Monoidal Category Theory

Monoidal Category Theory

Unifying Concepts in Mathematics, Physics, and Computing

Noson S. Yanofsky

The MIT Press
Cambridge, Massachusetts
London, England

The MIT Press would like to thank the anonymous peer reviewers who provided comments on drafts of this book. The generous work of academic experts is essential for establishing the authority and quality of our publications. We acknowledge with gratitude the contributions of these otherwise uncredited readers.

This book was set in —— by —–. Printed and bound in the United States of America.

Library of Congress Cataloging-in-Publication Data is available.

ISBN:

10 9 8 7 6 5 4 3 2 1

Dedicated to the memory of my father
Rabbi Moshe Yanofsky
(1942 - 2022)
and to my mother
Sharon Yanofsky

C'est l'harmonie des diverses parties, leur symétrie, l'eur heureux balancement; c'est en un mot tout ce qui y met de l'ordre, tout ce qui leur donne de l'unité, ce qui nous permet par conséquent d'y voir clair et d'en comprendre l'ensemble en même temps que les détails.

It is the harmony of the diverse parts, their symmetry, their happy balance; in a word it is all that introduces order, all that gives unity, that permits us to see clearly and to comprehend at once both the ensemble and the details.
Henri Poincaré [211], page 25, and [212], page 30.

Contents

Preface

... if we expand our experience into wilder and wilder regions of experience - every once in a while, we have these integrations when everything's pulled together into a unification, in which it turns out to be simpler than it looked before.
Richard P. Feynman [73], page 15

Over the past few decades category theory has been used in many different areas of mathematics, physics, and computers. The applications of category theory have arisen in (to name just a few) quantum field theory, database theory, abstract algebra, formal language theory, quantum algebra, theoretical biology, knot theory, universal algebra, string theory, quantum computing, self-referential paradoxes, etc. This book will introduce the category theory necessary to understand large parts of these different areas.

Category theory studies categories, which are collections of structures and ways of changing those structures. Categories have been used to describe many different phenomena in mathematics and science. Our central focus will be monoidal categories, which are souped-up categories that allow one to describe even more phenomena. The theory of monoidal categories has emerged as a theory of structures and processes.

Category theory is a simple, extremely clear and concise language in which various fields of science can be discussed. It is also a unifying language. Different fields are expressed in this single language so that one can see common themes and properties. Category theory also brings together different fields by actually establishing connections between them. It is particularly suited to show the "big picture." Once this language is understood, one is capable of easily learning an immense amount of science, mathematics, and computing.

This book is an introduction to category theory. It begins with the basic definitions of category theory and takes the reader all the way up to cutting-edge research topics. Rather than going "down the rabbit hole" with a lot of very technical "pure" category theory, our central focus will be examples and applications. In fact, an alternative title of this text could be *Category Theory by Example*. A major goal is to show the ubiquity of category theory and finally put an end to the silly canard that category theory is "general abstract nonsense." Another important goal is to show how a multitude of fields are related through category theory.

Within each chapter, whenever there is a definition or theorem, it is immediately followed by examples and exercises that clarify the categorical idea and makes it come alive. The fun is in the examples, and our examples are from many diverse disciplines.

This text contains twelve self-contained mini-courses on various fields. They are short introductions to major fields such as quantum computing, self-referential paradoxes, quantum algebra, etc. These sections do not introduce new categorical ideas. Rather, they use the category theory already presented to describe an entire field. The point we are making with these mini-courses is that, with the language of category theory in your toolbox, you can master totally new and diverse fields with ease.

This book is different from other books on category theory. In contrast to other books, we do not assume that the reader is already a mathematician, a physicist, or a computer scientist. Rather, this book is for anyone who wants to learn the wonders of category theory. We assume that reader is broad-minded, interested in many areas, and wants to see how diverse areas are related to each other. The reader will not only learn all about category theory, they will also learn an immense amount of science, mathematics, and computers.

