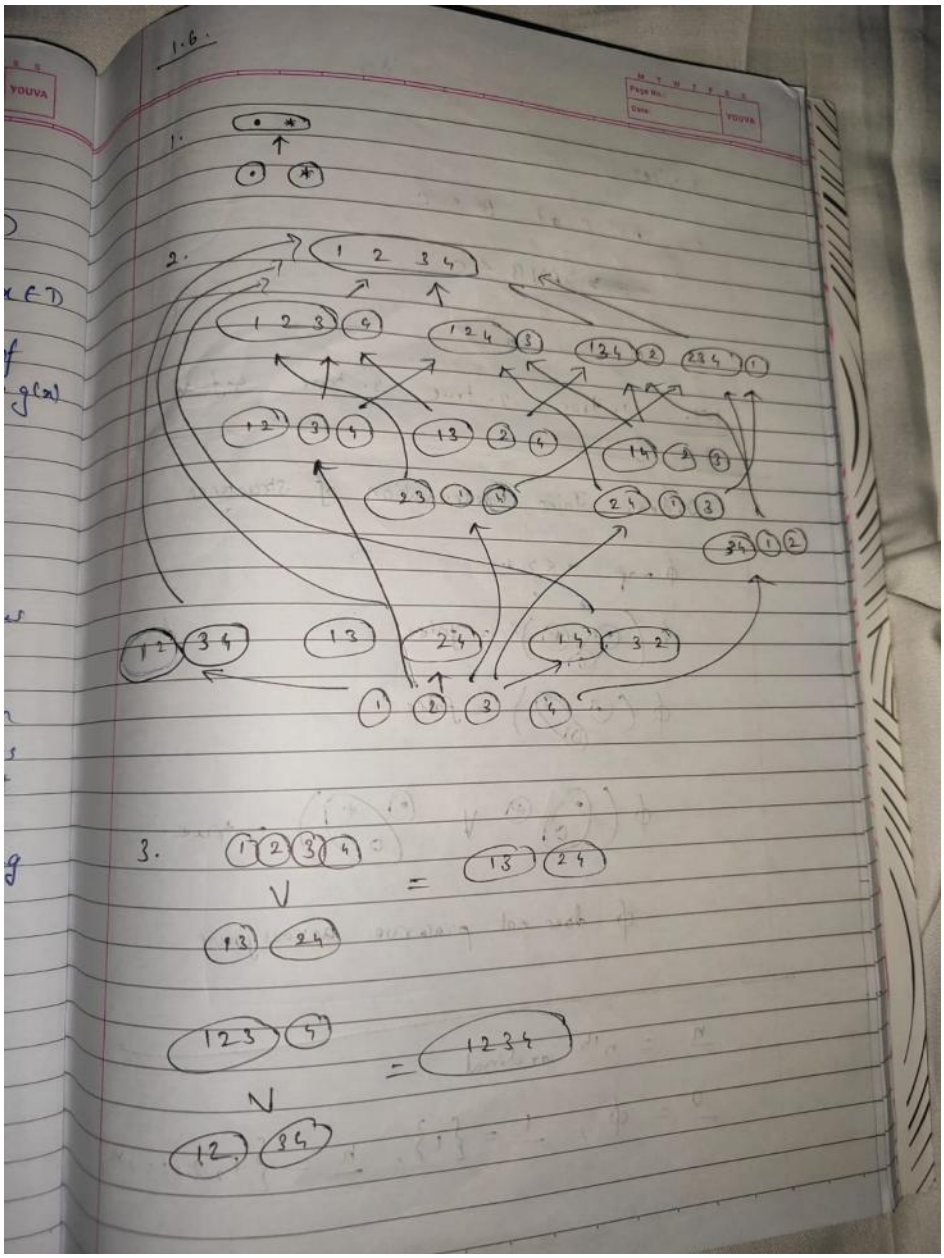
Exercises

08 December 2024 21:00



4. Yes

5. $A \leq C$ and $B \leq C$

$\Rightarrow A \vee B \leq C$

6. Yes

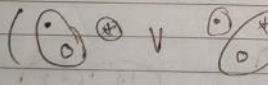
1.7. 1. true 2. true 3. true 4. false

Order, Joins, Preservation of structures

ϕ map $\cdot \leftrightarrow *$?

ϕ () = false

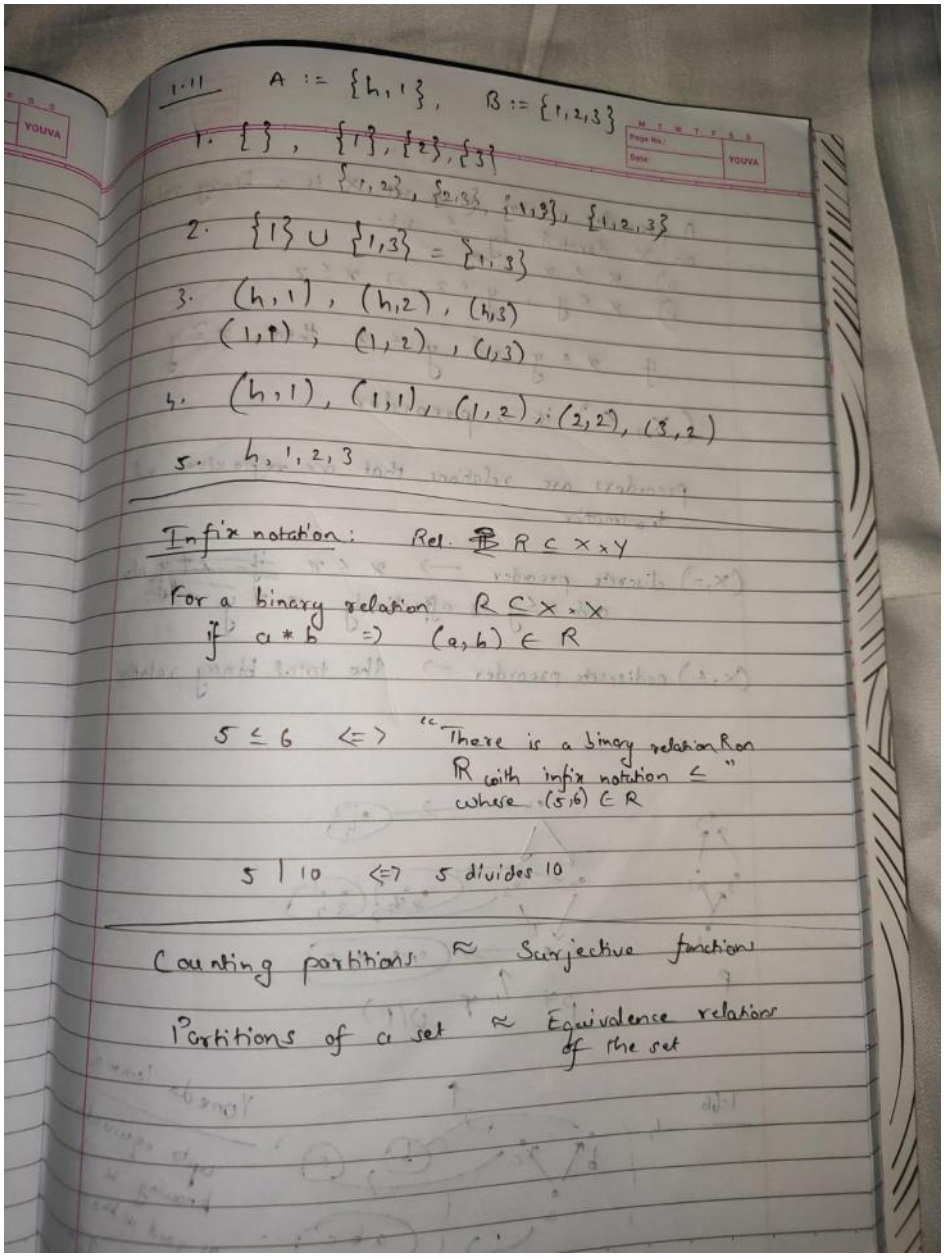
ϕ () = false

ϕ () = true

ϕ does not preserve across joins

$\underline{n}$ = n^{th} ordinal

$\underline{0} = \phi$, $\underline{1} = \{1\}$, $\underline{n} = \{1, 2, \cdots, n\}$


