An Invitation to APPLIED CATEGORY THEORY

Seven Sketches in Compositionality

BRENDAN FONG
DAVID I. SPIVAK

An Invitation to Applied Category Theory

Seven Sketches in Compositionality

BRENDAN FONG
Massachusetts Institute of Technology

DAVID I. SPIVAK
Massachusetts Institute of Technology

CAMBRIDGE
UNIVERSITY PRESS

CAMBRIDGE
UNIVERSITY PRESS

University Printing House, Cambridge CB2 8BS, United Kingdom

One Liberty Plaza, 20th Floor, New York, NY 10006, USA

477 Williamstown Road, Port Melbourne, VIC 3207, Australia

314–321, 3rd Floor, Plot 3, Splendor Forum, Jasola District Centre, New Delhi – 110025, India

79 Anson Road, #06–04/06, Singapore 079906

Cambridge University Press is part of the University of Cambridge.

It furthers the University's mission by disseminating knowledge in the pursuit of education, learning, and research at the highest international levels of excellence.

www.cambridge.org
Information on this title: www.cambridge.org/9781108482295
DOI: 10.1017/9781108668804

First published 2019

Printed in the United Kingdom by TJ International Ltd, Padstow Cornwall

A catalogue record for this publication is available from the British Library.

Library of Congress Cataloging-in-Publication Data
Names: Fong, Brendan, 1988– author. | Spivak, David I., 1978– author.
Title: An invitation to applied category theory : seven sketches in compositionality / Brendan Fong (Massachusetts Institute of Technology), David I. Spivak (Massachusetts Institute of Technology).
Description: Cambridge ; New York, NY : Cambridge University Press, 2019. | Includes bibliographical references and index.
Identifiers: LCCN 2018058456 | ISBN 9781108482295
Subjects: LCSH: Categories (Mathematics) | Computable functions. | Logic, Symbolic and mathematical. | Mathematical analysis.
Classification: LCC QA9.25 .F66 2019 | DDC 512/.62–dc23
LC record available at https://lccn.loc.gov/2018058456

ISBN 978-1-108-48229-5 Hardback
ISBN 978-1-108-71182-1 Paperback

Contents

Preface

Category theory is becoming a central hub for all of pure mathematics. It is unmatched in its ability to organize and layer abstractions, to find commonalities between structures of all sorts, and to facilitate communication between different mathematical communities.

But it has also been branching out into science, informatics, and industry. We believe that it has the potential to be a major cohesive force in the world, building rigorous bridges between disparate worlds, both theoretical and practical. The motto at MIT is *mens et manus*, Latin for mind and hand. We believe that category theory – and pure math in general – has stayed in the realm of mind for too long; it is ripe to be brought to hand.

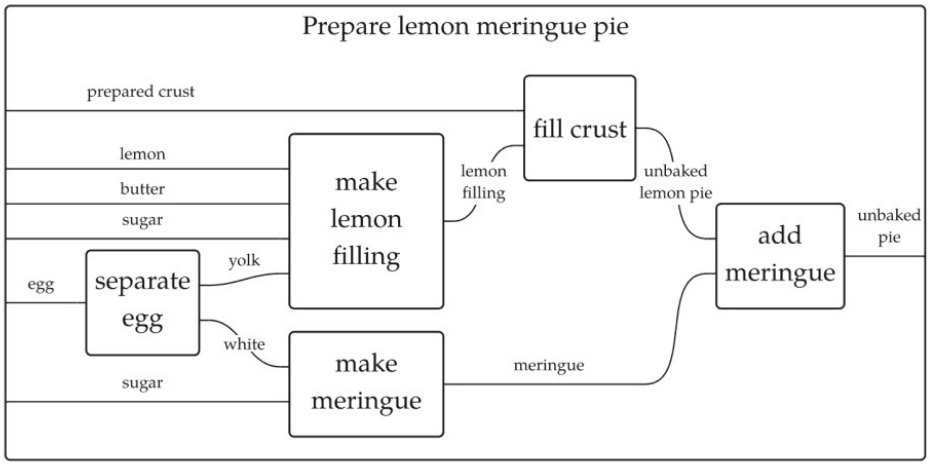
Purpose and audience

The purpose of this book is to offer a self-contained tour of applied category theory. It is an invitation to discover advanced topics in category theory through concrete real-world examples. Rather than try to give a comprehensive treatment of these topics – which include adjoint functors, enriched categories, proarrow equipments, toposes, and much more – we merely provide a taste of each. We want to give readers some insight into how it feels to work with these structures as well as some ideas about how they might show up in practice.

The audience for this book is quite diverse: anyone who finds the above description intriguing. This could include a motivated high school student who hasn't seen calculus yet but has loved reading a weird book on mathematical logic they found at the library. Or a machine-learning researcher who wants to understand what vector spaces, design theory, and dynamical systems could possibly have in common. Or a pure mathematician who wants to imagine what sorts of applications their work might have. Or a recently retired programmer who's always had an eerie feeling that category theory is what they've been looking for to tie it all together, but who's found the usual books on the subject impenetrable.

For example, we find it something of a travesty that at the time of publication there is almost no introductory material available on monoidal categories. Even beautiful modern introductions to category theory, e.g. by Riehl [Rie17] or Leinster [Lei14], do not include anything on this rather central topic. The only exceptions we can think of are [CK17, Chapter 3] and [CP10], each of which has a very user-friendly introduction to monoidal categories; however, readers who are not drawn to physics may not think to look there.

The basic idea of monoidal categories is certainly not too abstract; modern human intuition seems to include a pre-theoretical understanding of monoidal categories that is just waiting to be formalized. Is there anyone who wouldn't correctly understand the basic idea being communicated in the following diagram?



Many applied category theory topics seem to take monoidal categories as their jumping-off point. So one aim of this book is to provide a reference – even if unconventional – for this important topic.

We hope this book inspires both new visions and new questions. We intend it to be self-contained in the sense that it is approachable with minimal prerequisites, but not in the sense that the complete story is told here. On the contrary, we hope that readers use this as an invitation to further reading, to orient themselves in what is becoming a large literature, and to discover new applications for themselves.

This book is, unashamedly, our take on the subject. While the abstract structures we explore are important to any category theorist, the specific topics have simply been chosen to our personal taste. Our examples are ones that we find simple but powerful, concrete but representative, entertaining but in a way that feels important and expansive at the same time. We hope our readers will enjoy themselves and learn a lot in the process.

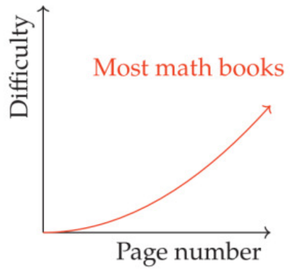
How to read this book

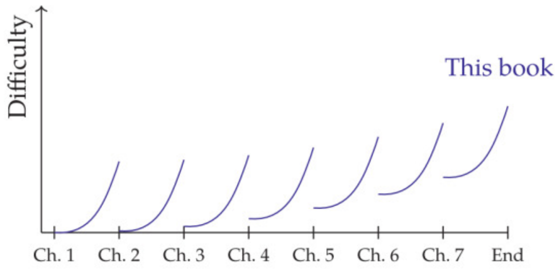
The basic idea of category theory – which threads through every chapter – is that if one pays careful attention to structures and coherence, the resulting systems will be extremely reliable and interoperable. For example, a category involves several structures: a collection of objects, a collection of morphisms relating objects, and a formula for combining any chain of morphisms into a morphism. But these structures need to *cohere* or work together in a simple commonsense way: a chain of chains is itself a long chain, so combining a chain of chains should be the same as combining the long chain. That's it!

We shall see structures and coherence come up in pretty much every definition we give: "here are some things and here are how they fit together." We ask the reader to be on the lookout for structures and coherence as they read the book, and to realize that as we layer abstraction upon abstraction, it is the coherence that makes all the parts work together harmoniously in concert.

Each chapter in this book is motivated by a real-world topic, such as electrical circuits, control theory, cascade failures, information integration, and hybrid systems. These motivations lead us into and through various sorts of category-theoretic concepts. We generally have one motivating idea and one category-theoretic purpose per chapter, and this forms the title of the chapter, e.g. Chapter 4 is "Co-design: Profunctors, Categorification, and Monoidal Categories."

In many math books, the difficulty is roughly a monotonically increasing function of the page number. In this book, this occurs in each chapter, but not so much in the book as a whole. The chapters start out fairly easy and progress in difficulty.





The upshot is that if you find the end of a chapter very difficult, hope is certainly not lost: you can start on the next one and make good progress. This format lends itself to giving you a first taste now, but also leaving open the opportunity for you to come back to the book at a later date and get more deeply into it. But by all means, if you have the gumption to work through each chapter to its end, we very much encourage that!

We include about 240 exercises throughout the text, with solutions in the Appendix at the end of the book. Usually these exercises are fairly straightforward; the only thing they demand is that readers change their mental state from passive to active, reread the previous paragraphs with intent, and put the pieces together. Readers become *students* when they work through the exercises; until then they are more tourists, riding on a bus and listening off and on to the tour guide. Hey, there's nothing wrong with that, but we do encourage you to get off the bus and make direct contact with the native population and local architecture as often as you can.

Acknowledgments

Thanks to Jared Briskman, James Brock, Ronnie Brown, Thrina Burana, David Chudzicki, Jonathan Castello, Margo Crawford, Fred Eisele, David Ellerman, Cam Fulton, Bruno Gavranović, Sebastian Galkin, John Garvin, Peter Gates, Juan Manuel Gimeno, Alfredo Gómez, Leo Gorodinski, Jason Grossman, Jason Hooper, Yuxi Liu, Jesús López, MTM, Nicolò Martini, Martin MacKerel, Pete Morcos, Nelson Niu, James Nolan, Dan Oneata, Paolo Perrone, Thomas Read, Rif A. Saurous, Dan Schmidt, Samantha Seaman, Marcello Seri, Robert Smart, Valter Sorana, Adam Theriault-Shay, Emmy Trewartha, Sergey Tselovalnikov, Andrew Turner, Joan Vazquez, Daniel Wang, Jerry Wedekind for helpful comments and conversations.

We also thank our sponsors at the AFOSR; this work was supported by grants FA9550–14–1–0031 and FA9550–17–1–0058.

Finally, we extend a very special thanks to John Baez for running an online course (https://forum.azimuthproject.org) on this material and generating tons of great feedback.

Personal note

Our motivations to apply category theory outside of math are, perhaps naively, grounded in the hope it can help bring humanity together to solve our big problems. But category theory is a tool for thinking, and like any tool it can be used for purposes we align with and those we don't.

In this personal note, we ask that readers try to use what they learn in this book to do something they would call "good," in terms of contributing to the society they'd want to live in. For example, if you're planning to study this material with others, consider specifically inviting someone from an underrepresented minority – a group that is more highly represented in society than in upper-level math classes – to your study group. As another example, perhaps you can use the material in this book to design software that helps people relate to and align with each other. What is the mathematics of a well-functioning society?

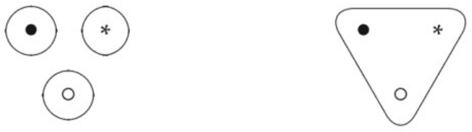
The way we use our tools affects all our lives. Our society has seen the results – both the wonders and the waste – resulting from rampant selfishness. We would be honored if readers found ways to use category theory as part of an effort to connect people, to create common ground, to explore the cross-cutting categories in which life, society, and environment can be represented, and to end the ignorance entailed by limiting ourselves to a singular ontological perspective on anything.