Another major difference between this book and other introductory category theory books is the way the book is organized. While in most books, the concepts of category, functor, and natural transformation are introduced in the first few pages, here we slowly present each idea separately and in its correct time. With such a presentation, the novice will not be overwhelmed.

Organization

Chapter 1 is an introduction that places category theory in its historical and philosophical context. The introduction ends with a discussion of constructions on sets which will be useful for the rest of the text. Chapters 2, 3, and 4 are a simple introduction to category theory. Chapter 2 contains the basic definitions and properties of categories. Chapter 3 deals with special structures within a category. The real magic begins in Chapter 4 where we see how different categories relate to each other.

Chapters 5, 6, and 7 discuss monoidal categories. Chapter 5 describes monoidal categories, which are categories with extra structure. Chapter 6 deals with the relationships between monoidal categories. The core of the book is Chapter 7, where several variations of monoidal categories are presented with many of their properties and applications.

The final three chapters contain some advanced topics. Many categorical ways of describing structures are explored in Chapter 8. Chapter 9 has a sampling of research areas in advanced category theory. We conclude with Chapter 10 which is a collection of more mini-courses from many different areas.

At the end of each chapter, there is a self-contained mini-course on a single topic. Every chapter and mini-course ends with several pointers to where you can learn more about the particular topic.

Appendix A contains Venn diagrams that describe the relationships of various structures used throughout the text.

Appendix B is an index of the categories that appear in the text.

Appendix C has some suggestions for further study of category theory and its applications.

Appendix D has answers to selected exercises.

Ancillaries

This text does not stand alone. I maintain a web page for the text at

`www.sci.brooklyn.cuny.edu/~noson/MCtext.html.`

Please send any and all comments and suggestions to

noson@sci.brooklyn.cuny.edu.

Acknowledgment

There are many people who have been instrumental in the production of this book.

Brooklyn College has been a large part of my life since I entered as a freshman in September of 1985. I am grateful for the intellectual environment that it provides. My colleagues and friends are always there for a chat, to help develop an idea, and editing a draft. In particular, I would like to thank David Arnow, Eva Cogan, James Cox, Scott Dexter, Lawrence Goetz, Jackie Jones, Keith Harrow, Yedidyah Langsam, Michael Mandel, Simon Parsons, Ira Rudowsky, Charles Schnabolk, Bridget Sheridan, Alexander Sverdlov, Joseph Thurm, Gerald Weiss, and Paula Whitlock.

A large part of this book was written during a sabbatical. I am thankful to President Michelle J. Anderson, Dean Louise Hainline, Dean Peter Tolias, and Chairman Yedidyah Langsam for making that sabbatical possible.

This book owes much to Professor Rohit Parikh who, besides teaching me many ideas, showed me that "You can teach anything to anyone... You just have to teach them the prerequisites first." How true! I am also in debt to Jamie Lennox for repeatedly reminding me that he is my target audience. I hope he is satisfied. Ralph Wojtowicz suggested the mini-course about sets and functions, which is critical for the novice.

My education in category theory owes much to many people. At the end of my first year in graduate school at The Graduate Center of the City University of New York, Alex Heller posted a sign stating that he will be teaching a class called "Categorical Logic" the following semester. He wrote that the prerequisites were the first four chapters of Saunders Mac Lane's *Categories for the Working Mathematician* [180]. Until then, I had never heard of category theory. Over that summer, Mirco Mannucci and I struggled through those chapters. Gradually, the ideas started taking shape. Eventually Heller became my thesis adviser. Between Heller's clarity and Mirco's infectious enthusiasm, I was hooked on category theory. I am forever in their debt.

After receiving my degree, I had the good fortune of doing a post-doc in McGill University. The main professors who organized the category theory group were Michael Barr, Marta Bunge, Jim Lambek, and Michael Makkai. I learned much from them.