1.11 A := {h, 1}, B := {1, 2, 3}
1. {}, {1}, {2}, {3}
{1, 2}, {2, 3}, {1, 3}, {1, 2, 3}
2. {1} ∪ {1, 3} = {1, 3}
3. (h, 1), (h, 2), (h, 3)
(1, 1), (1, 2), (1, 3)
4. (h, 1), (1, 1), (1, 2), (2, 2), (3, 2)
5. h, 1, 2, 3
Infix notation: Rel. R ⊆ X × Y
For a binary relation R ⊆ X × X
if a * b ⇒ (a, b) ∈ R
5 ≤ 6 ⟺ "There is a binary relation R on R with infix notation ≤" where (5, 6) ∈ R
5 | 10 ⟺ 5 divides 10
Counting partitions ≈ Surjective functions
Partitions of a set ≈ Equivalence relations of the set

A preorder relation on a set X is a binary relation on X denoted by $\leq$ s.t.

a) $x \leq x$

b) $x \leq y \cdot y \leq z \Rightarrow x \leq z$

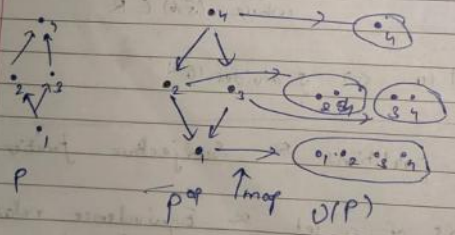
if $x \leq y$ and $y \leq x$ then $x \cong y$

$(X, \leq)$ is a preorder

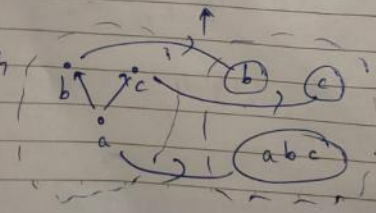
preorders are relations that are reflexive and transmitive

$(X, =)$ discrete preorder $\longrightarrow$ $x \leq x$ always and $x \leq y$ is effectively $x \neq y$

$(X, \leq)$ codiscrete preorder $\longrightarrow$ the total binary relation



1.66



Yoneda lemma

upto equivalence knowing x is as good as knowing $\uparrow(x)$

Notes

08 December 2024 21:02

Basic Stuff

Example 1.9. Here are some important sets from mathematics—and the notation we will use—that will appear again in this book.

- $\varnothing$ denotes the empty set; it has no elements.
- $\{1\}$ denotes a set with one element; it has one element, 1.
- $\mathbb{B}$ denotes the set of *booleans*; it has two elements, `true` and `false`.
- $\mathbb{N}$ denotes the set of *natural numbers*; it has elements $0, 1, 2, 3, \ldots, 90^{717}, \ldots$.
- $\underline{n}$, for any $n \in \mathbb{N}$, denotes the n^{th} *ordinal*; it has n elements $1, 2, \ldots, n$. For example, $\underline{0} = \varnothing$, $\underline{1} = \{1\}$, and $\underline{5} = \{1, 2, 3, 4, 5\}$.
- $\mathbb{Z}$, the set of *integers*; it has elements $\ldots, -2, -1, 0, 1, 2, \ldots, 90^{717}, \ldots$.
- $\mathbb{R}$, the set of *real numbers*; it has elements like $\pi, 3.14, 5 * \sqrt{2}, e, e^2, -1457, 90^{717}$, etc.

Definition 1.12. Let X and Y be sets. A *relation between* X *and* Y is a subset $R \subseteq X \times Y$. A *binary relation on* X is a relation between X and X, i.e. a subset $R \subseteq X \times X$.

Definition 1.18. Let A be a set. An *equivalence relation* on A is a binary relation, let's give it infix notation $\sim$, satisfying the following three properties:

(a) $a \sim a$, for all $a \in A$,
(b) $a \sim b$ iff[a] $b \sim a$, for all $a, b \in A$, and
(c) if $a \sim b$ and $b \sim c$ then $a \sim c$, for all $a, b, c \in A$.

Definition 1.21. Given a set A and an equivalence relation $\sim$ on A, we say that the *quotient* $A/\sim$ of A under $\sim$ is the set of parts of the corresponding partition.

Preorders

Definition 1.30. A *preorder relation* on a set X is a binary relation on X, here denoted with infix notation $\leq$, such that

(a) $x \leq x$; and
(b) if $x \leq y$ and $y \leq z$, then $x \leq z$.

The first condition is called *reflexivity* and the second is called *transitivity*. If $x \leq y$ and $y \leq x$, we write $x \cong y$ and say x and y are *equivalent*. We call a pair $(X, \leq)$ consisting of a set equipped with a preorder relation a *preorder*.

Example 1.52 (Partitions). We talked about getting a partition from a preorder; now let's think about how we might order the set $\mathsf{Prt}(A)$ of *all partitions* of A, for some set A. In fact, we have done this before in Eq. (1.5). Namely, we order on partitions by fineness: a partition P is *finer* than a partition Q if, for every part $p \in P$ there is a part $q \in Q$ such that $A_p \subseteq A_q$. We could also say that Q is *coarser* than P.

Recall from Example 1.26 that partitions on A can be thought of as surjective functions out of A. Then $f : A \twoheadrightarrow P$ is finer than $g : A \twoheadrightarrow Q$ if there is a function $h : P \to Q$ such that $f \fatsemi h = g$.

Example 1.56 (Product preorder). Given preorders $(P, \leq)$ and $(Q, \leq)$, we may define a preorder structure on the product set $P \times Q$ by setting $(p, q) \leq (p', q')$ if and only if $p \leq p'$ and $q \leq q'$. We call this the *product preorder*. This is a basic example of a more general construction known as the product of categories.

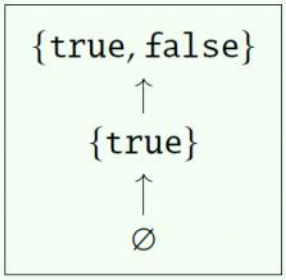
Example 1.58 (Opposite preorder). Given a preorder $(P, \leq)$, we may define the opposite preorder $(P, \leq^{\mathrm{op}})$ to have the same set of elements, but with $p \leq^{\mathrm{op}} q$ if and only if $q \leq p$.

Monotone Maps

Definition 1.59. A *monotone map* between preorders $(A, \leq_A)$ and $(B, \leq_B)$ is a function $f : A \to B$ such that, for all elements $x, y \in A$, if $x \leq_A y$ then $f(x) \leq_B f(y)$.

Example 1.54 (Upper sets). Given a preorder $(P, \leq)$, an *upper set* in P is a subset U of P satisfying the condition that if $p \in U$ and $p \leq q$, then $q \in U$. "If p is an element then so is anything bigger." Write $\mathsf{U}(P)$ for the set of upper sets in P. We can give the set U an order by letting $U \leq V$ if U is contained in V.

For example, if $(\mathbb{B}, \leq)$ is the booleans (Example 1.34), then its preorder of uppersets $\mathsf{U}(\mathbb{B})$ is



The subset $\{\texttt{false}\} \subseteq \mathbb{B}$ is not an upper set, because $\texttt{false} \leq \texttt{true}$ and $\texttt{true} \notin \{\texttt{false}\}$.

Example 1.68. Recall from Example 1.52 that given a set X we define $\mathsf{Prt}(X)$ to be the set of partitions on X, and that a partition may be defined using a surjective function $s : X \twoheadrightarrow P$ for some set P.