If you do something of the sort, please let us and the community know about it.

Brendan Fong and David I. Spivak

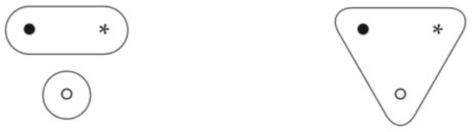
Connections are symmetric, so if a is connected to b, then b is connected to a. Connections are also transitive, meaning that if a is connected to b, and b is connected to c, then a is connected to c; that is, all a, b, and c are connected. Friendship is not transitive – my friend's friend is not necessarily my friend – but possible communication of a concept or a disease is.

Here we depict two more systems, one in which none of the points are connected, and one in which all three points are connected.

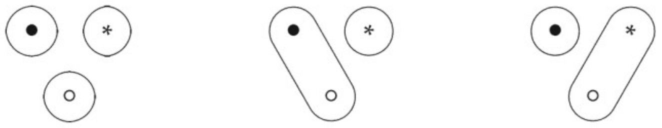


There are five systems in all, and we depict them below.

Now that we have defined the sort of system we want to discuss, suppose that Alice is observing this system. Her observation of interest, which we call Φ, extracts a single feature from a system, namely whether the point $\bullet$ is connected to the point $*$; this is what she wants to know. Her observation of the system will be an assignment of either `true` or `false`; she assigns `true` if $\bullet$ is connected to $*$, and `false` otherwise. So Φ assigns the value `true` to the following two systems:



and Φ assigns the value `false` to the three remaining systems:

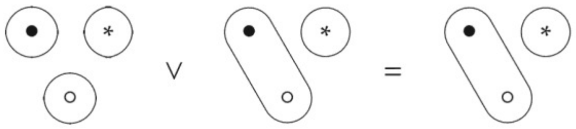


(1.1)

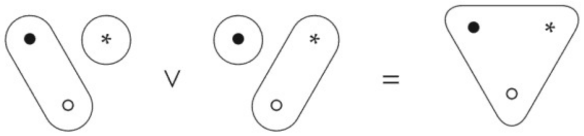
The last piece of setup is to give a sort of operation that Alice wants to perform on the systems themselves. It's a very common operation – one that will come up many times throughout the book – called *join*. If the reader has been following the story arc, the expectation here is that Alice's connectivity observation will not be compositional with respect to the operation of system joining; that is, there will be generative effects. Let's see what this means.

Joining our simple systems

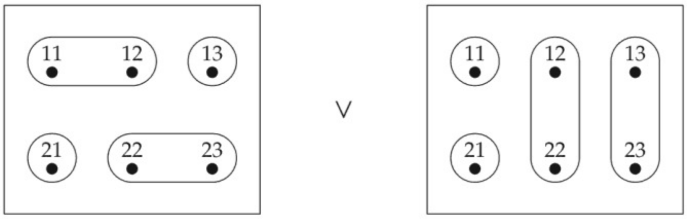
Joining two systems A and B is performed simply by combining their connections. That is, we shall say the *join* of systems A and B, denoted $A \vee B$, has a connection between points x and y if there are some points $z_1, \ldots, z_n$ such that each of the following is true in at least one of A or B: x is connected to z_1, z_i is connected to z_{i+1}, and z_n is connected to y. In a three-point system, the above definition is overkill, but we want to say something that works for systems with any number of elements. The high-level way to say it is "take the transitive closure of the union of the connections in A and B." In our three-element system, it means for example that



and (1.2)



Exercise 1.2. What is the result of joining the following two systems?



◇

We are now ready to see the generative effect. We don't want to build it up too much – this example has been made as simple as possible – but we shall see that Alice's observation fails to preserve the join operation. We've been denoting her observation – measuring whether • and $*$ are connected – by the symbol Φ; it returns a boolean result, either `true` or `false`.

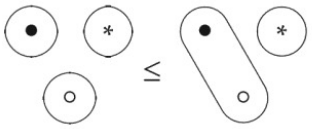
We see above in Eq. (1.1) that $\Phi($ $) = \Phi($ $) =$ `false`: in both cases • is not connected to $*$. On the other hand, when we join these two systems as in Eq. (1.2), we see that $\Phi($ $\vee$ $) = \Phi($ $) =$ `true`: in the joined system, • *is* connected to $*$. The question that Alice is interested in, that of Φ, is inherently lossy with respect to join, and there is no way to fix it without a more detailed observation, one that includes not only $*$ and • but also ∘.

While this was a simple example, it should be noted that whether the potential for such effects exist – i.e. determining whether an observation is operation-preserving – can be incredibly important information to know. For example, Alice could be in charge of putting together the views of two local authorities regarding possible contagion between an infected person • and a vulnerable person $*$. Alice has noticed that if they separately extract information from their raw data and combine the results, it gives a different answer than if they combine their raw data and extract information from it.

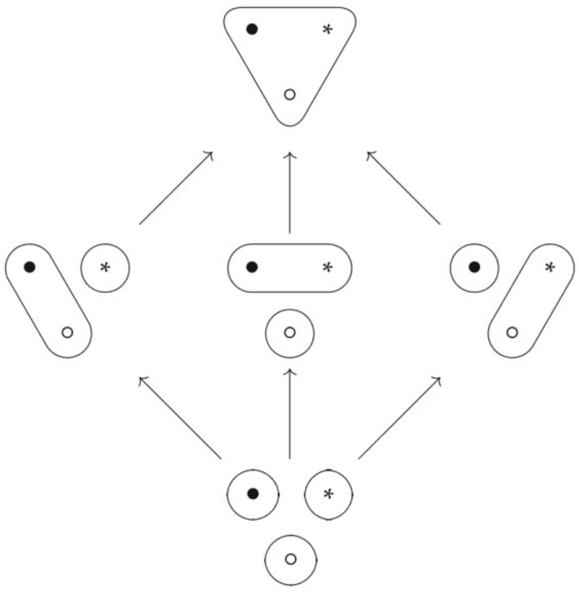
1.1.2 Ordering Systems

Category theory is all about organizing and layering structures. In this section we will explain how the operation of joining systems can be derived from a more basic structure: order. We shall see that while joining is not preserved by Alice's connectivity observation Φ, order is.

To begin, we note that the systems themselves are ordered in a hierarchy. Given systems A and B, we say that $A \leq B$ if, whenever x is connected to y in A, then x is connected to y in B. For example,



This notion of ≤ leads to the following diagram:



(1.3)

where an arrow from system A to system B means $A \leq B$. Such diagrams are known as *Hasse diagrams*.

As we were saying above, the notion of join is derived from this order. Indeed, for any two systems A and B in the Hasse diagram (1.3), the joined system $A \vee B$ is the smallest system that is bigger than both A and B. That is, $A \leq (A \vee B)$ and $B \leq (A \vee B)$, and for any C, if $A \leq C$ and $B \leq C$ then $(A \vee B) \leq C$. Let's walk through this with an exercise.

Exercise 1.3.

1. Write down all the partitions of a two-element set $\{\bullet, *\}$, order them as above, and draw the Hasse diagram.

2. Now do the same thing for a four-element set, say $\{1, 2, 3, 4\}$. There should be 15 partitions.

Choose any two systems in your 15-element Hasse diagram, call them A and B.

3. What is $A \vee B$, using the definition given in the paragraph above Eq. (1.2)?
4. Is it true that $A \leq (A \vee B)$ and $B \leq (A \vee B)$?
5. What are all the systems C for which both $A \leq C$ and $B \leq C$?
6. Is it true that in each case $(A \vee B) \leq C$? ◇

The set $\mathbb{B} = \{\mathtt{true}, \mathtt{false}\}$ of booleans also has an order, $\mathtt{false} \leq \mathtt{true}$:

$$\begin{tikzcd} \mathtt{true} \\ \mathtt{false} \arrow[u] \end{tikzcd}$$

Thus $\mathtt{false} \leq \mathtt{false}$, $\mathtt{false} \leq \mathtt{true}$, and $\mathtt{true} \leq \mathtt{true}$, but $\mathtt{true} \not\leq \mathtt{false}$. In other words, $A \leq B$ if A implies B.[2]

For any A, B in $\mathbb{B}$, we can again write $A \vee B$ to mean the least element that is greater than both A and B.

Exercise 1.4. Using the order $\mathtt{false} \leq \mathtt{true}$ on $\mathbb{B} = \{\mathtt{true}, \mathtt{false}\}$, what is:

1. $\mathtt{true} \vee \mathtt{false}$?
2. $\mathtt{false} \vee \mathtt{true}$?
3. $\mathtt{true} \vee \mathtt{true}$?
4. $\mathtt{false} \vee \mathtt{false}$? ◇

Let's return to our systems with $\bullet$, $\circ$, and $*$, and Alice's "$\bullet$ is connected to $*$" function, which we called Φ. It takes any such system and returns either $\mathtt{true}$ or $\mathtt{false}$. Note that the map Φ preserves the $\leq$ order: if $A \leq B$ and there is a connection between $\bullet$ and $*$ in A, then there is such a connection in B too. The possibility of a generative effect is captured in the inequality

$$\Phi(A) \vee \Phi(B) \leq \Phi(A \vee B). \tag{1.4}$$

We saw on page 4 that this can be a strict inequality: we showed two systems A and B with $\Phi(A) = \Phi(B) = \mathtt{false}$, so $\Phi(A) \vee \Phi(B) = \mathtt{false}$, but where $\Phi(A \vee B) = \mathtt{true}$. In this case, a generative effect exists.

These ideas capture the most basic ideas in category theory. Most directly, we have seen that the map Φ preserves some structure but not others: it preserves order but not join. In fact, we have seen here hints of more complex notions from category theory, without making them explicit; these include the notions of category, functor, colimit, and adjunction. In this chapter we will explore these ideas in the elementary setting of ordered sets.

[2] In mathematical logic, $\mathtt{false}$ implies $\mathtt{true}$ but $\mathtt{true}$ does not imply $\mathtt{false}$. That is "P implies Q" means, "if P is true, then Q is true too, but if P is not true, I'm making no claims."

1.2 What is Order?

Above we informally spoke of two different ordered sets: the order on system connectivity and the order on booleans `false` $\leq$ `true`. Then we related these two ordered sets by means of Alice's observation Φ. Before continuing, we need to make such ideas more precise. We begin in Section 1.2.1 with a review of sets and relations. In Section 1.2.2 we will give the definition of a *preorder* – short for preordered set – and a good number of examples.

1.2.1 Review of Sets, Relations, and Functions

We will not give a definition of *set* here, but informally we will think of a set as a collection of things, known as elements. These things could be all the leaves on a certain tree, or the names of your favorite fruits, or simply some symbols a, b, c. For example, we write $A = \{h, 1\}$ to denote the set, called A, that contains exactly two elements, one called h and one called 1. The set $\{h, h, 1, h, 1\}$ is exactly the same as A because they both contain the same elements, h and 1, and repeating an element more than once in the notation doesn't change the set.[3] For an arbitrary set X, we write $x \in X$ if x is an element of X; so we have $h \in A$ and $1 \in A$, but $0 \notin A$.

Example 1.5. 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 nth *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.