Many people were part of the category theory group and regularly joined the weekly seminar. I gained much from Rick Blute, Thomas Fox, Jonathan Funk, André Lebel, Jean-Pierre Marquis, Prakash Panangaden, Robert Paré, Phill Scott, Robert Seely, and others. Many became life-long friends.

On returning to New York, I continued to meet Alex Heller at the Graduate Center. In addition to being a brilliant polymath who could easily discuss myriad topics, he was an extremely kind, gentle man who always had a warm and encouraging word. His vision of mathematics was astonishing and unique. While chatting with him, one was able to get a fleeting glimpse of the clarity, interconnectedness, and broadness of that vision. From my graduation in 1996 until his passing in 2008, we met and chatted once or twice a week. In essence he gave me twelve years of a postdoctoral research fellowship. I am forever indebted to him for all his teaching and his sincere friendship.

My category theory education continues till today. Since 2009, I have been organizing the New York City Category Theory Seminar out of The Graduate Center of the City University of New York. We have a great group of regular participants including Gershom Bazerman and Raymond Puzio. World class category theorists from around the globe come and give talks at the seminar. They explain their research and answer our questions. I have gained much from the participants and the speakers.

I am grateful to the many people who helped with editing the book: Rio Alvarado, Edward Arroyo, Michael Barr, Dominic Barraclough, Tai-Danae Bradley, Miriam Briskman, Leo Caves, Ellen Cooper, Thierry Coquand, Eugene Dorokhin, Hindy Drillick, Vasili I. Galchin, Michael Goldenberg, Jonathan Hanon, Shai Hershfeld, Mark Hillery, Rick Jardine, Samantha Jarvis, Yigal Kamel, Deric Kwok, Moshe Lach, Steve Lack, Klaas Landsman, Jamie Lennox, Armando Matos, William Mayer, Ieke Moerdijk, Micah Miller, Elina Nourmand, Jason Parker, Arthur Parzygnat, Brian Porter, Saeed Salehi, Lorenzo Sauras, Nathan Schor, Dan Shiebler, Michael Shulman, Evan Siegel, David I. Spivak, Ross Street, Joshua Sussan, Karl Svozil, Walter Tholen, Do Anh Tuan, Hadassah Yanofsky, Sharon Yanofsky, Shayna Leah Yanofsky, Joel Zablow, and Marek Zawadowski. Thank you all!

Miriam Briskman did a magnificent job carefully editing the entire book. She also used her exceptional LaTeX skills to make most of the more complicated Ti*k*Z diagrams in the text. Thank you, Miriam! Adina Scheinfeld edited every chapter of this book. She made many great suggestions. Thank you, Adina!

I would also like to thank Elizabeth Swayze, Janet E Rossi, Matthew Valades, and everyone else at MIT Press for helping me get this book into shape.

I am thankful to my children Hadassah, Rivka, Boruch, and Miriam for giving me much joy and the strength to complete this text. Essentially, this book was written because my wife, Shayna Leah, took care of everything else in my life. It is her love and encouragement that made this book possible. I am forever indebted to her.

This book is dedicated to the memory of my father, Rabbi Moshe Yanofsky, and to my mother, Sharon Yanofsky. My father was a professor of mathematics at The City University of New York for many years and concurrently a prominent figure in Jewish education. At the age of twenty-five he became the principal of a Jewish girls high school with over a thousand students. Over the next fifty years, he continued educating and eventually established his own high school. His wisdom and warmth had a profound influence on everyone who met him. He was a wonderful caring father who was always there for us. My father led by example and showed us what it means to live a meaningful life and to have a strong moral code. He was predeceased by a son and is survived by my mother, four children, thirty-eight grandchildren, tens of great-grandchildren, and thousands of loving students. His warm smile is painfully missed.

My mother is a woman of many talents. She was a teacher, a principal, a bank vice president, and is currently a school administrator. Her main hobby is being best friends with every one of her grandchildren and great-grandchildren. She is excellent at everything she does, and does it all with class and a certain panache. My mother raised us with the understanding that anything is possible when you put enough perseverance into it. Everything that I am and all that I do is because of all the effort and love she put into me.