Any surjective function $f : X \twoheadrightarrow Y$ induces a monotone map $f^* : \mathsf{Prt}(Y) \to \mathsf{Prt}(X)$, going "backwards." It is defined by sending a partition $s : Y \twoheadrightarrow P$ to the composite $f \fatsemi s : X \twoheadrightarrow P$.[7]

Meet and Join

Definition 1.81. Let $(P, \leq)$ be a preorder, and let $A \subseteq P$ be a subset. We say that an element $p \in P$ is a *meet* of A if

(a) for all $a \in A$, we have $p \leq a$, and

(b) for all q such that $q \leq a$ for all $a \in A$, we have that $q \leq p$.

We write $p = \bigwedge A$, $p = \bigwedge_{a \in A} a$, or, if the dummy variable a is clear from context, just $p = \bigwedge_A a$. If A just consists of two elements, say $A = \{a, b\}$, we can denote $\bigwedge A$ simply by $a \wedge b$.

Similarly, we say that p is a *join* of A if

(a) for all $a \in A$ we have $a \leq p$, and

(b) for all q such that $a \leq q$ for all $a \in A$, we have that $p \leq q$.

We write $p = \bigvee A$ or $p = \bigvee_{a \in A} a$, or when $A = \{a, b\}$ we may simply write $p = a \vee b$.

Definition 1.92. We say that a monotone map $f : P \to Q$ *preserves meets* if $f(a \wedge b) \cong f(a) \wedge f(b)$ for all $a, b \in P$. We similarly say f *preserves joins* if $f(a \vee b) \cong f(a) \vee f(b)$ for all $a, b \in P$.

Definition 1.93. We say that a monotone map $f : P \to Q$ *has a generative effect* if there exist elements $a, b \in P$ such that

$$f(a) \vee f(b) \ncong f(a \vee b).$$

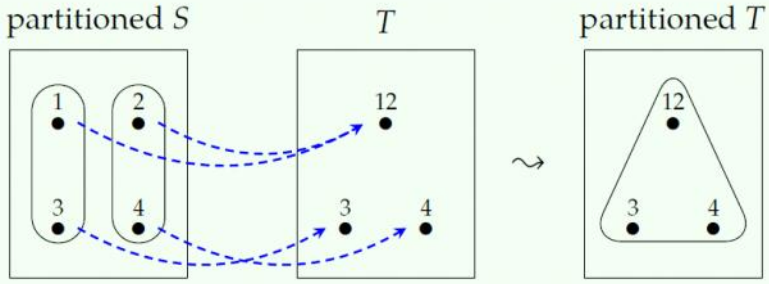
Galois Connections

Definition 1.95. A *Galois connection* between preorders P and Q is a pair of monotone maps $f : P \to Q$ and $g : Q \to P$ such that

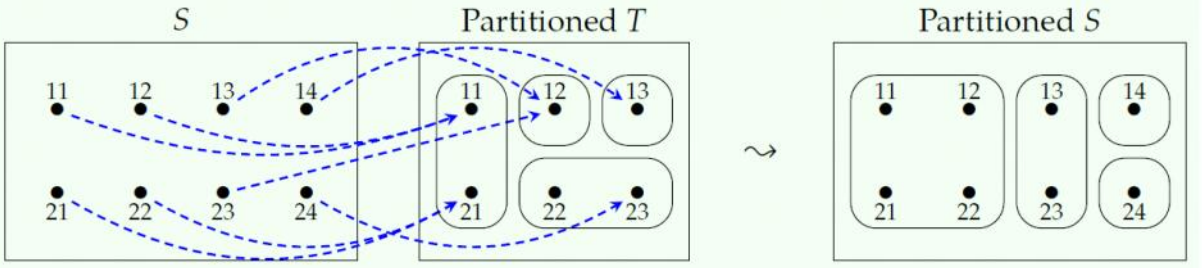
$$f(p) \leq q \quad \text{if and only if} \quad p \leq g(q). \tag{1.96}$$

We say that f is the *left adjoint* and g is the *right adjoint* of the Galois connection.

Example 1.102. Let $S = \{1, 2, 3, 4\}$, $T = \{12, 3, 4\}$, and $g : S \to T$ by $g(1) := g(2) := 12$, $g(3) := 3$, and $g(4) := 4$. The partition shown left below is translated by $g_!$ to the partition shown on the right.



Example 1.104. Let S, T be as below, and let $g\colon S \to T$ be the function shown in blue. Here is a picture of how g^* takes a partition on T and "pulls it back" to a partition on S:



Proposition 1.107. Suppose that $f\colon P \to Q$ and $g\colon Q \to P$ are monotone maps. The following are equivalent

(a) f and g form a Galois connection where f is left adjoint to g,

(b) for every $p \in P$ and $q \in Q$ we have

$$p \leq g(f(p)) \qquad \text{and} \qquad f(g(q)) \leq q. \tag{1.108}$$

Go left and then right or right and then left, no matter what you preserve x-> adjunctions
Here x is monotonicity

Preventing Generative Effects

Proposition 1.111 (Right adjoints preserve meets)**.** Let $f\colon P \to Q$ be left adjoint to $g\colon Q \to P$. Suppose $A \subseteq Q$ any subset, and let $g(A) := \{g(a) \mid a \in A\}$ be its image. Then if A has a meet $\bigwedge A \in Q$ then $g(A)$ has a meet $\bigwedge g(A)$ in P, and we have

$$g\left(\bigwedge A\right) \cong \bigwedge g(A).$$

That is, right adjoints preserve meets. Similarly, left adjoints preserve joins: if $A \subseteq P$ is any subset that has a join $\bigvee A \in P$, then $f(A)$ has a join $\bigvee f(A)$ in Q, and we have

$$f\left(\bigvee A\right) \cong \bigvee f(A).$$

Theorem 1.115 (Adjoint functor theorem for preorders)**.** Suppose Q is a preorder that has all meets and let P be any preorder. A monotone map $g\colon Q \to P$ preserves meets if and only if it is a right adjoint.

Similarly, if P has all joins and Q is any preorder, a monotone map $f\colon P \to Q$ preserves joins if and only if it is a left adjoint.

Notes

14 December 2024 12:17

Consider the following three questions you might ask yourself:

- Given what I have, is it *possible* to get what I want? → bool
- Given what I have, what is the *minimum cost* to get what I want? → cost
- Given what I have, what is the *set of ways* to get what I want? → Power

We begin with a formal definition of symmetric monoidal preorders.

Definition 2.2. A *symmetric monoidal structure* on a preorder $(X, \leq)$ consists of two constituents:

(i) an element $I \in X$, called the *monoidal unit*, and
(ii) a function $\otimes\colon X \times X \to X$, called the *monoidal product*.

These constituents must satisfy the following properties, where we write $\otimes(x_1, x_2) = x_1 \otimes x_2$:

(a) for all $x_1, x_2, y_1, y_2 \in X$, if $x_1 \leq y_1$ and $x_2 \leq y_2$, then $x_1 \otimes x_2 \leq y_1 \otimes y_2$,
(b) for all $x \in X$, the equations $I \otimes x = x$ and $x \otimes I = x$ hold,
(c) for all $x, y, z \in X$, the equation $(x \otimes y) \otimes z = x \otimes (y \otimes z)$ holds, and
(d) for all $x, y \in X$, the equation $x \otimes y = y \otimes x$ holds.

We call these conditions *monotonicity*, *unitality*, *associativity*, and *symmetry* respectively. A preorder equipped with a symmetric monoidal structure, $(X, \leq, I, \otimes)$, is called a *symmetric monoidal preorder*.