Given sets X and Y, we say that X is a *subset* of Y, and write $X \subseteq Y$, if every element in X is also in Y. For example $\{h\} \subseteq A$. Note that the empty set $\varnothing := \{\}$ is a subset of every other set.[4] Given a set Y and a property P that is either true or false for each element of Y, we write $\{y \in Y \mid P(y)\}$ to mean the subset of those y's that satisfy P.

Exercise 1.6.

1. Is it true that $\mathbb{N} = \{n \in \mathbb{Z} \mid n \geq 0\}$?

[3] If you want a notion where "$h, 1$" is different from "$h, h, 1, h, 1$," you can use something called *bags*, where the number of times an element is listed matters, or *lists*, where order also matters. All of these are important concepts in applied category theory, but sets will come up the most for us.

[4] When we write $Z :=$ `foo`, it means "assign the meaning `foo` to variable Z," whereas $Z =$ `foo` means simply that Z is equal to `foo`, perhaps as discovered via some calculation. In particular, $Z :=$ `foo` implies $Z =$ `foo` but not vice versa; indeed it *would not* be proper to write $3 + 2 := 5$ or $\{\} := \varnothing$.

Definition 1.13. 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[6] $b \sim a$, for all $a, b \in A$,
(c) if $a \sim b$ and $b \sim c$ then $a \sim c$, for all $a, b, c \in A$.

These properties are called *reflexivity*, *symmetry*, and *transitivity*, respectively.

Proposition 1.14. Let A be a set. There is a one-to-one correspondence between the ways to partition A and the equivalence relations on A.

Proof. We first show that every partition gives rise to an equivalence relation, and then that every equivalence relation gives rise to a partition. Our two constructions will be mutually inverse, proving the proposition.

Suppose we are given a partition $\{A_p\}_{p \in P}$; we define a relation $\sim$ and show it is an equivalence relation. Define $a \sim b$ to mean that a and b are in the same part: there is some $p \in P$ such that $a \in A_p$ and $b \in A_p$. It is obvious that a is in the same part as itself. Similarly, it is obvious that if a is in the same part as b then b is in the same part as a, and that if further b is in the same part as c then a is in the same part as c. Thus $\sim$ is an equivalence relation as defined in Definition 1.13.

Suppose we are given an equivalence relation $\sim$; we will form a partition on A by saying what the parts are. Say that a subset $X \subseteq A$ is $(\sim)$-closed if, for every $x \in X$ and $x' \sim x$, we have $x' \in X$. Say that a subset $X \subseteq A$ is $(\sim)$-connected if it is nonempty and $x \sim y$ for every $x, y \in X$. Then the parts corresponding to $\sim$ are exactly the $(\sim)$-closed, $(\sim)$-connected subsets. It is not hard to check that these indeed form a partition. $\square$

Exercise 1.15. Let's complete the "it's not hard to check" part in the proof of Proposition 1.14. Suppose that $\sim$ is an equivalence relation on a set A, and let P be the set of $(\sim)$-closed and $(\sim)$-connected subsets $\{A_p\}_{p \in P}$.

1. Show that each part A_p is nonempty.
2. Show that if $p \neq q$, i.e. if A_p and A_q are not exactly the same set, then $A_p \cap A_q = \varnothing$.
3. Show that $A = \bigcup_{p \in P} A_p$. $\diamond$

Definition 1.16. 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.

Functions

The most frequently used sort of relation between sets is that of functions.

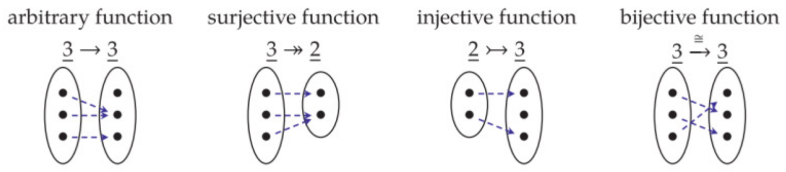
[6] "Iff" is short for "if and only if."

Definition 1.17. Let S and T be sets. A *function from S to T* is a subset $F \subseteq S \times T$ such that for all $s \in S$, there exists a unique $t \in T$ with $(s, t) \in F$.

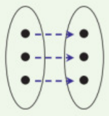
The function F is often denoted $F \colon S \to T$. From now on, we write $F(s) = t$, or sometimes $s \mapsto t$, to mean $(s, t) \in F$. For any $t \in T$, the *preimage of t along F* is the subset $f^{-1}(t) := \{s \in S \mid F(s) = t\}$.

A function is called *surjective*, or a *surjection*, if for all $t \in T$, there exists $s \in S$ with $F(s) = t$. A function is called *injective*, or an *injection*, if for all $t \in T$ and $s_1, s_2 \in S$ with $F(s_1) = t$ and $F(s_2) = t$, we have $s_1 = s_2$. A function is called *bijective* if it is both surjective and injective.

We use various decorations on arrows, $\rightarrow, \twoheadrightarrow, \rightarrowtail, \xrightarrow{\cong}$ to denote these special sorts of functions. Here is a table with the name, arrow decoration, and an example of each sort of function:


arbitrary function
$\underline{3} \to \underline{3}$
surjective function
$\underline{3} \twoheadrightarrow \underline{2}$
injective function
$\underline{2} \rightarrowtail \underline{3}$
bijective function
$\underline{3} \xrightarrow{\cong} \underline{3}$

Example 1.18. An important but very simple sort of function is the *identity function* on a set X, denoted id_X. It is the bijective function $\mathrm{id}_X(x) = x$.

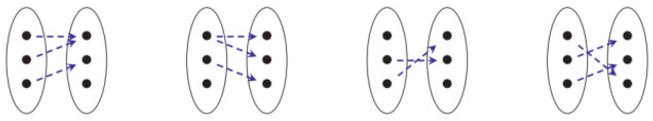


For notational consistency with Definition 1.17, the arrows in Example 1.18 might be drawn as $\mapsto$ rather than $\dashrightarrow$. The $\dashrightarrow$-style arrows were drawn because we thought it was prettier, i.e. easier on the eye. Beauty is important too; an imbalanced preference for strict correctness over beauty becomes *pedantry*. But, outside of pictures, we will be careful.

Exercise 1.19. In the following, do not use any examples already drawn above.

1. Find two sets A and B and a function $f \colon A \to B$ that is injective but not surjective.
2. Find two sets A and B and a function $f \colon A \to B$ that is surjective but not injective.

Now consider the four relations shown here:



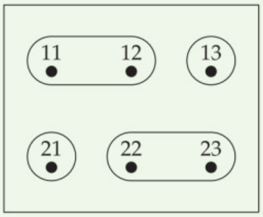
For each relation, answer the following two questions.

3. Is it a function?
4. If not, why not? If so, is it injective, surjective, both (i.e. bijective), or neither? ◇

Exercise 1.20. Suppose that A is a set and $f\colon A \to \varnothing$ is a function to the empty set. Show that A is empty. ◇

Example 1.21. A partition on a set A can also be understood in terms of surjective functions out of A. Given a surjective function $f\colon A \twoheadrightarrow P$, where P is any other set, the preimages $f^{-1}(p) \subseteq A$, one for each element $p \in P$, form a partition of A. Here is an example.

Consider the partition of $S \coloneqq \{11, 12, 13, 21, 22, 23\}$ shown below:

$S \coloneqq$ 

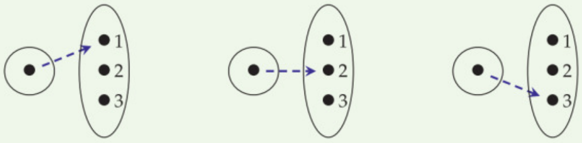
It has been partitioned into four parts, so let $P = \{a, b, c, d\}$ and let $f\colon S \twoheadrightarrow P$ be given by

$$f(11) = a, \quad f(12) = a, \quad f(13) = b, \quad f(21) = c, \quad f(22) = d, \quad f(23) = d.$$

Exercise 1.22. Write down a surjection corresponding to each of the five partitions in Eq. (1.3). ◇

Definition 1.23. If $F\colon X \to Y$ is a function and $G\colon Y \to Z$ is a function, their *composite* is the function $X \to Z$ defined to be $G(F(x))$ for any $x \in X$. It is often denoted $G \circ F$, but we prefer to denote it $F \fatsemi G$. It takes any element $x \in X$, evaluates F to get an element $F(x) \in Y$ and then evaluates G to get an element $G(F(x))$.

Example 1.24. If X is any set and $x \in X$ is any element, we can think of x as a function $\{1\} \to X$, namely the function sending 1 to x. For example, the three functions $\{1\} \to \{1, 2, 3\}$ shown below correspond to the three elements of $\{1, 2, 3\}$:



Suppose we are given a function $F\colon X \to Y$ and an element of X, thought of as a function $x\colon \{1\} \to X$. Then evaluating F at x is given by the composite $F(x) = x \fatsemi F$.

1.2.2 Preorders

In Section 1.1, we often used the symbol $\leq$ to denote a sort of order. Here is a formal definition of what it means for a set to have an order.

Definition 1.25. A *preorder relation* on a set X is a binary relation on X, here denoted with infix notation $\leq$, such that

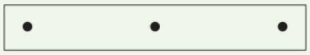
(a) $x \leq x$; and
(b) if $x \leq y$ and $y \leq z$, then $x \leq z$.

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*.

Remark 1.26. Observe that reflexivity and transitivity are familiar from Definition 1.13: equivalence relations are preorders with an additional symmetry condition.

Example 1.27 (Discrete preorders). Every set X can be considered as a discrete preorder $(X, =)$. This means that the only order relationships on X are of the form $x \leq x$; if $x \neq y$ then neither $x \leq y$ nor $y \leq x$ hold.

We depict discrete preorders as simply a collection of points:



Example 1.28 (Codiscrete preorders). From every set we may also construct its codiscrete preorder $(X, \leq)$ by equipping it with the total binary relation $X \times X \subseteq X \times X$. This is a very trivial structure: it means that for *all* x and y in X we have $x \leq y$ (and hence also $y \leq x$).

Example 1.29 (Booleans). The booleans $\mathbb{B} = \{\texttt{false}, \texttt{true}\}$ form a preorder with $\texttt{false} \leq \texttt{true}$.

$$\begin{tikzcd} \texttt{true} \\ \texttt{false} \arrow[u] \end{tikzcd}$$

Remark 1.30 (Partial orders are skeletal preorders). A preorder is a *partial order* if we additionally have that

(c) $x \cong y$ implies $x = y$.

In category theory terminology, the requirement that $x \cong y$ implies $x = y$ is known as *skeletality*, so partial orders are *skeletal preorders*. For short, we also use the term *poset*, a contraction of partially ordered set.

The difference between preorders and partial orders is rather minor. A partial order already is a preorder, and every preorder can be made into a partial order by equating any two elements x, y for which $x \cong y$, i.e. for which $x \leq y$ and $y \leq x$.

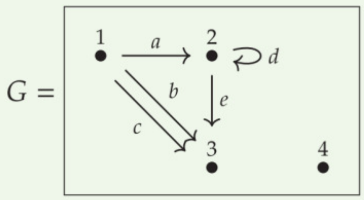
For example, any discrete preorder is already a partial order, while any codiscrete preorder simply becomes the unique partial order on a one-element set.

We have already introduced a few examples of preorders using Hasse diagrams. It will be convenient to continue to do this, so let us be a bit more formal about what we mean. First, we need to define a graph.