Chapter 1

Introduction

So the problem is not so much to see what nobody has yet seen, as to think what nobody has yet thought concerning that which everybody sees.
Arthur Schopenhauer

We gently introduce the world of category theory. In Section 1.1, a little of the historical and philosophical context is provided. It shows how unification has always been a motivating factor of category theory. Section 1.2 explains the motivation for monoidal categories. We then lay out the structure of the text in Section 1.3. Some standard notation is also presented there. Section 1.4 is a mini-course that uses constructs on sets and functions to teach many categorical ideas.

1.1 Categories

Category theory began with the intention of relating and unifying two different areas of study. The aim was to characterize and classify certain types of geometric objects by assigning to each of them certain types of algebraic objects as depicted in Figure 1.1. (In detail, the geometric objects are structures called topological spaces, manifolds, bundles, etc. The algebraic objects are called groups, rings, abelian groups, etc. The assignments have exotic names like homology, cohomology, homotopy and K-theory, etc.) Researchers realized that if they were going to relate geometric objects with algebraic objects they needed a language that is neither specialized to a geometric content nor an algebraic content. Only with such a general language can one discuss both fields.

Category theory was invented by Samuel Eilenberg and Saunders Mac Lane [67]. They described various collections of mathematical objects. Each collection was called a **category**. There was a collection of geometric objects and a collection of algebraic objects. Eilenberg and Mac Lane were interested in many different categories and in order to relate one category with another, they formulated the notion of a **functor** which — like a function — assigns to each entity in one category an entity in another category. They went further and formulated the notion of a **natural transformation** which is a way of relating one functor to another functor. (In a sense, a natural transformation

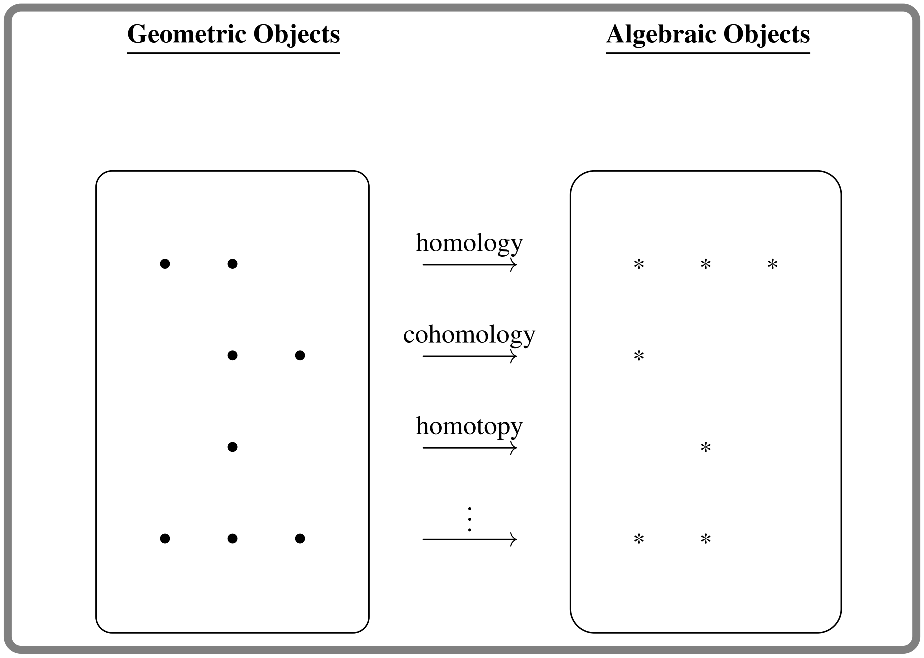


Figure 1.1: Relating geometric objects to algebraic objects.

transfers the results of one functor to the results of another functor.) These structures can be visualized as

$$\begin{tikzcd}
\text{category } \mathbb{A} \arrow[rr, bend left, "\text{functor } F"] \arrow[rr, bend right, "\text{functor } G"'] \arrow[rr, phantom, "\text{natural transformation } \alpha \Downarrow"] & & \text{category } \mathbb{B}.
\end{tikzcd} \tag{1.1}$$

There is category $\mathbb{A}$ and category $\mathbb{B}$. These categories are related by functor F and by functor G. And, finally, these functors are related by natural transformation α.