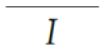
monoid ⇒ way of combining

Wires = elements
Boxes = relationships
Parallelism = combination

$$\frac{x}{\dfrac{}{y}} \qquad \overline{x \otimes y}$$

We consider wires in parallel to represent the monoidal product of their labels, so we consider both cases above to represent the element $x \otimes y$. Note also that a wire labeled

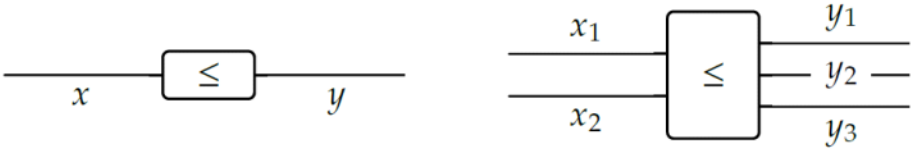
I or an absence of wires:



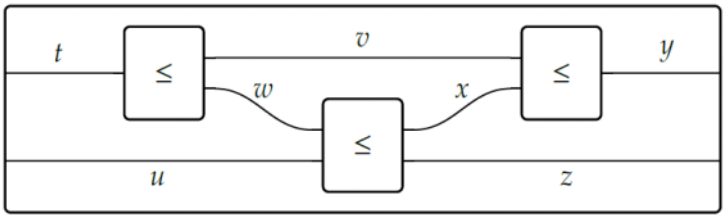
nothing

both represent the monoidal unit I; another way of thinking of this is that the unit is the empty monoidal product.

A wiring diagram runs between a set of parallel wires on the left and a set of parallel wires on the right. We say that a wiring diagram is *valid* if the monoidal product of the elements on the left is less than the monoidal product of those on the right. For example, if we have the inequality $x \leq y$, the the diagram that is a box with a wire labeled x on the left and a wire labeled y on the right is valid; see the first box below:



Finally, the symmetry condition (d), that $x \otimes y = y \otimes x$, says that a diagram is valid even if its wires cross:



(2.15)

The inner boxes in Eq. (2.15) translate into the assertions:

$$t \leq v + w \qquad w + u \leq x + z \qquad v + x \leq y \tag{2.16}$$

and the outer box translates into the assertion:

$$t + u \leq y + z. \tag{2.17}$$

You can add more axioms to symmetric monoidal preorders and get -> discard, copy, etc

Example 2.27 (Booleans with AND). We can define a monoidal structure on $\mathbb{B}$ by letting the monoidal unit be `true` and the monoidal product be $\wedge$ (AND). If one thinks of `false` $= 0$ and `true` $= 1$, then $\wedge$ corresponds to the usual multiplication operation $*$. That is, with this correspondence, the two tables below match up:

$\wedge$	`false`	`true`
`false`	`false`	`false`
`true`	`false`	`true`

$*$	0	1
0	0	0
1	0	1

(2.28)

One can check that all the properties in Definition 2.2 hold, so we have a monoidal preorder which we denote **Bool** $:= (\mathbb{B}, \leq, \texttt{true}, \wedge)$.

The Cost Preorder

Example 2.37 (Lawvere's monoidal preorder, **Cost**). Let $[0, \infty]$ denote the set of non-negative real numbers—such as 0, 1, $15.33\overline{3}$, and 2π—together with ∞. Consider the preorder $([0, \infty], \geq)$, with the usual notion of $\geq$, where of course $\infty \geq x$ for all $x \in [0, \infty]$.

There is a monoidal structure on this preorder, where the monoidal unit is 0 and the monoidal product is $+$. In particular, $x + \infty = \infty$ for any $x \in [0, \infty]$. Let's call this

monoidal preorder

$$\textbf{Cost} := ([0, \infty], \geq, 0, +),$$

because we can think of the elements of $[0, \infty]$ as costs. In terms of structuring "getting from here to there," **Cost** seems to say "getting from a to b is a question of cost." The monoidal unit being 0 will translate into saying that you can always get from a to a at no cost. The monoidal product being $+$ will translate into saying that the cost of getting from a to c is at most the cost of getting from a to b *plus* the cost of getting from b to c. Finally, the "at most" in the previous sentence is coming from the $\geq$.

Opposite Categories

Proposition 2.38. Suppose $\mathcal{X} = (X, \leq)$ is a preorder and $\mathcal{X}^{\mathrm{op}} = (X, \geq)$ is its opposite. If $(X, \leq, I, \otimes)$ is a symmetric monoidal preorder then so is its opposite, $(X, \geq, I, \otimes)$.

Monoidal Monotones

Definition 2.41. Let $\mathcal{P} = (P, \leq_P, I_P, \otimes_P)$ and $\mathcal{Q} = (Q, \leq_Q, I_Q, \otimes_Q)$ be monoidal preorders. A *monoidal monotone* from $\mathcal{P}$ to $\mathcal{Q}$ is a monotone map $f : (P, \leq_P) \to (Q, \leq_Q)$, satisfying two conditions:

(a) $I_Q \leq_Q f(I_P)$, and

(b) $f(p_1) \otimes_Q f(p_2) \leq_Q f(p_1 \otimes_P p_2)$

for all $p_1, p_2 \in P$.

There are strengthenings of these conditions that are also important. If f satisfies the following conditions, it is called a *strong monoidal monotone*:

(a′) $I_Q \cong f(I_P)$, and
(b′) $f(p_1) \otimes_Q f(p_2) \cong f(p_1 \otimes_P p_2)$;
and if it satisfies the following conditions it is called a *strict monoidal monotone*:
(a″) $I_Q = f(I_P)$, and
(b″) $f(p_1) \otimes_Q f(p_2) = f(p_1 \otimes_P p_2)$.

Enriched Categories

Definition 2.46. Let $\mathcal{V} = (V, \leq, I, \otimes)$ be a symmetric monoidal preorder. A *$\mathcal{V}$-category* $\mathcal{X}$ consists of two constituents, satisfying two properties. To specify $\mathcal{X}$,
(i) one specifies a set $\mathrm{Ob}(\mathcal{X})$, elements of which are called *objects*;
(ii) for every two objects x, y, one specifies an element $\mathcal{X}(x, y) \in V$, called the *hom-object*.[2]
The above constituents are required to satisfy two properties:
(a) for every object $x \in \mathrm{Ob}(\mathcal{X})$ we have $I \leq \mathcal{X}(x, x)$, and
(b) for every three objects $x, y, z \in \mathrm{Ob}(\mathcal{X})$, we have $\mathcal{X}(x, y) \otimes \mathcal{X}(y, z) \leq \mathcal{X}(x, z)$.
We call $\mathcal{V}$ the *base of the enrichment* for $\mathcal{X}$ or say that $\mathcal{X}$ is *enriched* in $\mathcal{V}$.

Theorem 2.49. There is a one-to-one correspondence between preorders and **Bool**-categories.

Definition 2.53. A *Lawvere metric space* is a **Cost**-category.

Definition 2.51. A *metric space* (X, d) consists of:
(i) a set X, elements of which are called *points*, and
(ii) a function $d\colon X \times X \to \mathbb{R}_{\geq 0}$, where $d(x, y)$ is called the *distance between x and y*.

A lawvere metric space drops criteria (b), (c) from the distance function

(a) for every $x \in X$, we have $d(x, x) = 0$,
(b) for every $x, y \in X$, if $d(x, y) = 0$ then $x = y$,
(c) for every $x, y \in X$, we have $d(x, y) = d(y, x)$, and
(d) for every $x, y, z \in X$, we have $d(x, y) + d(y, z) \geq d(x, z)$.

Changing enrichment

2.4.1 Changing the base of enrichment

Any monoidal monotone $\mathcal{V} \to \mathcal{W}$ between symmetric monoidal preorders lets us convert $\mathcal{V}$-categories into $\mathcal{W}$-categories.

Construction 2.64. Let $f\colon \mathcal{V} \to \mathcal{W}$ be a monoidal monotone. Given a $\mathcal{V}$-category $\mathcal{C}$, one forms the associated $\mathcal{W}$-category, say $\mathcal{C}_f$ as follows.
(i) We take the same objects: $\mathrm{Ob}(\mathcal{C}_f) := \mathrm{Ob}(\mathcal{C})$.
(ii) For any $c, d \in \mathrm{Ob}(\mathcal{C})$, put $\mathcal{C}_f(c, d) := f(\mathcal{C}(c, d))$.