Definition 1.31. A *graph* $G = (V, A, s, t)$ consists of a set V whose elements are called *vertices*, a set A whose elements are called *arrows*, and two functions $s, t \colon A \to V$ known as the *source* and *target* functions respectively. Given $a \in A$ with $s(a) = v$ and $t(a) = w$, we say that a is an arrow from v to w.

By a *path* in G we mean any sequence of arrows such that the target of one arrow is the source of the next. This includes sequences of length 1, which are just arrows $a \in A$ in G, and sequences of length 0, which just start and end at the same vertex v, without traversing any arrows.

Example 1.32. Here is a picture of a graph:



It has $V = \{1, 2, 3, 4\}$ and $A = \{a, b, c, d, e\}$. The source and target functions, $s, t \colon A \to V$ are given by the following partially filled-in tables (see Exercise 1.33):

Arrow a	**source** $s(a) \in V$	**target** $t(a) \in V$
a	1	?
b	1	3
c	?	?
d	?	?
e	?	?

For example, taking $X = \{0, 1, 2\}$, we depict $\mathsf{P}(X)$ as

$$\begin{tikzcd}
 & X & \\
\{0,1\} \arrow[ur] & \{0,2\} \arrow[u] & \{1,2\} \arrow[ul] \\
\{0\} \arrow[u] \arrow[ur] & \{1\} \arrow[ul] \arrow[ur] & \{2\} \arrow[ul] \arrow[u] \\
 & \varnothing \arrow[ul] \arrow[u] \arrow[ur] &
\end{tikzcd}$$

See the cube? The Hasse diagram for the power set of a finite set, say $\mathsf{P}\{1, 2, \ldots, n\}$,[7] always looks like a cube of dimension n.

Exercise 1.46. Draw the Hasse diagrams for $\mathsf{P}(\varnothing)$, $\mathsf{P}\{1\}$, and $\mathsf{P}\{1, 2\}$. ◇

Example 1.47 (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. (1.3). Namely, we order 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 1.21 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 $f \mathbin{;} h = g$.

Exercise 1.48. For any set S there is a coarsest partition, having just one part. What surjective function does it correspond to?

There is also a finest partition, where everything is in its own part. What surjective function does it correspond to? ◇

Example 1.49 (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 1.29), then its preorder of upper sets $\mathsf{U}(\mathbb{B})$ is

$$\begin{tikzcd}
\{\texttt{true}, \texttt{false}\} \\
\{\texttt{true}\} \arrow[u] \\
\varnothing \arrow[u]
\end{tikzcd}$$

[7] Note that we omit the parentheses here, writing $\mathsf{P}X$ instead of $\mathsf{P}(X)$; throughout this book we will omit parentheses if we judge the presentation is cleaner and it is unlikely to cause confusion.

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

Exercise 1.50. Prove that the preorder of upper sets on a discrete preorder (see Example 1.27) on a set X is simply the power set $\mathsf{P}(X)$. ◇

Example 1.51 (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.

Exercise 1.52. Draw the Hasse diagram for the product of the two preorders drawn below:

$$\begin{tikzcd} c & & b & & 2 \\ & a \arrow[ul] \arrow[ur] & & & 1 \arrow[u] \end{tikzcd}$$

For bonus points, compute the upper set preorder on the result. ◇

Example 1.53 (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 $q \leq p$.

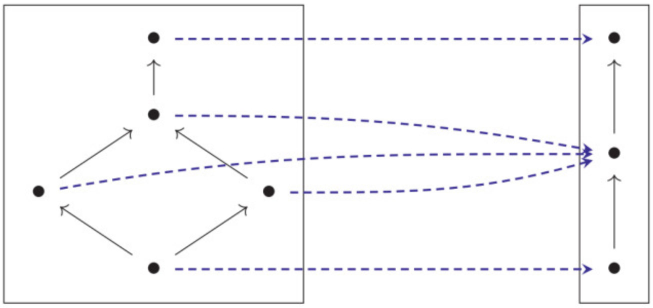
1.2.3 Monotone Maps

We have said that the categorical perspective emphasizes relationships between things. For example, a preorder is a setting – or world – in which we have one sort of relationship, $\leq$, and any two objects may be, or may not be, so-related. Jumping up a level, the categorical perspective emphasizes that preorders themselves – each a miniature world composed of many relationships – can be related to one another.

The most important sort of relationship between preorders is called a *monotone map*. These are functions that preserve preorder relations – in some sense mappings that respect $\leq$ – and are hence considered the right notion of *structure-preserving map* for preorders.

Definition 1.54. A *monotone map* between preorders $(A, \leq_A)$ and $(B, \leq_B)$ is a function $f \colon A \to B$ such that, for all elements $x, y \in A$, if $x \leq_A y$ then $f(x) \leq_B f(y)$.

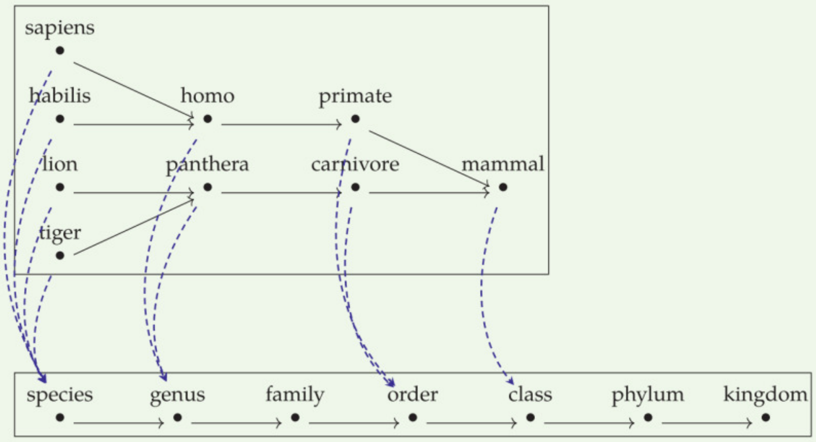
A monotone map $A \to B$ between two preorders associates to each element of preorder A an element of the preorder B. We depict this by drawing a dotted arrow from each element $x \in A$ to its image $f(x) \in B$. Note that the order must be preserved in order to count as a valid monotone map, so if element x is above element y in the left-hand preorder A, then the image $f(x)$ will be above the image $f(y)$ in the right-hand preorder.



Example 1.55. Let $\mathbb{B}$ and $\mathbb{N}$ be the preorders of booleans from Example 1.29 and $\mathbb{N}$ be the preorder of natural numbers from Example 1.40. The map $\mathbb{B} \to \mathbb{N}$ sending `false` to 17 and `true` to 24 is a monotone map, because it preserves order.

$$\begin{tikzcd}
 & & & & \texttt{false} \arrow[r] \arrow[dl, dashed] & \texttt{true} \arrow[drr, dashed] & & & \\
0 \arrow[r] & 1 \arrow[r] & \cdots \arrow[r] & 17 \arrow[r] & 18 \arrow[r] & \cdots \arrow[r] & 23 \arrow[r] & 24 \arrow[r] & \cdots
\end{tikzcd}$$

Example 1.56 (The tree of life). Consider the set of all animal classifications, for example "tiger," "mammal," "sapiens," "carnivore," etc. These are ordered by specificity: since "tiger" is a type of "mammal," we write tiger $\leq$ mammal. The result is a preorder, which in fact forms a tree, often called the tree of life. At the top of the following diagram we see a small part of it:



At the bottom we see the hierarchical structure as a preorder. The dashed arrows show a monotone map, call it F, from the classifications to the hierarchy. It is monotone because it preserves order: whenever there is a path $x \to y$ upstairs, there is a path $F(x) \to F(y)$ downstairs.

Example 1.57. Given a finite set X, recall the power set $\mathsf{P}(X)$ and its natural order relation from Example 1.45. The map $|\cdot|\colon \mathsf{P}(X) \to \mathbb{N}$ sending each subset S to its number of elements $|S|$, also called its *cardinality*, is a monotone map.

Exercise 1.58. Let $X = \{0, 1, 2\}$.

1. Draw the Hasse diagram for $\mathsf{P}(X)$.
2. Draw the Hasse diagram for the preorder $0 \leq 1 \leq 2 \leq 3$.
3. Draw the cardinality map $|\cdot|$ from Example 1.57 as dashed lines between them. ◇

Example 1.59. Recall the notion of upper set from Example 1.49. Given a preorder $(P, \leq)$, the map $\mathsf{U}(P) \to \mathsf{P}(P)$ sending each upper set of $(P, \leq)$ to itself – considered as a subset of P – is a monotone map.

Exercise 1.60. Consider the preorder $\mathbb{B}$. The Hasse diagram for $\mathsf{U}(\mathbb{B})$ was drawn in Example 1.49, and you drew the Hasse diagram for $\mathsf{P}(\mathbb{B})$ in Exercise 1.46. Now draw the monotone map between them, as described in Example 1.59. ◇

Exercise 1.61. Let $(P, \leq)$ be a preorder, and recall the notion of opposite preorder from Example 1.53.

1. Show that the set $\uparrow p := \{p' \in P \mid p \leq p'\}$ is an upper set, for any $p \in P$.
2. Show that this construction defines a monotone map $\uparrow\colon P^{\text{op}} \to \mathsf{U}(P)$.
3. Show that $p \leq p'$ in P if and only if $\uparrow(p') \subseteq \uparrow(p)$.
4. Draw a picture of the map $\uparrow$ in the case where P is the preorder $(b \geq a \leq c)$ from Exercise 1.52.

This is known as the *Yoneda lemma* for preorders. The if and only if condition proved in part 3 implies that, up to equivalence, knowing an element p is the same as knowing its upper set P– that is, knowing its web of relationships with the other elements of the preorder. The general Yoneda lemma is a powerful tool in category theory, and a fascinating philosophical idea besides. ◇

Exercise 1.62. As you know, a monotone map $f\colon (P, \leq_P) \to (Q, \leq_Q)$ consists of a function $f\colon P \to Q$ that satisfies a "monotonicity" property. Show that when $(P, \leq_P)$ is a discrete preorder, then *every* function $P \to Q$ satisfies the monotonicity property, regardless of the order $\leq_Q$. ◇

Example 1.63. Recall from Example 1.47 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 $s\colon X \twoheadrightarrow P$ for some set P.

Any surjective function $f\colon X \twoheadrightarrow Y$ induces a monotone map $f^*\colon \mathsf{Prt}(Y) \to \mathsf{Prt}(X)$, going "backwards." It is defined by sending a partition $s\colon Y \twoheadrightarrow P$ to the composite $f \mathbin{⨾} s\colon X \twoheadrightarrow P$.[8]

Exercise 1.64. Choose two sets X and Y with at least three elements each and choose a surjective, non-identity function $f\colon X \twoheadrightarrow Y$ between them. Write down two different partitions P and Q of Y, and then find $f^*(P)$ and $f^*(Q)$. ◇

The following proposition, Proposition 1.65, is straightforward to check. Recall the definition of the identity function from Example 1.18 and the definition of composition from Definition 1.23.

Proposition 1.65. For any preorder $(P, \leq_P)$, the identity function is monotone.

If $(Q, \leq_Q)$ and $(R, \leq_R)$ are preorders and $f\colon P \to Q$ and $g\colon Q \to R$ are monotone, then $(f \mathbin{⨾} g)\colon P \to R$ is also monotone.