What is a category? It is a collection of structures called **objects** of a particular type, and a collection of transformations or processes between the objects. We call these transformations or processes **morphisms** or **maps**. If a and b are objects and f is a morphism from a to b we write it as $f\colon a \longrightarrow b$ or $a \xrightarrow{f} b$. We can visualize part of a category as Figure 1.2. The morphisms are to be thought of as ways of transforming objects. As time went on, the morphisms between objects took central stage. Category theory became not only the study of structures but also the study of transformations or processes between structures. One of the main properties of processes is that they can be combined. That is, one process followed by another process can be combined into

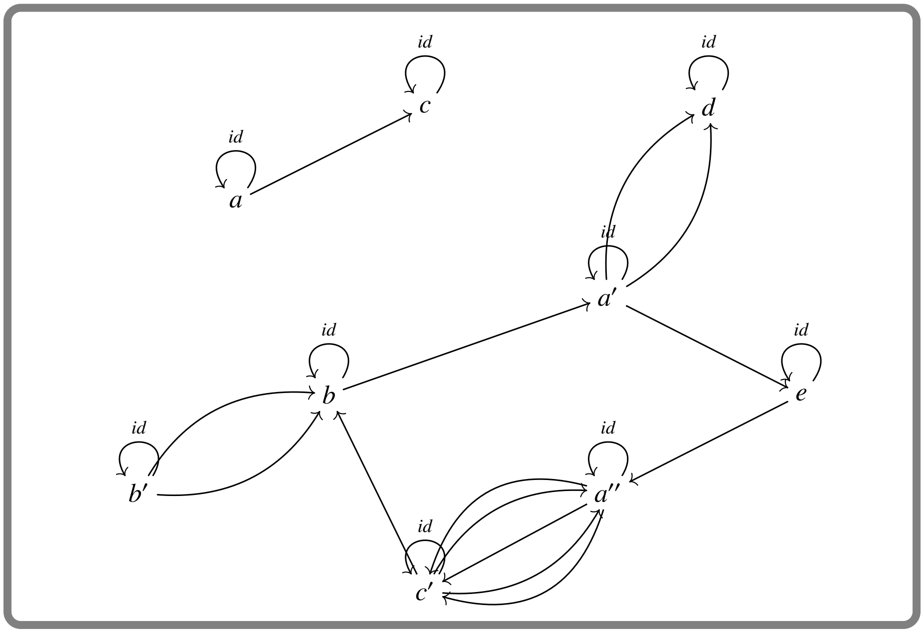


Figure 1.2: An example of part of a category. Every object has an identity morphism (*id*).

a single process. In a category, if there is a morphism from object a to object b called f and a morphism from object b to object c called g, then there exists an associated morphism from object a to object c written as $g \circ f$ and called "g composed f," or "g following f," or "g after f." This can be drawn as follows:

$$\begin{tikzcd} a \arrow[r, "f"] \arrow[rr, bend left, "g\circ f"] & b \arrow[r, "g"] & c. \end{tikzcd} \tag{1.2}$$

This composition is the most fundamental part of a category.

Categories are related to more familiar structures called directed graphs. A **directed graph** is a structure that has objects (also called "vertices," "nodes," or "points") and morphisms (also called "arrows," or "directed edges") between them. One can view a category as a souped-up directed graph. Categories, like directed graphs, also have objects and morphisms, but within categories, one morphism after another can be composed. A directed graph is used to deal with various phenomena of interconnectivity. A category, with its extra structure, deals with more sophisticated notions of interconnectivity (such as reachability). A category can also be seen as a generalization of a group. A **group** is a set where one can combine elements to form other elements. In a category, one can combine morphisms that follow one another. Graphs and groups are ubiquitous in modern science and mathematics. Categories — as generalizations of both structures — are even more pervasive.