Enriched Functors

Definition 2.69. Let $\mathcal{X}$ and $\mathcal{Y}$ be $\mathcal{V}$-categories. A *$\mathcal{V}$-functor from $\mathcal{X}$ to $\mathcal{Y}$*, denoted $F\colon \mathcal{X} \to \mathcal{Y}$, consists of one constituent:
(i) a function $F\colon \mathrm{Ob}(\mathcal{X}) \to \mathrm{Ob}(\mathcal{Y})$
subject to one constraint
(a) for all $x_1, x_2 \in \mathrm{Ob}(\mathcal{X})$, one has $\mathcal{X}(x_1, x_2) \leq \mathcal{Y}(F(x_1), F(x_2))$.

Product Category

Definition 2.74. Let $\mathcal{X}$ and $\mathcal{Y}$ be $\mathcal{V}$-categories. Define their *$\mathcal{V}$-product*, or simply *product*, to be the $\mathcal{V}$-category $\mathcal{X} \times \mathcal{Y}$ with
(i) $\mathrm{Ob}(\mathcal{X} \times \mathcal{Y}) := \mathrm{Ob}(\mathcal{X}) \times \mathrm{Ob}(\mathcal{Y})$,
(ii) $(\mathcal{X} \times \mathcal{Y})((x, y), (x', y')) := \mathcal{X}(x, x') \otimes \mathcal{Y}(y, y')$,
for two objects (x, y) and (x', y') in $\mathrm{Ob}(\mathcal{X} \times \mathcal{Y})$.

Example 2.76. Let $\mathcal{X}$ and $\mathcal{Y}$ be the Lawvere metric spaces (i.e. **Cost**-categories) defined by the following weighted graphs:

$$\mathcal{X} := \boxed{A \xrightarrow{2} B \xrightarrow{3} C} \qquad \boxed{\begin{array}{c} p \\ 5 \left(\uparrow\downarrow\right) 8 \\ q \end{array}} =: \mathcal{Y} \tag{2.77}$$

Their product is defined by taking the product of their sets of objects, so there are six objects in $\mathcal{X} \times \mathcal{Y}$. And the distance $d_{X\times Y}((x,y),(x',y'))$ between any two points is given by the sum $d_X(x,x') + d_Y(y,y')$.

Examine the following graph, and make sure you understand how easy it is to derive from the weighted graphs for $\mathcal{X}$ and $\mathcal{Y}$ in Eq. (2.77):

$$\mathcal{X} \times \mathcal{Y} = \boxed{\begin{array}{ccccc} (A,p) & \xrightarrow{2} & (B,p) & \xrightarrow{3} & (C,p) \\ 5\left(\uparrow\downarrow\right)8 & & 5\left(\uparrow\downarrow\right)8 & & 5\left(\uparrow\downarrow\right)8 \\ (A,q) & \xrightarrow{2} & (B,q) & \xrightarrow{3} & (C,q) \end{array}}$$

Monoidal Closed Preorders

Definition 2.79. A symmetric monoidal preorder $\mathcal{V} = (V, \leq, I, \otimes)$ is called *symmetric monoidal closed* (or just *closed*) if, for every two elements $v, w \in V$, there is an element $v \multimap w$ in $\mathcal{V}$, called the *hom-element*, with the property

$$(a \otimes v) \leq w \quad \text{iff} \quad a \leq (v \multimap w). \tag{2.80}$$

for all $a, v, w \in V$.

Proposition 2.87. Suppose $\mathcal{V} = (V, \leq, I, \otimes, \multimap)$ is a symmetric monoidal preorder that is closed. Then

(a) For every $v \in V$, the monotone map $- \otimes v \colon (V,\leq) \to (V,\leq)$ is left adjoint to $v \multimap - \colon (V,\leq) \to (V,\leq)$.

(b) For any element $v \in V$ and set of elements $A \subseteq V$, if the join $\bigvee_{a\in A} a$ exists then so does $\bigvee_{a\in A} v \otimes a$ and we have

$$\left(v \otimes \bigvee_{a\in A} a\right) \cong \bigvee_{a\in A} (v \otimes a). \tag{2.88}$$

(c) For any $v, w \in V$, we have $v \otimes (v \multimap w) \leq w$.

(d) For any $v \in V$, we have $v \cong (I \multimap v)$.

(e) For any $u, v, w \in V$, we have $(u \multimap v) \otimes (v \multimap w) \leq (u \multimap w)$.

Quantales

18 December 2024 10:00

Definition 2.90. A *unital commutative quantale* is a symmetric monoidal closed preorder $\mathcal{V} = (V, \leq, I, \otimes, \multimap)$ that has all joins: $\bigvee A$ exists for every $A \subseteq V$. In particular, we often denote the empty join by $0 := \bigvee \emptyset$.

Proposition 2.96. Let $\mathcal{P} = (P, \leq)$ be a preorder. It has all joins iff it has all meets.

Remark 2.97. The notion of Hausdorff distance can be generalized, allowing the role of **Cost** to be taken by any quantale $\mathcal{V}$. If $\mathcal{X}$ is a $\mathcal{V}$-category with objects X, and $U \subseteq X$ and $V \subseteq X$, we can generalize the usual Hausdorff distance, on the left below, to the formula on the right:

$$d(U, V) := \sup_{u \in U} \inf_{v \in V} d(u, v) \qquad \mathcal{X}(U, V) := \bigwedge_{u \in U} \bigvee_{v \in V} \mathcal{X}(u, v).$$

For example, if $\mathcal{V} = \mathbf{Bool}$, the Hausdorff distance between sub-preorders U and V answers the question "can I get into V from every $u \in U$," i.e. $\forall_{u \in U}. \exists_{v \in V}. u \leq v$. Or for another example, use $\mathcal{V} = \mathsf{P}(M)$ with its interpretation as modes of transportation, as in Exercise 2.62. Then the Hausdorff distance $d(U, V) \in \mathsf{P}(M)$ tells us those modes of transportation that will get us into V from every point in U.

Proposition 2.98. Suppose $\mathcal{V} = (V, \leq, I, \otimes)$ is any symmetric monoidal preorder that has all joins. Then $\mathcal{V}$ is closed—i.e. it has a $\multimap$ operation and hence is a quantale—if and only if $\otimes$ distributes over joins; i.e. if Eq. (2.88) holds for all $v \in V$ and $A \subseteq V$.

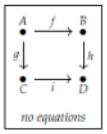
Notes

16 January 2025 18:22

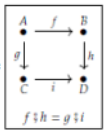
Free Category

Definition 3.7. For any graph $G = (V, A, s, t)$, we can define a category **Free**(G), called the *free category on* G, whose objects are the vertices V and whose morphisms from c to d are the paths from c to d. The identity morphism on an object c is simply the trivial path at c. Composition is given by concatenation of paths.

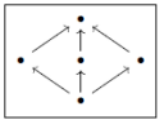
Free_square :=

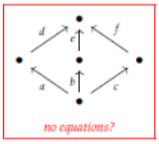


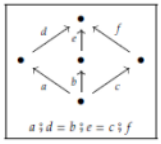
Comm_square :=



Preorders







Remark 3.23 (Ends of a spectrum). The main point of this subsection is that both preorders and free categories are specified by a graph without path equations, but they denote opposite ends of a spectrum. In both cases, the vertices of the graph become the objects of a category and the paths become morphisms. But in the case of free categories, there are no equations so each path becomes a different morphism. In the case of preorders, all parallel paths become the same morphism. Every category presentation, i.e. graph with some equations, lies somewhere in between the free category (no equations) and its preorder reflection (all possible equations).