Exercise 1.66. Check the two claims made in Proposition 1.65. ◇

Example 1.67. Recall again the definition of opposite preorder from Example 1.53. The identity function $\mathrm{id}_P\colon P \to P$ is a monotone map $(P, \leq) \to (P, \leq^{\mathrm{op}})$ if and only if for all $p, q \in P$ we have $q \leq p$ whenever $p \leq q$. For historical reasons connected to linear algebra, when this is true, we call $(P, \leq)$ a *dagger preorder*.

But in fact, we have seen dagger preorders before in another guise. Indeed, if $(P, \leq)$ is a dagger preorder, then the relation $\leq$ is symmetric: $p \leq q$ if and only if $q \leq p$, and it is also reflexive and transitive by definition of preorder. So in fact $\leq$ is an equivalence relation (Definition 1.13).

Exercise 1.68. Recall the notion of skeletal preorders (Remark 1.30) and discrete preorders (Example 1.27). Show that a skeletal dagger preorder is just a discrete preorder, and hence can be identified with a set. ◇

Remark 1.69. We say that an A "can be identified with" a B when any A gives us a unique B and any B gives us a unique A, and both round-trips – from an A to a B and back to an A, or from a B to an A and back to a B – return us where we started. For example, any discrete preorder $(P, \leq)$ has an underlying set P, and any set P can be made into a discrete preorder ($p_1 \leq p_2$ iff $p_1 = p_2$), and the round-trips return us where we started. So what's the difference? It's like the notion of *object-permanence*

[8] We shall later see that any function $f\colon X \to Y$, not necessarily surjective, induces a monotone map $f^*\colon \mathsf{Prt}(Y) \to \mathsf{Prt}(X)$, but it involves an extra step. See Section 1.4.2.

Definition 1.76. 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 $p \leq a$,
(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 $a \leq p$,
(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 $p = a \vee b$.

Remark 1.77. In Definition 1.76, we committed a seemingly egregious abuse of notation. We shall see next in Example 1.79 that there could be two different meets of $A \subseteq P$, say $p = \bigwedge A$ and $q = \bigwedge A$ with $p \neq q$, which does not make sense if $p \neq q$!

But in fact, as we use the symbol $\bigwedge A$, this abuse won't matter because any two meets p, q are automatically isomorphic: the very definition of meet forces both $p \leq q$ and $q \leq p$, and thus we have $p \cong q$. So, for any $x \in P$, we have $p \leq x$ iff $q \leq x$ and $x \leq p$ iff $x \leq q$. Thus as long as we are only interested in elements of P based on their relationships to other elements (and in category theory, this is the case: we should only care about things based on how they interact with other things, rather than on some sort of "internal essence"), the distinction between p and q will never matter.

This foreshadows a major theme of – as well as standard abuse of notation in – category theory, where any two things defined by the same universal property are automatically equivalent in a way known as "unique up to unique isomorphism"; this means that we generally do not run into trouble if we pretend they are equal. We'll pick up this theme of "the" vs. "a" again in Remark 3.70.

Example 1.78 (Meets or joins may not exist). Note that, in an arbitrary preorder $(P, \leq)$, a subset A need not have a meet or a join. Consider the three-element set $P = \{p, q, r\}$ with the discrete ordering. The set $A = \{p, q\}$ does not have a join in P because if x was a join, we would need $p \leq x$ and $q \leq x$, and there is no such element x.

Example 1.79 (Multiple meets or joins may exist). It may also be the case that a subset A has more than one meet or join. Here is an example.

$$\begin{tikzcd}
\overset{a}{\bullet} & \overset{b}{\bullet} \\
\overset{c}{\bullet} \arrow[u] \arrow[ur] \arrow[r, bend right] & \overset{d}{\bullet} \arrow[u] \arrow[ul] \arrow[l, bend right]
\end{tikzcd}$$

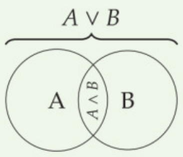
Let A be the subset $\{a, b\}$ in the preorder specified by this Hasse diagram. Then both c and d are meets of A: any element less than both a and b is also less than c, and also less than d. Note that, as in Remark 1.77, $c \leq d$ and $d \leq c$, so $c \cong d$. Such will always the case when there is more than one meet: any two meets of the same subset will be isomorphic.

Exercise 1.80. Let $(P, \leq)$ be a preorder and $p \in P$ an element. Consider the set $A = \{p\}$ with one element.

1. Show that $\bigwedge A \cong p$.
2. Show that if P is in fact a partial order, then $\bigwedge A = p$.
3. Are the analogous facts true when $\bigwedge$ is replaced by $\bigvee$? $\diamond$

Example 1.81. In any partial order P, we have $p \vee p = p \wedge p = p$. The reason is that our notation says $p \vee p$ means $\bigvee\{p, p\}$. But $\{p, p\} = \{p\}$ (see Section 1.2.1), so $p \vee p = p$ by Exercise 1.80.

Example 1.82. In a power set $\mathsf{P}(X)$, the meet of a collection of subsets, say $A, B \subseteq X$, is their intersection $A \wedge B = A \cap B$, while the join is their union, $A \vee B = A \cup B$.



Perhaps this justifies the terminology: the joining of two sets is their union, the meeting of two sets is their intersection.

Example 1.83. In the booleans $\mathbb{B} = \{\texttt{false}, \texttt{true}\}$ (Example 1.29), the meet of any two elements is given by AND and the join of any two elements is given by OR (recall Exercise 1.4).

Example 1.84. In a total order, the meet of a set is its infimum, while the join of a set is its supremum. Note that $\mathbb{B}$ is a total order, and this generalizes Example 1.83.

Exercise 1.85. Recall the division ordering on $\mathbb{N}$ from Example 1.40: we write $n|m$ if n divides perfectly into m. The meet of any two numbers in this preorder has a common name, which you may have learned when you were around 10 years old; what is it? Similarly the join of any two numbers has a common name; what is it? ◇

Proposition 1.86. Suppose $(P, \leq)$ is a preorder and $A \subseteq B \subseteq P$ are subsets that have meets. Then $\bigwedge B \leq \bigwedge A$.

Similarly, if A and B have joins, then $\bigvee A \leq \bigvee B$.

Proof. Let $m = \bigwedge A$ and $n = \bigwedge B$. Then for any $a \in A$ we also have $a \in B$, so $n \leq a$ because n is a lower bound for B. Thus n is also a lower bound for A and hence $n \leq m$, because m is A's greatest lower bound. The second claim is proved similarly. □

1.3.2 Back to Observations and Generative Effects

In his thesis [Ada17], Adam thinks of monotone maps as observations. A monotone map $\Phi\colon P \to Q$ is a phenomenon (we might say "feature") of P as observed by Q. Adam defines the *generative effect* of such a map Φ to be its failure to preserve joins (or more generally, for categories, its failure to preserve colimits).

Definition 1.87. We say that a monotone map $f\colon P \to Q$ *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 1.88. We say that a monotone map $f\colon P \to Q$ *has a generative effect* if there exist elements $a, b \in P$ such that

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

In Definition 1.88, if we think of Φ as an observation or measurement of the systems a and b, then the left-hand side $f(a) \vee f(b)$ may be interpreted as the combination of the observation of a with the observation of b. On the other hand, the right-hand side $f(a \vee b)$ is the observation of the combined system $a \vee b$. The inequality implies that we see something when we observe the combined system that we could not expect by merely combining our observations of the pieces. That is, that there are generative effects from the interconnection of the two systems.

Exercise 1.89. In Definition 1.88, we defined generativity of f as the inequality $f(a \vee b) \neq f(a) \vee f(b)$, but in the subsequent text we seemed to imply there would be not just a difference, but *more stuff* in $f(a \vee b)$ than in $f(a) \vee f(b)$.

Prove that for any monotone map $f: P \to Q$, if $a, b \in P$ have a join and $f(a), f(b) \in Q$ have a join, then indeed $f(a) \vee f(b) \leq f(a \vee b)$. ◇

In his work on generative effects, Adam restricts his attention to generative maps that preserve meets (but do not preserve joins). The preservation of meets implies that the map Φ behaves well when restricting to subsystems, even though it can throw up surprises when joining systems.

This discussion naturally leads into Galois connections, which are pairs of monotone maps between preorders, one of which preserves all joins and the other of which preserves all meets.

1.4 Galois Connections

The preservation of meets and joins, and in particular issues concerning generative effects, is tightly related to the theory of *Galois connections*, which is a special case of a more general theory we will discuss later, namely that of *adjunctions*. We will use some adjunction terminology when describing Galois connections.

1.4.1 Definition and Examples of Galois Connections

Galois connections between preorders were first considered by Évariste Galois – who didn't call them by that name – in the context of a connection he found between "field extensions" and "automorphism groups." We will not discuss this further, but the idea is that, given two preorders P and Q, a Galois connection is a pair of maps back and forth – from P to Q and from Q to P – with certain properties, which make it like a relaxed version of isomorphisms. To be a bit more precise, preorder isomorphisms are examples of Galois connections, but Galois connections need not be preorder isomorphisms.

Definition 1.90. 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

$$f(p) \leq q \quad \text{if and only if} \quad p \leq g(q). \tag{1.6}$$

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

Example 1.91. Consider the map $(3 \times -): \mathbb{Z} \to \mathbb{R}$ which sends $x \in \mathbb{Z}$ to $3x$, which we can consider as a real number $3x \in \mathbb{Z} \subseteq \mathbb{R}$. Let's find a left adjoint for the map $(3 \times -)$.

Write $\lceil z \rceil$ for the smallest natural number above $z \in \mathbb{R}$, and write $\lfloor z \rfloor$ for the largest integer below $z \in \mathbb{R}$, e.g. $\lceil 3.14 \rceil = 4$ and $\lfloor 3.14 \rfloor = 3$.[9] As the left adjoint $\mathbb{R} \to \mathbb{Z}$, let's see if $\lceil -/3 \rceil$ works.

It is easily checked that

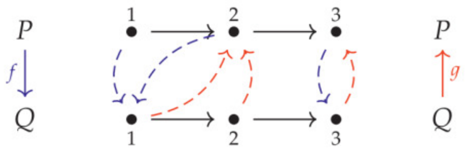
$$\lceil x/3 \rceil \le y \text{ if and only if } x \le 3y.$$

Success! Thus we have a Galois connection between $\lceil -/3 \rceil$ and $(3 \times -)$.

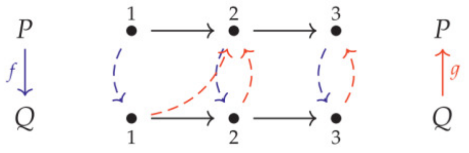
Exercise 1.92. In Example 1.91 we found a left adjoint for the monotone map $(3 \times -) \colon \mathbb{Z} \to \mathbb{R}$. Now find a right adjoint for the same map, and show it is correct. ◇

Exercise 1.93. Consider the preorder $P = Q = \underline{3}$.

1. Let f, g be the monotone maps shown below:



   Is it the case that f is left adjoint to g? Check that for each $1 \le p, q \le 3$, one has $f(p) \le q$ iff $p \le g(q)$.
2. Let f, g be the monotone maps shown below:



   Is it the case that f is left adjoint to g? ◇

Remark 1.94. The diagrams in Exercise 1.93 suggest the following idea. If P and Q are total orders and $f \colon P \to Q$ and $g \colon Q \to P$ are drawn with arrows bending counterclockwise, then f is left adjoint to g iff the arrows do not cross. With a little bit of thought, this can be formalized. We think this is a pretty neat way of visualizing Galois connections between total orders!