Since categories are disassociated from any specific field or area, category theory received the reputation of being a language without content. Because of this, the field was derided by some as "general abstract nonsense." However, it is precisely this independence from any field which gives category theory its power. By not being formulated for one particular field, it is capable of dealing with *any* field. At first, category theory was extremely successful in dealing with various fields of mathematics. As time went on, researchers realized that many branches of science that deal with structures or processes can be discussed in the language of category theory. Computer science is the study of computational processes, and hence, has taken a deep interest in category theory. More recently, category theory has been shown to be very adept at discussing structures and processes in physics. Researchers have also shown that category theory is great at discussing the structures and processes of chemistry, biology, artificial intelligence, and linguistics.

Many diverse fields are shown to be related because they are discussed in the single language of category theory. Researchers have found similar theorems and patterns in areas that were thought to be unrelated. Moreover, in the past few decades, category theory has further unified different fields by revealing amazing relationships between them. There are functors from a category in one field to a category in a totally different field that preserve properties and structures. Such property-preserving functors show that the two fields are similar. For example, quantum algebra is a field that uses categorical language to show how certain algebraic structures are related to geometric structures like knot theory. Another prominent example is topological quantum field theory, which is a branch of math and physics that uses functors to unite relativity and quantum theory. Quantum computing is a field that sits at the intersection of computer science, physics, and mathematics and can easily be understood using various categorical structures.

1.2 Monoidal Categories

In the early 1960s, Jean Bénabou and Saunders Mac Lane described categories that have more structure called **monoidal categories** or **tensor categories**. In these categories, one can "multiply" or "combine" objects. Symbolically, within a monoidal category, object a and object b can be combined to form object $a \otimes b$ (read "a tensor b"). As always in category theory, one is interested not only in combining objects but also in combining morphisms. With morphisms $f\colon a \longrightarrow a'$ and $g\colon b \longrightarrow b'$, there exists objects $a \otimes b$, $a' \otimes b'$, and there also is a morphism $f \otimes g$ which we write as

$$\begin{tikzcd} a \arrow[r, "f"] & a' & & & \\ \otimes & & \text{or} & a \otimes b \arrow[r, "f\otimes g"] & a' \otimes b'. \\ b \arrow[r, "g"'] & b' & & & \end{tikzcd} \tag{1.3}$$

Notice that there are two ways of combining morphisms in a monoidal category. There is $f \circ g$ and there is $f \otimes g$. In physics, the combination $f \circ g$ corresponds to performing one process after another while the combination $f \otimes g$ corresponds to performing two independent processes. In computers, the combination $f \circ g$ corresponds to sequential processes, while $f \otimes g$ corresponds to parallel processes. In mathematics, the interplay of the two combinations of morphisms is very important.

Classical algebra is the branch of mathematics that deals with sets and operations on those sets. For sets of numbers and the addition operation, we have the rule that $x+y = y+x$ while in general, for subtraction, $x-y \neq y-x$. In the theory of monoidal categories there are rules that govern the relationship between $a \otimes b$ and $b \otimes a$. What about the relationship between $(a \otimes b) \otimes c$ and $a \otimes (b \otimes c)$? Within monoidal categories there are many possible relationships when dealing with these combined objects. For each rule relating these operations, there will be a corresponding type of monoidal category. In Chapter 7, we will see many different types of monoidal categories. This variability allows for many phenomena to be modeled by monoidal categories. The area that deals with the different types of rules among operations is called **coherence theory** (i.e., how the various operations *cohere* with each other) or **higher-dimensional algebra**. This area of study has become pervasive, and it is believed that higher-dimensional algebra will arise even more frequently in the science and mathematics of the coming decades.