Functors

Definition 3.35. Let $\mathcal{C}$ and $\mathcal{D}$ be categories. To specify a *functor from* $\mathcal{C}$ *to* $\mathcal{D}$, denoted $F\colon \mathcal{C} \to \mathcal{D}$,

(i) for every object $c \in \mathrm{Ob}(\mathcal{C})$, one specifies an object $F(c) \in \mathrm{Ob}(\mathcal{D})$;

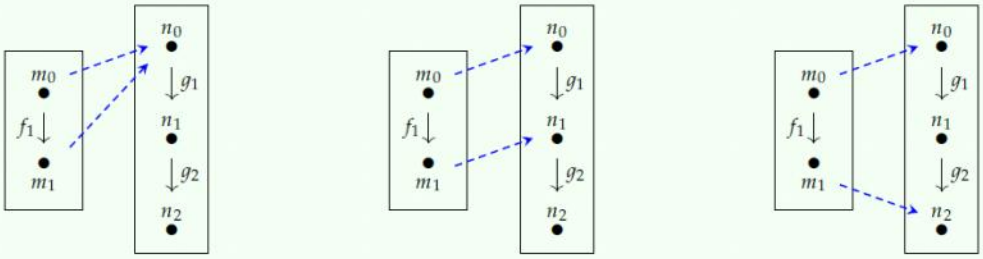
(ii) for every morphism $f\colon c_1 \to c_2$ in $\mathcal{C}$, one specifies a morphism $F(f)\colon F(c_1) \to F(c_2)$ in $\mathcal{D}$.

The above constituents must satisfy two properties:

(a) for every object $c \in \mathrm{Ob}(\mathcal{C})$, we have $F(\mathrm{id}_c) = \mathrm{id}_{F(c)}$.

(b) for every three objects $c_1, c_2, c_3 \in \mathrm{Ob}(\mathcal{C})$ and two morphisms $f \in \mathcal{C}(c_1, c_2)$, $g \in \mathcal{C}(c_2, c_3)$, the equation $F(f \fatsemi g) = F(f) \fatsemi F(g)$ holds in $\mathcal{D}$.

Example 3.36. For example, here we draw three functors $F\colon \mathbf{2} \to \mathbf{3}$:



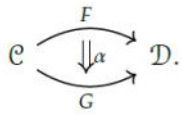
In each case, the dotted arrows show what the functor F does to the vertices in $\mathbf{2}$; once that information is specified, it turns out—in this special case—that what F does to the three paths in $\mathbf{2}$ is completely determined. In the left-hand diagram, F sends every path to the trivial path, i.e. the identity on n_0. In the middle diagram $F(m_0) = n_0$, $F(f_1) = g_1$, and $F(m_1) = n_1$. In the right-hand diagram, $F(m_0) = n_0$, $F(m_1) = n_2$, and $F(f_1) = g_1 \fatsemi g_2$.

Databases as set-valued functors

Definition 3.44. Let $\mathcal{C}$ be a schema, i.e. a finitely-presented category. A $\mathcal{C}$-*instance* is a functor $I\colon \mathcal{C} \to \mathbf{Set}$.[5]

Natural Transformations

If $\mathcal{C}$ is a schema—i.e. a finitely-presented category—then there are many database instances on it, which we can organize into a category. But this is part of a larger story, namely that of natural transformations. An abstract picture to have in mind is this:



Definition 3.49. Let $\mathcal{C}$ and $\mathcal{D}$ be categories, and let $F, G\colon \mathcal{C} \to \mathcal{D}$ be functors. To specify a *natural transformation* $\alpha\colon F \Rightarrow G$,

(i) for each object $c \in \mathcal{C}$, one specifies a morphism $\alpha_c\colon F(c) \to G(c)$ in $\mathcal{D}$, called the *c-component of* α.

These components must satisfy the following, called the *naturality condition*:

(a) for every morphism $f\colon c \to d$ in $\mathcal{C}$, the following equation must hold:

$$F(f) \fatsemi \alpha_d = \alpha_c \fatsemi G(f).$$

A natural transformation $\alpha\colon F \to G$ is called a *natural isomorphism* if each component α_c is an isomorphism in $\mathcal{D}$.

$$\begin{array}{ccc} F(c) & \xrightarrow{\alpha_c} & G(c) \\ {\scriptstyle F(f)}\downarrow & & \downarrow{\scriptstyle G(f)} \\ F(d) & \xrightarrow[\alpha_d]{} & G(d) \end{array}$$

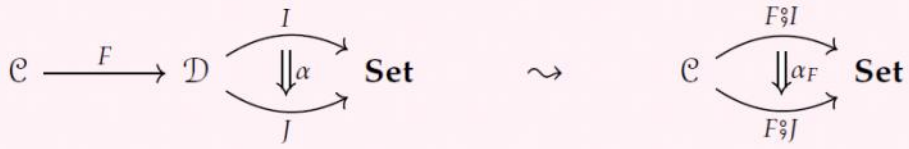
Definition 3.51. A *diagram* D in $\mathcal{C}$ is a functor $D\colon \mathcal{J} \to \mathcal{C}$ from any category $\mathcal{J}$, called the *indexing category* of the diagram D. We say that D *commutes* if $D(f) = D(f')$ holds for every parallel pair of morphisms $f, f'\colon a \to b$ in $\mathcal{J}$.[7]

Definition 3.54. Let $\mathcal{C}$ and $\mathcal{D}$ be categories. We denote by $\mathcal{D}^{\mathcal{C}}$ the category whose objects are functors $F\colon \mathcal{C} \to \mathcal{D}$ and whose morphisms $\mathcal{D}^{\mathcal{C}}(F, G)$ are the natural transformations $\alpha\colon F \to G$. This category $\mathcal{D}^{\mathcal{C}}$ is called the *functor category*, or the *category of functors from* $\mathcal{C}$ *to* $\mathcal{D}$.

Definition 3.60. Suppose that $\mathcal{C}$ is a database schema and $I, J\colon \mathcal{C} \to \mathbf{Set}$ are database instances. An *instance homomorphism* between them is a natural transformation $\alpha\colon I \to J$. Write $\mathcal{C}$-**Inst** $:=$ $\mathbf{Set}^{\mathcal{C}}$ to denote the functor category as defined in Definition 3.54.

Definition 3.68. Let $\mathcal{C}$ and $\mathcal{D}$ be categories and let $F\colon \mathcal{C} \to \mathcal{D}$ be a functor. For any set-valued functor $I\colon \mathcal{D} \to \mathbf{Set}$, we refer to the composite functor $F \fatsemi I\colon \mathcal{C} \to \mathbf{Set}$ as the *pullback of I along F*.

Given a natural transformation $\alpha\colon I \Rightarrow J$, there is a natural transformation $\alpha_F\colon F\fatsemi I \Rightarrow F \fatsemi J$, whose component $(F \fatsemi I)(c) \to (F \fatsemi J)(c)$ for any $c \in \mathrm{Ob}(\mathcal{C})$ is given by $(\alpha_F)_c \coloneqq \alpha_{Fc}$.



This uses the data of F to define a functor $\Delta_F\colon \mathcal{D}\text{-}\mathbf{Inst} \to \mathcal{C}\text{-}\mathbf{Inst}$.

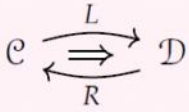
Definition 3.70. Let $\mathcal{C}$ and $\mathcal{D}$ be categories, and $L\colon \mathcal{C} \to \mathcal{D}$ and $R\colon \mathcal{D} \to \mathcal{C}$ be functors. We say that L *is left adjoint to* R (and that R *is right adjoint to* L) if, for any $c \in \mathcal{C}$ and $d \in \mathcal{D}$, there is an isomorphism of hom-sets

$$\alpha_{c,d}\colon \mathcal{C}(c, R(d)) \xrightarrow{\cong} \mathcal{D}(L(c), d)$$

that is natural in c and d.[8]

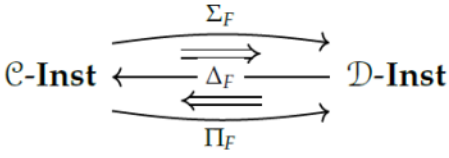
Given a morphism $f\colon c \to R(d)$ in $\mathcal{C}$, its image $g \coloneqq \alpha_{c,d}(f)$ is called its *mate*. Similarly, the mate of $g\colon L(c) \to d$ is f.