Exercise 1.95.

1. Does $\lceil -/3 \rceil$ have a left adjoint $L \colon \mathbb{Z} \to \mathbb{R}$?
2. If not, why? If so, does its left adjoint have a left adjoint? ◇

[9] By "above" and "below," we mean *greater than or equal to* or *less than or equal to*; the latter being a mouthful. Anyway, $\lfloor 3 \rfloor = 3 = \lceil 3 \rceil$.

Proof. Suppose f is left adjoint to g. Take any $p \in P$, and let $q := f(p)$. By reflexivity, we have $f(p) \leq q$, so by Definition 1.90 of Galois connection we have $p \leq g(q)$, but this means $p \leq g(f(p))$. The proof that $f(g(q)) \leq q$ is similar.

Now suppose that Eq. (1.7) holds for all $p \in P$ and $q \in Q$. We want to show that $f(p) \leq q$ iff $p \leq g(q)$. Suppose $f(p) \leq q$; then since g is monotonic, $g(f(p)) \leq g(q)$, but $p \leq g(f(p))$ so $p \leq g(q)$. The proof that $p \leq g(q)$ implies $f(p) \leq q$ is similar. □

Exercise 1.102. Complete the proof of Proposition 1.101 by showing that

1. if f is left adjoint to g then for any $q \in Q$, we have $f(g(q)) \leq q$,
2. if Eq. (1.7) holds, then $p \leq g(q)$ iff $f(p) \leq q$ holds, for all $p \in P$ and $q \in Q$. ◇

If we replace $\leq$ with $=$ in Eq. (1.7), we get back the definition of isomorphism (Definition 1.70); this is why we said at the beginning of Section 1.4.1 that Galois connections are a kind of relaxed version of isomorphisms.

Exercise 1.103.

1. Show that if $f \colon P \to Q$ has a right adjoint g, then it is unique up to isomorphism. That means, for any other right adjoint g', we have $g(q) \cong g'(q)$ for all $q \in Q$.
2. Is the same true for left adjoints? That is, if $h \colon P \to Q$ has a left adjoint, is it necessarily unique up to isomorphism? ◇

Proposition 1.104 (Right adjoints preserve meets)**.** Let $f \colon P \to Q$ be left adjoint to $g \colon Q \to P$. Suppose that $A \subseteq Q$ is 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

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

Proof. Let $f \colon P \to Q$ and $g \colon Q \to P$ be adjoint monotone maps, with g right adjoint to f. Let $A \subseteq Q$ be any subset and let $m := \bigwedge A$ be its meet. Then since g is monotone $g(m) \leq g(a)$ for all $a \in A$, so $g(m)$ is a lower bound for the set $g(A)$. We will be done if we can show $g(m)$ is a greatest lower bound.

So take any other lower bound b for $g(A)$; that is, suppose that for all $a \in A$ we have $b \leq g(a)$ and we want to show that $b \leq g(m)$. Then by definition of g being a right adjoint (Definition 1.90), we also have $f(b) \leq a$. This means that $f(b)$ is a lower bound for A in Q. Since the meet m is the greatest lower bound, we have $f(b) \leq m$.

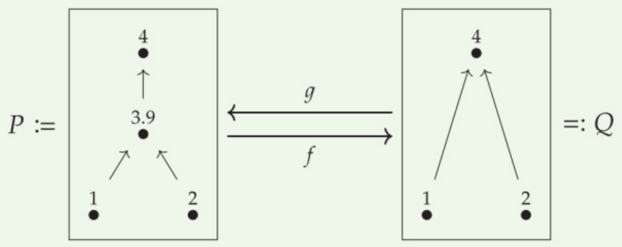
Once again using the Galois connection, $b \leq g(m)$, proving that $g(m)$ is indeed the greatest lower bound for $g(A)$, as desired.

The second claim is proved similarly; see Exercise 1.105. □

Exercise 1.105. Complete the proof of Proposition 1.104 by showing that left adjoints preserve joins. ◇

Since left adjoints preserve joins, we know that they cannot have generative effects. In fact, we shall see in Theorem 1.108 that a monotone map does not have generative effects – i.e. it preserves joins – if and only if it is a left adjoint to some other monotone.

Example 1.106. Right adjoints need not preserve joins. Here is an example:



Let g be the map that preserves labels, and let f be the map that preserves labels as far as possible but with $f(3.9) := 4$. Both f and g are monotonic, and one can check that g is right adjoint to f (see Exercise 1.107). But g does not preserve joins because $1 \vee 2 = 4$ holds in Q, whereas $g(1) \vee g(2) = 1 \vee 2 = 3.9 \neq 4 = g(4)$ in P.

Exercise 1.107. To be sure that g really is right adjoint to f in Example 1.106, there are twelve tiny things to check; do so. That is, for every $p \in P$ and $q \in Q$, check that $f(p) \leq q$ iff $p \leq g(q)$. ◇

Theorem 1.108 (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.

Proof. We will prove only the claim about meets; the claim about joins follows similarly.

We proved one direction in Proposition 1.104, namely that right adjoints preserve meets. For the other, suppose that g is a monotone map that preserves meets; we will construct a left adjoint f. We define our candidate $f\colon P \to Q$ on any $p \in P$ by

$$f(p) := \bigwedge\{q \in Q \mid p \leq g(q)\}; \tag{1.8}$$

this meet is well defined because Q has all meets, but for f to really be a candidate, we need to show it is monotone. So suppose that $p \leq p'$. Then $\{q' \in Q \mid p' \leq g(q')\} \subseteq \{q \in Q \mid p \leq g(q)\}$. By Proposition 1.86, this implies $f(p) \leq f(p')$. Thus f is monotone.

By Proposition 1.104, it suffices to show that $p_0 \leq g(f(p_0))$ and that $f(g(q_0)) \leq q_0$ for all $p_0 \in P$ and $q_0 \in Q$. For the first, we have

$$p_0 \leq \bigwedge\{g(q) \in P \mid p_0 \leq g(q)\} \cong g\left(\bigwedge\{q \in Q \mid p_0 \leq g(q)\}\right) = g(f(p_0)),$$

where the first inequality follows from the fact that if p_0 is below every element of a set, then it is below their meet, and the isomorphism is by definition of g preserving meets. For the second, we have

$$f(g(q_0)) = \bigwedge\{q \in Q \mid g(q_0) \leq g(q)\} \leq \bigwedge\{q_0\} = q_0,$$

where the first inequality follows from Proposition 1.86 since $\{q_0\} \subseteq \{q \in Q \mid g(q_0) \leq g(q)\}$, and the fact that $\bigwedge\{q_0\} = q_0$. □

Example 1.109. Let $f: A \to B$ be a function between sets. We can imagine A as a set of apples, B as a set of buckets, and f as putting each apple in a bucket.

Then we have the monotone map $f^*: \mathsf{P}(B) \to \mathsf{P}(A)$ that category theorists call "pullback along f." This map takes a subset $B' \subseteq B$ to its preimage $f^{-1}(B') \subseteq A$: that is, it takes a collection B' of buckets, and tells you all the apples that they contain in total. This operation is monotonic (more buckets means more apples) and it has both a left and a right adjoint. (There are two different adjunctions here involving f^*.)

The left adjoint $f_!(A)$ is given by the direct image: it maps a subset $A' \subseteq A$ to

$$f_!(A') := \{b \in B \mid \text{there exists } a \in A' \text{ such that } f(a) = b\}.$$

This map takes a set A' of apples, and tells you all the buckets that contain at least one of those apples.

The right adjoint f_* maps a subset $A' \subseteq A$ to

$$f_*(A') := \{b \in B \mid \text{for all } a \text{ such that } f(a) = b, \text{ we have } a \in A'\}.$$

This map takes a set A' of apples, and tells you all the buckets b that are all-A': all the apples in b are from the chosen subset A'. Note that if a bucket doesn't contain any apples at all, then vacuously all its apples are from A', so empty buckets count as far as f_* is concerned.

Notice that all three of these operations turn out to be interesting: start with a set B' of buckets and return all the apples in them, or start with a set A' of apples and find either the buckets that contain at least one apple from A', or the buckets whose only apples are from A'. But we did not invent these mappings f^*, $f_!$, and f_*: they were *induced* by the function f. They were automatic. It is one of the pleasures of category theory that adjoints so often turn out to have interesting semantic interpretations.

Exercise 1.110. Choose sets X and Y with between two and four elements each, and choose a function $f\colon X \to Y$.

1. Choose two different subsets $B_1, B_2 \subseteq Y$ and find $f^*(B_1)$ and $f^*(B_2)$.
2. Choose two different subsets $A_1, A_2 \subseteq X$ and find $f_!(A_1)$ and $f_!(A_2)$.
3. With the same $A_1, A_2 \subseteq X$, find $f_*(A_1)$ and $f_*(A_2)$. ◇

1.4.4 Closure Operators

Given a Galois connection with $f\colon P \to Q$ left adjoint to $g\colon Q \to P$, we may compose f and g to arrive at a monotone map $f \mathbin{⨾} g\colon P \to P$ from preorder P to itself. This monotone map has the property that $p \leq (f⨾g)(p)$, and that $(f⨾g⨾f⨾g)(p) \cong (f⨾g)(p)$ for any $p \in P$. This is an example of a *closure operator*.[10]

Exercise 1.111. Suppose that f is left adjoint to g. Use Proposition 1.101 to show the following.

1. $p \leq (f ⨾ g)(p)$.
2. $(f ⨾ g ⨾ f ⨾ g)(p) \cong (f ⨾ g)(p)$. To prove this, show inequalities in both directions, $\leq$ and $\geq$. ◇

Definition 1.112. A *closure operator* $j\colon P \to P$ on a preorder P is a monotone map such that for all $p \in P$ we have

(a) $p \leq j(p)$,
(b) $j(j(p)) \cong j(p)$.

Example 1.113. Here is an example of closure operators from computation, very roughly presented. Imagine computation as a process of rewriting input expressions to output expressions. For example, a computer can rewrite the expression 7 + 2 + 3 as the expression `12`. The set of arithmetic expressions has a partial order according to whether one expression can be rewritten as another.

We might think of a computer program, then, as a method of taking an expression and reducing it to another expression. So it is a map $j\colon \texttt{exp} \to \texttt{exp}$. It furthermore is desirable to require that this computer program is a closure operator. Monotonicity means that if an expression x can be rewritten into expression y, then the reduction $j(x)$ can be rewritten into $j(y)$. Moreover, the requirement $x \leq j(x)$ implies that j can only turn one expression into another if doing so is a permissible rewrite. The requirement $j(j(x)) = j(x)$ implies that if you try to reduce an expression that has already been reduced, the computer program leaves it as is. These properties provide useful structure in the analysis of program semantics.

[10] The other composite $g ⨾ f$ satisfies the dual properties: $(g ⨾ f)(q) \leq q$ and $(g ⨾ f ⨾ g ⨾ f)(q) \cong (g ⨾ f)(q)$ for all $q \in Q$. This is called an *interior operator*, though we will not discuss this concept further.

Example 1.114 (Adjunctions from closure operators). Just as every adjunction gives rise to a closure operator, from every closure operator we may construct an adjunction.

Let P be a preorder and let $j\colon P \to P$ be a closure operator. We can define a preorder fix_j to have elements the fixed points of j; that is,

$$\mathrm{fix}_j := \{p \in P \mid j(p) \cong p\}.$$

This is a subset of P and inherits an order as a result; hence fix_j is a sub-preorder of P. Note that $j(p)$ is a fixed point for all $p \in P$, since $j(j(p)) \cong j(p)$.