1.3 The Examples and the Mini-courses

This text is centered on the examples. Our goal is to show the pervasiveness of categories, and in particular, monoidal categories. We also want to emphasize how categories can reveal the interconnectedness of various fields. We do so by introducing many examples from many different areas. Immediately following a definition or a theorem of category theory there are lots of examples that illustrate the idea. There are also some examples that are left to the reader as exercises. It is important to realize that although this book is chock-full of examples, we have barely scratched the surface. The literature of category theory has many more examples. We chose the examples that arise most frequently or are the easiest to understand. The reader will be directed to places in the literature where other examples are described. We are showing the beauty of category theory but only revealing the tip of the iceberg.

Most of the examples can be loosely split into three broad groupings: mathematics, physics, and computers. The problem is that the boundaries between these different areas are hazy. For example, is quantum computing a part of computer science, physics or abstract mathematics? Is knot theory a part of mathematics or physics? There are no firm boundaries.

Since most readers are familiar with sets and functions between sets, we usually try to first show an idea or definition in terms of sets. In later chapters, it will become apparent that sets and functions between sets are not the right context to examine certain phenomena. This is where category theory really gets interesting.

The examples are spread throughout the book. To illustrate, in Chapter 2, a category will be introduced. In Chapter 3, some properties of this category will be described. This category will be related to other categories in Chapter 4. In Chapter 5 we will show that the category has a monoidal structure, and we will see how that monoidal structure relates to the monoidal structure of other categories in Chapter 6. This same category and variations of this category will be shown to have even more structure in Chapter 7. We will also see how this category arises in various mini-courses. By the time the reader finishes the book, the category will be an old friend.

Not all categories are introduced early on. In order not to overwhelm the reader in the beginning, we will introduce many categories in later chapters as well. Our aim is readability and understanding.

These examples will take the reader rather far. In mathematics, the reader will meet lots of algebra and topology. In physics we will see the basics of quantum theory. In computers we will see how categories are good for describing certain models of computation and some advanced logic.

Due to space limitations and by concentrating on examples, we are going to omit some results in pure category theory. We only describe the category theory required to understand the examples. In Appendix C, we point out various places where one can learn more (pure) category theory.

Category theory is a language that can deal with many different areas of science. The really fun part of category theory is that once one has this language in their toolbox, one can easily pick up whole new branches of science. We show this flexibility with little mini-courses. At the end of every chapter is a little self-contained section that describes a whole field with the category theory already learned. Mini-courses in later chapters depend on the knowledge of earlier mini-courses. In Chapter 10, we offer several other mini-courses.

One of the intended mini-courses for this book was on the topic of theoretical computer science. Within that mini-course there were sections on models of computation, computability theory, complexity theory, and several other topics. Some of the deepest ideas and theorems of modern computers and mathematics, e.g., Turing machines, unsolvable problems, the P=NP question, Kurt Gödel's incompleteness theorem, intractable problems, Turing's Halting problem, etc.,were met and explained using simple categorical language. While writing about these topics, I was surprised at the ease in which complicated ideas of theoretical computer science can be expressed and proved using category theory. That mini-course took on a life of its own and grew out of this book and it became its own book, *Theoretical Computer Science for the Working Category Theorist* [274]. Consider that book a companion volume of this text for readers interested in those fascinating topics.

This book owes a tremendous debt to previous works.

- I "cut my teeth" learning category theory from Saunders Mac Lane's *Categories for the Working Mathematician* [180]. This is *the* classic text by one of the founders of category theory. It influenced my thinking and this book in the most profound way. As the title implies, Mac Lane assumed that the reader knows large parts of mathematics before opening his book. My goal with this book is to give to a larger audience the beauty of category theory as Mac Lane did.
- John Baez and Michael Stay wrote a wonderful paper "Physics, Topology, Logic and Computation: A Rosetta Stone" [24] that highlights connections between many different fields. I would like to think of this book as an explanation and an

expansion of that paper.