To denote an adjunction we write $L \dashv R$, or in diagrams,

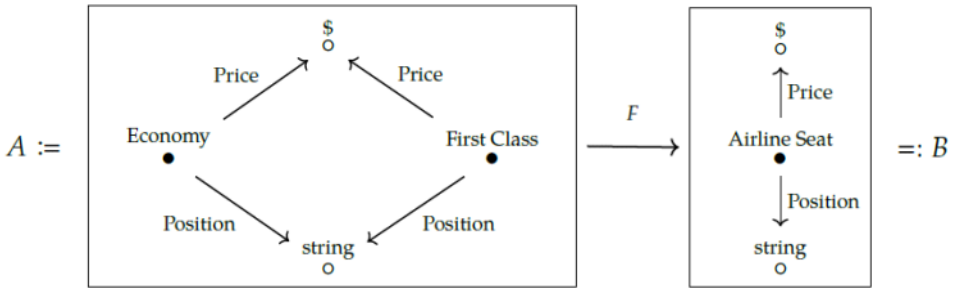


with the $\Rightarrow$ in the direction of the left adjoint.

Given $F\colon \mathcal{C} \to \mathcal{D}$, the data migration functor Δ_F turns $\mathcal{D}$-instances into $\mathcal{C}$-instances. This functor has both a left and a right adjoint:



Migration Functor	Pronounced	Reminiscent of	Database idea
Δ	Delta	Duplicate or destroy	Duplicate or destroy tables or columns
Σ	Sigma	Sum	Union (sum up) data
Π	Pi	Product	Pair[9] and query data



$\Sigma_F(I)(\text{Airline Seat}) = I(\text{Economy}) \sqcup I(\text{First Class})$.

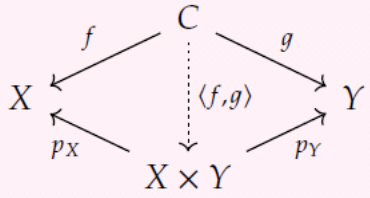
Limits and Colimits

20 January 2025 01:12

Definition 3.79. Let $\mathcal{C}$ be a category. Then an object Z in $\mathcal{C}$ is a *terminal object* if, for each object C of $\mathcal{C}$, there exists a unique morphism $!\colon C \to Z$.

Proposition 3.84. All terminal objects in a category $\mathcal{C}$ are isomorphic.

Definition 3.86. Let $\mathcal{C}$ be a category, and let X, Y be objects in $\mathcal{C}$. A *product* of X and Y is an object, denoted $X \times Y$, together with morphisms $p_X\colon X \times Y \to X$ and $p_Y\colon X \times Y \to Y$ such that for all objects C together with morphisms $f\colon C \to X$ and $g\colon C \to Y$, there exists a unique morphism $C \to X \times Y$, denoted $\langle f, g\rangle$, for which the following diagram commutes:



Definition 3.92. Let $D\colon \mathcal{J} \to \mathcal{C}$ be a diagram. A *cone* (C, c_*) over D consists of

(i) an object $C \in \mathcal{C}$;

(ii) for each object $j \in \mathcal{J}$, a morphism $c_j\colon C \to D(j)$.

To be a cone, these must satisfy the following property:

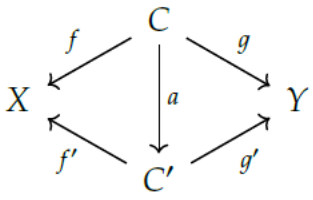
(a) for each $f : j \to k$ in $\mathcal{J}$, we have $c_k = c_j \fatsemi D(f)$.

A *morphism of cones* $(C, c_*) \to (C', c'_*)$ is a morphism $a : C \to C'$ in $\mathcal{C}$ such that for all $j \in \mathcal{J}$ we have $c_j = a \fatsemi c'_j$. Cones over D, and their morphisms, form a category $\mathbf{Cone}(D)$.

The *limit* of D, denoted $\lim(D)$, is the terminal object in the category $\mathbf{Cone}(D)$. Say it is the cone $\lim(D) = (C, c_*)$; we refer to C as the *limit object* and the map c_j for any $j \in \mathcal{J}$ as the j^{th} *projection map*.

It turns out that products *are* terminal objects, but of a different category, which we'll call **Cone**(X, Y), *the category of cones over* X *and* Y *in* $\mathcal{C}$. We will see in Exercise 3.91 that $X \xleftarrow{p_X} X \times Y \xrightarrow{p_Y} Y$ is a terminal object in **Cone**(X, Y).

An object of **Cone**(X, Y) is simply a pair of maps $X \xleftarrow{f} C \xrightarrow{g} Y$. A morphism from $X \xleftarrow{f} C \xrightarrow{g} Y$ to $X \xleftarrow{f'} C' \xrightarrow{g'} Y$ in **Cone**(X, Y) is a morphism $a : C \to C'$ in $\mathcal{C}$ such that the following diagram commutes:



Recall from Definition 3.51 that formally speaking, a diagram in $\mathcal{C}$ is just a functor $D : \mathcal{J} \to \mathcal{C}$. Here $\mathcal{J}$ is called the *indexing category* of the diagram D.

Theorem 3.95. Let $\mathcal{J}$ be a category presented by the finite graph (V, A, s, t) together with some equations, and let $D : \mathcal{J} \to$ **Set** be a set-valued functor. Write $V = \{v_1, \ldots, v_n\}$. The set

$$\lim_{\mathcal{J}} D := \big\{(d_1, \ldots, d_n) \mid d_i \in D(v_i) \text{ for all } 1 \le i \le n \text{ and } \\ \text{for all } a : v_i \to v_j \in A, \text{ we have } D(a)(d_i) = d_j\big\}.$$

together with the projection maps $p_i : (\lim_{\mathcal{J}} D) \to D(v_i)$ given by $p_i(d_1, \ldots, d_n) := d_i$, is a limit of D.

Definition 3.102. Given a category $\mathcal{C}$ we say that a *cocone* in $\mathcal{C}$ is a cone in $\mathcal{C}^{\mathrm{op}}$.

Given a diagram $D : \mathcal{J} \to \mathcal{C}$, we may take the limit of the functor $D^{\mathrm{op}} : \mathcal{J}^{\mathrm{op}} \to \mathcal{C}^{\mathrm{op}}$. This is a cone in $\mathcal{C}^{\mathrm{op}}$, and so by definition a cocone in $\mathcal{C}$. The *colimit* of D is this cocone.

Notes

18 January 2025 18:22

The theory of co-design is based on preorders: each resource—e.g. velocity, torque, or $—is structured as a preorder. The order $x \leq y$ represents the *availability of x given y*, i.e. that whenever you have y, you also have x. For example, in our preorder of wattage, if 5W ≤ 10W, it means that whenever we are provided 10W, we implicitly also have 5W. Above we referred to this as an order from less useful to more useful: if x is always available given y, then x is less useful than y.

Bool-category	preorder
Bool-functor	monotone map
Bool-profunctor	feasibility relation

Definition 4.2. Let $\mathcal{X} = (X, \leq_X)$ and $\mathcal{Y} = (Y, \leq_Y)$ be preorders. A *feasibility relation* for $\mathcal{X}$ given $\mathcal{Y}$ is a monotone map

$$\Phi\colon \mathcal{X}^{\mathrm{op}} \times \mathcal{Y} \to \mathbf{Bool}. \tag{4.3}$$

We denote this by $\Phi\colon \mathcal{X} \nrightarrow \mathcal{Y}$.

Given $x \in X$ and $y \in Y$, if $\Phi(x, y) = \texttt{true}$ we say *x can be obtained given y*.