We define an adjunction with left adjoint $j\colon P \to \mathrm{fix}_j$ sending p to $j(p)$, and right adjoint $g\colon \mathrm{fix}_j \to P$ simply the inclusion of the sub-preorder. To see it's really an adjunction, we need to see that for any $p \in P$ and $q \in \mathrm{fix}_j$, we have $j(p) \leq q$ if and only if $p \leq q$. Let's check it. Since $p \leq j(p)$, we have that $j(p) \leq q$ implies $p \leq q$ by transitivity. Conversely, since q is a fixed point, $p \leq q$ implies $j(p) \leq j(q) \cong q$.

Example 1.115. Another example of closure operators comes from logic. This will be discussed in the final chapter of the book, in particular Section 7.4.5, but we will give a quick overview here. In essence, logic is the study of when one formal statement – or proposition – implies another. For example, if n is prime then n is not a multiple of 6, or if it is raining then the ground is getting wetter. Here "n is prime," "n is not a multiple of 6," "it is raining," and "the ground is getting wetter" are propositions, and we gave two implications.

Take the set of all propositions and order them by $p \leq q$ iff p implies q, denoted $p \Rightarrow q$. Since $p \Rightarrow p$ and since whenever $p \Rightarrow q$ and $q \Rightarrow r$, we also have $p \Rightarrow r$, this is indeed a preorder.

A closure operator on it is often called a *modal operator*. It is a function j from propositions to propositions, for which $p \Rightarrow j(p)$ and $j(j(p)) = j(p)$. An example of a j is "assuming Bob is in San Diego … ." Think of this as a proposition B; so "assuming Bob is in San Diego, p" means $B \Rightarrow p$. Let's see why $B \Rightarrow -$ is a closure operator.

If p is true then "assuming Bob is in San Diego, p" is still true. Suppose that "assuming Bob is in San Diego it is the case that, 'assuming Bob is in San Diego, p' is true." It follows that "assuming Bob is in San Diego, p" is true. So we have seen, at least informally, that "assuming Bob is in San Diego …" is a closure operator.

1.4.5 Level Shifting

The last thing we want to discuss in this chapter is a phenomenon that happens often in category theory, something we might informally call "level shifting." It is easier to give an example of this than to explain it directly.

2 Resource Theories: Monoidal Preorders and Enrichment

2.1 Getting from *a* to *b*

You can't make an omelette without breaking an egg. To obtain the things we want requires resources, and the process of transforming what we have into what we want is often an intricate one. In this chapter, we will discuss how monoidal preorders can help us think about this matter.

Consider the following three questions you might ask yourself:

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

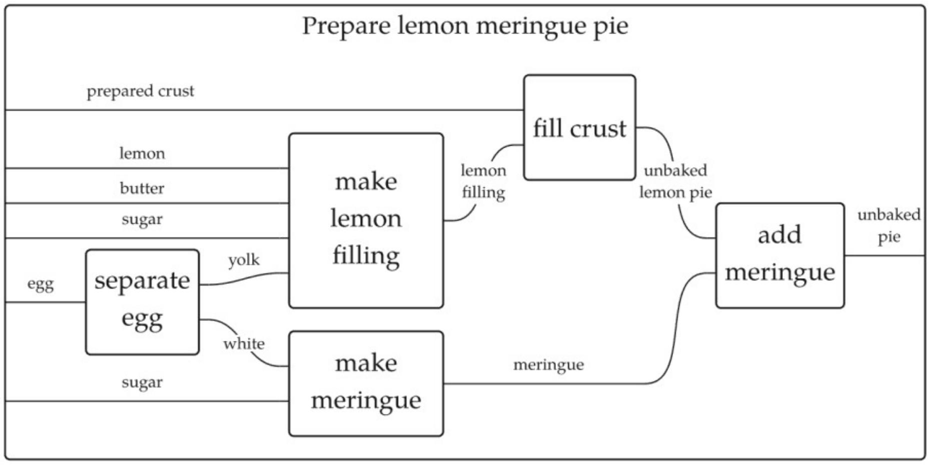
These questions are about resources – those you have and those you want – but, perhaps more importantly, they are about moving from have to want: possibility of, cost of, and ways to.

Such questions come up not only in our lives, but also in science and industry. In chemistry, one asks whether a certain set of compounds can be transformed into another set, how much energy such a reaction will require, or what methods exist for making it happen. In manufacturing, one asks similar questions.

From an external point of view, both a chemist and an industrial firm might be regarded as store-houses of information on the above subjects. The chemist knows which compounds she can make given other ones, and how to do so; the firm has stored knowledge of the same sort. The research work of the chemist and the firm is to use what they know in order to derive – or discover – new knowledge.

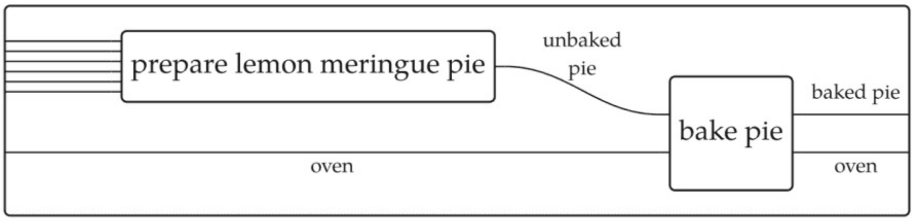
This is roughly the first goal of this chapter: to discuss a formalism for expressing recipes – methods for transforming one set of resources into another – and for deriving new recipes from old. The idea here is not complicated, neither in life nor in our mathematical formalism. The value added then is to simply see how it works, so we can build on it within the book, and so others can build on it in their own work.

We briefly discuss the categorical approach to this idea – namely that of *monoidal preorders* – for building new recipes from old. The following *wiring diagram* shows, assuming one knows how to implement each of the interior boxes, how to implement the preparation of a lemon meringue pie:



(2.1)

The wires show resources: we start with prepared crust, lemon, butter, sugar, and egg resources, and we end up with an unbaked pie resource. We could take this whole method and combine it with others, e.g. baking the pie:



In the above example we see that resources are not always consumed when they are used. For example, we use an oven to convert – or catalyze the transformation of – an unbaked pie into a baked pie, and we get the oven back after we are done. It's a nice feature of ovens! To use economic terms, the oven is a "means of production" for pies.

String diagrams are important mathematical objects that will come up repeatedly in this book. They were invented in the mathematical context – more specifically in the context of monoidal categories – by Joyal and Street [JS93], but they have been used less formally by engineers and scientists in various contexts for a long time.

As we said above, our first goal in this chapter is to use monoidal preorders, and the corresponding wiring diagrams, as a formal language for building new recipes from old. Our second goal is to discuss something called $\mathcal{V}$-categories for various monoidal preorders $\mathcal{V}$.

A $\mathcal{V}$-category is a set of objects, which one may think of as points on a map, where $\mathcal{V}$ somehow "structures the question" of getting from point a to point b. The examples of monoidal preorders $\mathcal{V}$ that we will be most interested in are called **Bool** and **Cost**. Roughly speaking, a **Bool**-category is a set of points where the question of getting from point a to point b has a `true` / `false` answer. A **Cost**-category is a set of points where the question of getting from a to b has an answer $d \in [0, \infty]$, a cost.

This story works in more generality than monoidal preorders. Indeed, in Chapter 4 we will discuss something called a monoidal category, a notion which generalizes monoidal preorders, and we will generalize the definition of $\mathcal{V}$-category accordingly. In this more general setting, $\mathcal{V}$-categories can also address our third question above, describing *methods* of getting between points. For example a **Set**-category is a set of points where the question of getting from point a to point b has a set of answers (elements of which might be called methods).

We will begin in Section 2.2 by defining symmetric monoidal preorders, giving a few preliminary examples and discussing wiring diagrams. We then give many more examples of symmetric monoidal preorders, including some real-world examples, in the form of resource theories, and some mathematical examples that will come up again throughout the book. In Section 2.3 we discuss enrichment and $\mathcal{V}$-categories – how a monoidal preorder $\mathcal{V}$ can "structure the question" of getting from a to b – and then give some important constructions on $\mathcal{V}$-categories (Section 2.4), and analyze them using a sort of matrix multiplication technique (Section 2.5).

2.2 Symmetric Monoidal Preorders

In Section 1.2.2 we introduced preorders. The notation for a preorder, namely $(X, \leq)$, refers to two pieces of structure: a set called X and a relation called $\leq$ that is reflexive and transitive.

We want to add to the concept of preorders a way of combining elements in X, an operation taking two elements and adding or multiplying them together. However, the operation does not have to literally be addition or multiplication; it only needs to satisfy some of the properties one expects from them.

2.2.1 Definition and First Examples

We begin with a formal definition of symmetric monoidal preorders.

Definition 2.1. A (strict) *symmetric monoidal structure* on a preorder $(X, \leq)$ consists of two constituents:

(i) an element $I \in X$, called the *monoidal unit*,
(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,
(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*.

Anyone can propose a set X, an order $\leq$ on X, an element I in X, and a binary operation $\otimes$ on X and ask whether $(X, \leq, I, \otimes)$ is a symmetric monoidal preorder. And it will indeed be one, as long as it satisfies rules (a), (b), (c), and (d) of Definition 2.1.

Remark 2.2. It is often useful to replace $=$ with $\cong$ throughout Definition 2.1. The result is a perfectly good notion, called a *weak monoidal structure*. The reason we chose equality is that it makes equations look simpler, which we hope aids first-time readers.

The notation for the monoidal unit and the monoidal product may vary: monoidal units we have seen include I (as in the definition), 0, 1, `true`, `false`, $\{*\}$, and more. Monoidal products we have seen include $\otimes$ (as in the definition), $+$, $*$, $\wedge$, $\vee$, and $\times$. The *preferred notation* in a given setting is whatever best helps our brains remember what we're trying to do; the names I and $\otimes$ are just defaults.

Example 2.3. There is a well-known preorder structure, denoted $\leq$, on the set $\mathbb{R}$ of real numbers; e.g. $-5 \leq \sqrt{2}$. We propose 0 as a monoidal unit and $+ : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ as a monoidal product. Does $(\mathbb{R}, \leq, 0, +)$ satisfy the conditions of Definition 2.1?

If $x_1 \leq y_1$ and $x_2 \leq y_2$, it is true that $x_1 + x_2 \leq y_1 + y_2$. It is also true that $0 + x = x$ and $x + 0 = x$, that $(x+y)+z = x+(y+z)$, and that $x + y = y + x$. Thus $(\mathbb{R}, \leq, 0, +)$ satisfies the conditions of being a symmetric monoidal preorder.

Exercise 2.4. Consider again the preorder $(\mathbb{R}, \leq)$ from Example 2.3. Someone proposes 1 as a monoidal unit and $*$ (usual multiplication) as a monoidal product. But an expert walks by and says "that won't work." Figure out why, or prove the expert wrong! ◇

Example 2.5. A *monoid* consists of a set M, a function $* : M \times M \to M$ called the *monoid multiplication*, and an element $e \in M$ called the *monoid unit*, such that, when you write $*(m, n)$ as $m * n$, i.e. using infix notation, the equations

$$m * e = m, \qquad e * m = m, \qquad (m * n) * p = m * (n * p) \tag{2.2}$$

hold for all $m, n, p \in M$. It is called *commutative* if also $m * n = n * m$.

