product* $V \otimes V'$

explicitly as follows.

$$V \otimes V' = \{\textstyle\sum_{i,j} c_{ij}(v_i \otimes v'_j) \mid v_i \in V, v'_j \in V', c_{ij} \in k\}/\sim,$$

where the equivalence relation encodes the multilinearity that we wish to be preserved by an outgoing linear map. More precisely:

$$[\textstyle\sum_i c_i v_i] \otimes [\sum_j c'_j v'_j] \sim \sum_{i,j} c_i c'_j (v_i \otimes v_j).$$

A linear map $\tilde{R} : V \otimes V' \to W$ then behaves like a bilinear map $R : V \times V' \to W$:

$$\begin{aligned}\tilde{R}([\textstyle\sum_i c_i v_i] \otimes [\sum_j c'_j v'_j]) &= \tilde{R}(\textstyle\sum_{i,j} c_i c'_j (v_i \otimes v_j)) \\ &= \textstyle\sum_{i,j} c_i c'_j \tilde{R}(v_i \otimes v_j)\end{aligned}$$

This can be made more precise via a universal property:

$$\begin{array}{ccc} V \times V' & \xrightarrow{R} & W \\ {\scriptstyle\rho}\downarrow & \nearrow_{\tilde{R}} & \\ V \otimes V' & & \end{array}$$

The map ρ is given by the rule $(v, v') \mapsto v \otimes v'$, and the dashed line, as usual, represents the unique arrow—in $\mathbf{Vect}_k$—that makes the diagram commute. Note that the non-dashed arrows are in this case *not* arrows in $\mathbf{Vect}_k$ but rather are bilinear maps. As usual, this universal property can be recast as a representable functor. More precisely let $\mathrm{Bilin}(V \times V', -) : \mathbf{Vect}_k \to \mathbf{Set}$ map a vector space W to the set of bilinear maps $V \times V' \to W$. Then the universal property gives the following natural isomorphism:

$$\mathrm{Bilin}(V \times V', -) \cong \mathbf{Vect}_k(V \otimes V', -).$$

Thus, although the tensor product is neither product nor coproduct in $\mathbf{Vect}_k$, it is nonetheless a universal binary operation on vector spaces, and one that plays the role of multiplication—at the least at the level of dimensionality—relative to the direct sum's role of addition. We will explore this further in the subsequent section.