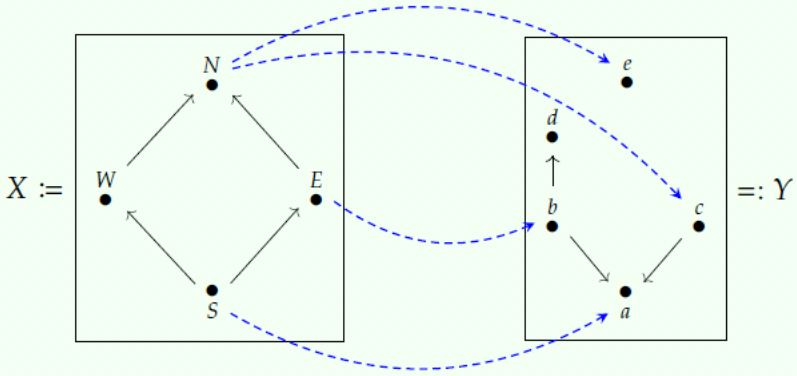
Definition 4.8. Let $\mathcal{V} = (V, \leq, I, \otimes)$ be a (unital commutative) quantale,[1] and let $\mathcal{X}$ and $\mathcal{Y}$ be $\mathcal{V}$-categories. A *$\mathcal{V}$-profunctor* from $\mathcal{X}$ to $\mathcal{Y}$, denoted $\Phi\colon \mathcal{X} \nrightarrow \mathcal{Y}$, is a $\mathcal{V}$-functor

$$\Phi\colon \mathcal{X}^{\mathrm{op}} \times \mathcal{Y} \to \mathcal{V}.$$

Quantale = unital commutative quantale (preorder with all joins)

Example 4.11 (**Bool**-profunctors and their interpretation as bridges). Let's discuss Definition 4.8 in the case $\mathcal{V} = \mathbf{Bool}$. One way to imagine a **Bool**-profunctor $\Phi\colon X \nrightarrow Y$ is in terms of building bridges between two cities. Recall that a preorder (a **Bool**-category) can be drawn using a Hasse diagram. We'll think of the preorder as a city, and each vertex in it as some point of interest. An arrow $A \to B$ in the Hasse diagram means that there exists a way to get from point A to point B in the city. So what's a profunctor?

A profunctor is just a bunch of bridges connecting points in one city to points in another. Let's see a specific example. Here is a picture of a **Bool**-profunctor $\Phi\colon X \nrightarrow Y$:



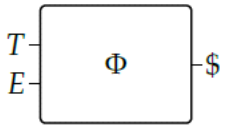
Both X and Y are preorders, e.g. with $W \leq N$ and $b \leq a$. With bridges coming from the profunctor in blue, one can now use both paths within the cities and the bridges to get from points in city X to points in city Y. For example, since there is a path from N to e and E to a, we have $\Phi(N, e) = \texttt{true}$ and $\Phi(E, a) = \texttt{true}$. On the other hand, since there is no path from W to d, we have $\Phi(W, d) = \texttt{false}$.

In fact, one could put a box around this entire picture and see a new preorder with $W \leq N \leq c \leq a$, etc. This is called the *collage* of Φ; we'll explore this in more detail later; see Definition 4.42.

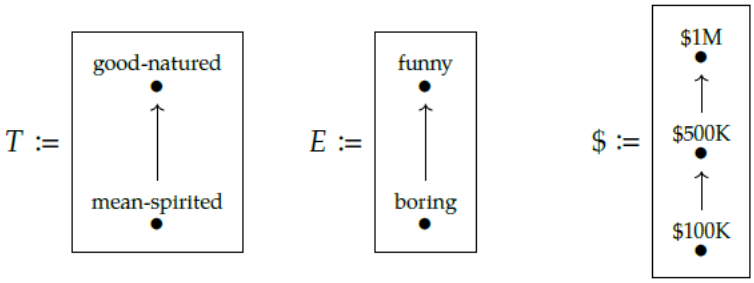
Definition 4.21. Let $\mathcal{V}$ be a quantale, let $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{Z}$ be $\mathcal{V}$-categories, and let $\Phi\colon \mathcal{X} \nrightarrow \mathcal{Y}$ and $\Psi\colon \mathcal{Y} \nrightarrow \mathcal{Z}$ be $\mathcal{V}$-profunctors. We define their *composite*, denoted $\Phi \fatsemi \Psi\colon \mathcal{X} \nrightarrow \mathcal{Z}$ by the formula

$$(\Phi \fatsemi \Psi)(p, r) = \bigvee_{q \in Q} \big(\Phi(p, q) \otimes \Psi(q, r)\big).$$

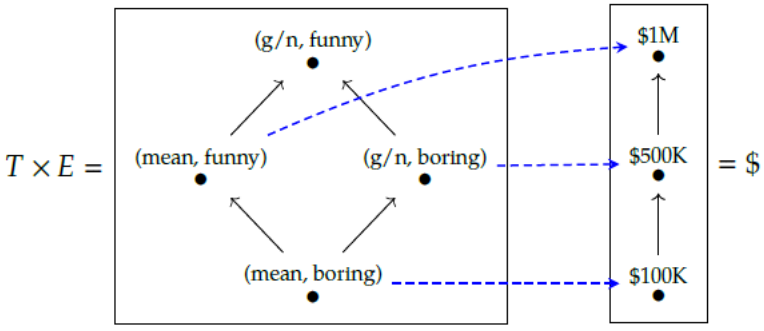
Let's walk through this a bit more concretely. Consider the design problem of filming a movie, where you must pit the tone and entertainment value against the cost. A feasibility relation describing this situation details what tone and entertainment value can be obtained at each cost; as such, it is described by a feasibility relation $\Phi\colon (T \times E) \nrightarrow \$$. We represent this by the box



where T, E, and $\$$ are the preorders drawn below:



A possible feasibility relation is then described by the profunctor



Theorem 4.23. For any skeletal quantale $\mathcal{V}$, there is a category $\mathbf{Prof}_{\mathcal{V}}$ whose objects are $\mathcal{V}$-categories $\mathcal{X}$, whose morphisms are $\mathcal{V}$-profunctors $\mathcal{X} \nrightarrow \mathcal{Y}$, and with composition defined as in Definition 4.21.

Definition 4.24. We define $\mathbf{Feas} := \mathbf{Prof}_{\mathbf{Bool}}$.

Companions, conjoints, collages

Definition 4.34. Let $F\colon \mathcal{P} \to \mathcal{Q}$ be a $\mathcal{V}$-functor. The *companion of* F, denoted $\widehat{F}\colon \mathcal{P} \nrightarrow \mathcal{Q}$ and the *conjoint of* F, denoted $\check{F}\colon \mathcal{Q} \nrightarrow \mathcal{P}$ are defined to be the following $\mathcal{V}$-profunctors:

$$\widehat{F}(p,q) := \mathcal{Q}(F(p),q) \quad \text{and} \quad \check{F}(q,p) := \mathcal{Q}(q,F(p))$$

Definition 4.42. Let $\mathcal{V}$ be a quantale, let $\mathcal{X}$ and $\mathcal{Y}$ be $\mathcal{V}$-categories, and let $\Phi\colon \mathcal{X} \nrightarrow \mathcal{Y}$ be a $\mathcal{V}$-profunctor. The *collage of* Φ, denoted $\mathbf{Col}(\Phi)$ is the $\mathcal{V}$-category defined as follows:

(i) $\mathrm{Ob}(\mathbf{Col}(\Phi)) := \mathrm{Ob}(\mathcal{X}) \sqcup \mathrm{Ob}(\mathcal{Y})$;

(ii) For any $a, b \in \mathrm{Ob}(\mathbf{Col}(\Phi))$, define $\mathbf{Col}(\Phi)(a,b) \in \mathcal{V}$ to be

$$\mathbf{Col}(\Phi)(a,b) := \begin{cases} \mathcal{X}(a,b) & \text{if } a,b \in \mathcal{X} \\ \Phi(a,b) & \text{if } a \in \mathcal{X}, b \in \mathcal{Y} \\ \varnothing & \text{if } a \in \mathcal{Y}, b \in \mathcal{X} \\ \mathcal{Y}(a,b) & \text{if } a,b \in \mathcal{Y} \end{cases}$$

There are obvious functors $i_{\mathcal{X}}\colon \mathcal{X} \to \mathbf{Col}(\Phi)$ and $i_{\mathcal{Y}}\colon \mathcal{Y} \to \mathbf{Col}(\Phi)$, sending each object and morphism to "itself," called *collage inclusions*.