Every set S determines a discrete preorder $\mathbf{Disc}_S$ (where $m \leq n$ iff $m = n$; see Example 1.27), and it is easy to check that if $(M, e, *)$ is a commutative monoid then $(\mathbf{Disc}_M, =, e, *)$ is a symmetric monoidal preorder.

Exercise 2.6. We said it was easy to check that if $(M, *, e)$ is a commutative monoid then $(\mathbf{Disc}_M, =, *, e)$ is a symmetric monoidal preorder. Were we telling the truth? ◇

Example 2.7. Here is a non-example for people who know the game "standard poker." Let H be the set of all poker hands, where a hand means a choice of five cards from the standard 52-card deck. As an order, put $h \leq h'$ if h' beats or equals h in poker.

One could propose a monoidal product $\otimes \colon H \times H \to H$ by assigning $h_1 \otimes h_2$ to be "the best hand one can form out of the ten cards in h_1 and h_2." If some cards are in both h_1 and h_2, just throw the duplicates away. So for example $\{2\heartsuit,\ 3\heartsuit,\ 4\heartsuit,\ 6\spadesuit,\ 7\spadesuit\} \otimes \{2\heartsuit,\ 5\heartsuit,\ 6\heartsuit,\ 6\spadesuit,\ 7\spadesuit\} = \{2\heartsuit,\ 3\heartsuit,\ 4\heartsuit,\ 5\heartsuit,\ 6\heartsuit\}$, because the latter is the best hand you can make with the former two.

This proposal for a monoidal structure will fail the condition (a) of Definition 2.1: it could be the case that $h_1 \leq i_1$ and $h_2 \leq i_2$, and yet *not* be the case that $h_1 \otimes h_2 \leq i_1 \otimes i_2$. For example, consider this case:

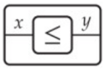
$$\begin{aligned} h_1 &:= \{2\heartsuit,\ 3\heartsuit,\ 10\spadesuit,\ \mathrm{J}\spadesuit,\ \mathrm{Q}\spadesuit\}, & i_1 &:= \{4\clubsuit,\ 4\spadesuit,\ 6\heartsuit,\ 6\diamondsuit,\ 10\diamondsuit\}, \\ h_2 &:= \{2\diamondsuit,\ 3\diamondsuit,\ 4\diamondsuit,\ \mathrm{K}\spadesuit,\ \mathrm{A}\spadesuit\}, & i_2 &:= \{5\spadesuit,\ 5\heartsuit,\ 7\heartsuit,\ J\diamondsuit,\ Q\diamondsuit\}. \end{aligned}$$

Here, $h_1 \leq i_1$ and $h_2 \leq i_2$, but $h_1 \otimes h_2 = \{10\spadesuit,\ \mathrm{J}\spadesuit,\ \mathrm{Q}\spadesuit,\ \mathrm{K}\spadesuit,\ \mathrm{A}\spadesuit\}$ is the best possible hand and beats $i_1 \otimes i_2 = \{5\spadesuit,\ 5\heartsuit,\ 6\heartsuit,\ 6\diamondsuit,\ Q\diamondsuit\}$.

Subsections 2.2.3 and 2.2.4 are dedicated to examples of symmetric monoidal preorders. Some are aligned with the notion of resource theories, others come from pure math. When discussing the former, we will use wiring diagrams, so here is a quick primer.

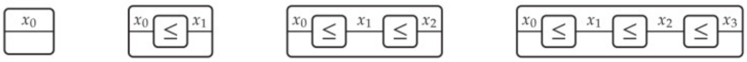
2.2.2 Introducing Wiring Diagrams

Wiring diagrams are visual representations for building new relationships from old. In a preorder without a monoidal structure, the only sort of relationship between objects is $\leq$, and the only way you build a new $\leq$ relationship from old ones is by chaining them together. We denote the relationship $x \leq y$ by



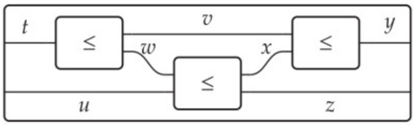
(2.3)

We can chain some number of these $\leq$-relationships – say 0, 1, 2, or 3 of them – together in series as shown here

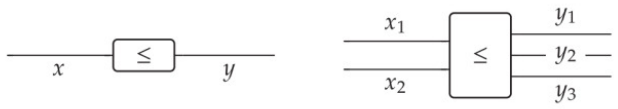


$\cdots$ (2.4)

If we add a symmetric monoidal structure, we can combine relationships not only in series but also in parallel. Here is an example:

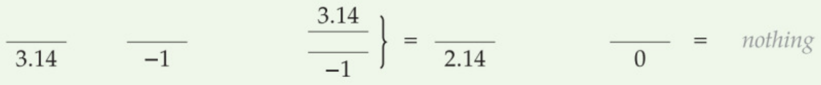


(2.5)

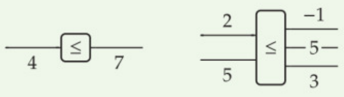


The validity of the second box corresponds to the inequality $x_1 \otimes x_2 \leq y_1 \otimes y_2 \otimes y_3$. Before going on to the properties from Definition 2.1, let us pause for an example of what we've discussed so far.

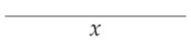
Example 2.8. Recall the symmetric monoidal preorder $(\mathbb{R}, \leq, 0, +)$ from Example 2.3. The wiring diagrams for it allow wires labeled by real numbers. Drawing wires in parallel corresponds to adding their labels, and the wire labeled 0 is equivalent to no wires at all.



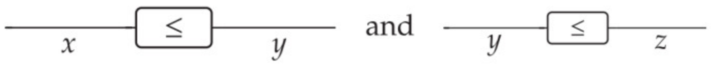
And here we express a couple of facts about $(\mathbb{R}, \leq, 0, +)$ in this language: $4 \leq 7$ and $2 + 5 \leq -1 + 5 + 3$.



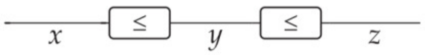
We now return to how the properties of symmetric monoidal preorders correspond to properties of this sort of wiring diagram. Let's first talk about the order structure: conditions (a) – reflexivity – and (b) – transitivity – from Definition 1.25. Reflexivity says that $x \leq x$, this means the diagram just consisting of a wire



is always valid. Transitivity allows us to connect facts together: it says that if $x \leq y$ and $y \leq z$, then $x \leq z$. This means that if the diagrams



are valid, we can put them together and obtain the valid diagram



Next let's talk about the properties (a)–(d) from the definition of symmetric monoidal structure (Definition 2.1). Property (a) says that if $x_1 \leq y_1$ and $x_2 \leq y_2$ then $x_1 \otimes x_2 \leq y_1 \otimes y_2$. This corresponds to the idea that stacking any two valid boxes in parallel is still valid:

For example, the matrix associated to Y in Eq. (2.18) would be

$$M_Y := \begin{array}{c|ccc} \nearrow & x & y & z \\ \hline x & 0 & 4 & 3 \\ y & 3 & 0 & \infty \\ z & \infty & 4 & 0 \end{array} \tag{2.20}$$

As soon as you see how we did this, you'll understand that it takes no thinking to turn a weighted graph G into a matrix M_G in this way. We shall see in Section 2.5.3 that the more difficult "distance matrices" d_Y, such as (2.19), can be obtained from the easy graph matrices M_Y, such as (2.20), by repeating a certain sort of "matrix multiplication."

Exercise 2.40. Copy and complete the matrix M_X associated to the graph X in Eq. (2.18):

$$M_X = \begin{array}{c|cccc} \nearrow & A & B & C & D \\ \hline A & 0 & ? & ? & ? \\ B & 2 & 0 & \infty & ? \\ C & ? & ? & ? & ? \\ D & ? & ? & ? & ? \end{array}$$

◇

2.3.4 $\mathcal{V}$-Variations on Preorders and Metric Spaces

We have told the story of **Bool** and **Cost**. But in Section 2.2.4 we gave examples of many other monoidal preorders, and each one serves as the base of enrichment for a kind of enriched category. Which of them are useful? Something only becomes useful when someone finds a use for it. We will find uses for some and not others, though we encourage readers to think about what it would mean to enrich in the various monoidal categories discussed above; maybe they can find a use we have not explored.

Exercise 2.41. Recall the monoidal preorder **NMY** $:= (P, \leq, \texttt{yes}, \min)$ from Exercise 2.18. Interpret what a **NMY**-category is. ◇

In the next two exercises, we use $\mathcal{V}$-weighted graphs to construct $\mathcal{V}$-categories. This is possible because we will use preorders that, like **Bool** and **Cost**, have joins.

Exercise 2.42. Let M be a set and let $\mathcal{M} := (\mathsf{P}(M), \subseteq, M, \cap)$ be the monoidal preorder whose elements are subsets of M.

Someone gives the following interpretation, "for any set M, imagine it as the set of modes of transportation (e.g. car, boat, foot). Then an $\mathcal{M}$-category $\mathcal{X}$ tells you all the modes that will get you from a all the way to b, for any two points $a, b \in \mathrm{Ob}(\mathcal{X})$."

allowed to equate two paths p and q when they are *parallel*, meaning they have the same source vertex and the same target vertex.

A finite graph with path equations is called a *finite presentation* for a category, and the category that results is known as a *finitely presented category*. Here are two examples:

$$\text{Free\_square} := \boxed{\begin{tikzcd} \overset{A}{\bullet} \arrow[r, "f"] \arrow[d, "g"'] & \overset{B}{\bullet} \arrow[d, "h"] \\ \underset{C}{\bullet} \arrow[r, "i"'] & \underset{D}{\bullet} \\ \end{tikzcd} \quad \textit{no equations}} \qquad \text{Comm\_square} := \boxed{\begin{tikzcd} \overset{A}{\bullet} \arrow[r, "f"] \arrow[d, "g"'] & \overset{B}{\bullet} \arrow[d, "h"] \\ \underset{C}{\bullet} \arrow[r, "i"'] & \underset{D}{\bullet} \\ \end{tikzcd} \quad f \mathbin{;} h = g \mathbin{;} i}$$

Both of these are presentations of categories: in the left-hand one, there are no equations so it presents a free category, as discussed in Section 3.2.1. The free square category has ten morphisms, because every path is a unique morphism.

Exercise 3.9.

1. Write down the ten paths in the free square category above.
2. Name two different paths that are parallel.
3. Name two different paths that are not parallel. ◇

On the other hand, the category presented on the right has only nine morphisms, because $f \mathbin{;} h$ and $g \mathbin{;} i$ are made equal. This category is called the "commutative square." Its morphisms are

$$\{A, B, C, D, f, g, h, i, f \mathbin{;} h\}.$$

One might say "the missing one is $g \mathbin{;} i$," but that is not quite right: $g \mathbin{;} i$ is there too, because it is equal to $f \mathbin{;} h$. As usual, A denotes id_A, etc.

Exercise 3.10. Write down all the morphisms in the category presented by the following diagram:

$$\boxed{\begin{tikzcd} \overset{A}{\bullet} \arrow[r, "f"] \arrow[d, "g"'] \arrow[dr, "j"] & \overset{B}{\bullet} \arrow[d, "h"] \\ \underset{C}{\bullet} \arrow[r, "i"'] & \underset{D}{\bullet} \\ \end{tikzcd} \quad f \mathbin{;} h = j = g \mathbin{;} i}$$

◇