- I learned much from Christian Kassel's textbook *Quantum Groups* [134]. His clarity and exactness is an inspiration.
- This book attempts to be as readable as Michael Barr and Charles Wells' textbook *Category Theory for Computing Science* [33]. Their work goes through large segments of category theory with many examples along the way. We try to do the same but with examples from physics and mathematics as well.

It must be noted that this is not a history book. We are not going to say who thought of some particular construction or example first. Some of the examples in this book came from other books and papers. Some examples are just known in the folklore of category theory. And we made up some examples. The history is too complicated for us to disentangle and is of absolutely no pedagogical use to the novice. We name some places to learn about the history of category theory in Appendix C.

There are many potential topics that could have gone into this book. Painful choices had to be made. In the end, topics were chosen based on a desire to provide as diverse a set of examples as possible to satisfy a broad readership, with a natural bias to those areas which the author feels more confident to address. I would like to believe that the topics chosen will be important as we march into the unfathomable future.

Finally, I would like to apologize to all my friends in the category theory community if I neglected their favorite example or did not discuss an area in which they did great work. It was not my intention to omit anyone's work.

In order to improve readability, for the most part, we keep to the following notation.

- Categories are in blackboard bold font: $\mathbb{A}, \mathbb{B}, \mathbb{C}, \mathbb{D}, \mathbb{Circuit}, \mathbb{Set}, \ldots$
- Objects in general categories are the first few lowercase Latin letters: $a, b, c, d, a', b', a'' \ldots$
- Morphisms in general categories are lowercase Latin letters: $f, g, h, i, j, k, f', g'', \ldots$
- Functors are capital Latin letters: $F, G, H, I, J, \ldots$
- Natural transformations are lowercase Greek letters: $\alpha, \beta, \gamma, \delta, \eta, \kappa, \ldots$
- Higher-dimensional morphisms will be capital Greek letters: $\Gamma, \Delta, \Theta, \Phi, \Psi, \ldots$
- Sets of numbers: $\mathbf{N}, \mathbf{Z}, \mathbf{Q}, \mathbf{R}, \mathbf{C}$.
- 2-Categories are in blackboard bold font with a line above: $\overline{\mathbb{A}}, \overline{\mathbb{B}}, \overline{\mathbb{C}}, \overline{\mathbb{D}}, \overline{\mathbb{Cat}} \ldots$
- 3-Categories are in blackboard bold font with a two lines above: $\overline{\overline{\mathbb{A}}}, \overline{\overline{\mathbb{B}}}, \overline{\overline{\mathbb{C}}}, \overline{\overline{\mathbb{D}}}, \overline{\overline{2\mathbb{Cat}}} \ldots$

There are several different types of arrows in this book.

- Morphism, map or functor: $\longrightarrow$
- The input and output of a function or a functor: $\longmapsto$ or $\vdash\!\!\rightsquigarrow$
- Inclusion or injection: $\hookrightarrow$
- Surjection or full functor: $\twoheadrightarrow$
- Natural transformation: $\Longrightarrow$

1.4 Mini-course: Sets and Categorical Thinking

Category theory is not just a language that is capable of describing an immense amount of science and mathematics. Rather, it is a *new and innovative way of thinking.* One of the central ideas is that we define properties of objects by the way they interact with other objects.

Important Categorical Idea 1.4.1. Morphisms Are Central. Properties and structures in a category can be described by the morphisms of the category. That is, the objects do not stand alone. One must see how the objects relate to each other with morphisms. The objects have to be seen in context of the morphisms.

In particular, many properties of an object b can be understood by looking at the collections of morphisms $a \longrightarrow b$ for various simple objects a. Similarly, many properties of b are described by looking at collections of morphisms $b \longrightarrow c$ for various simple objects c. ○

In order to get a feel for this, we take an in-depth look at the familiar world of sets and functions between sets. We show that many of the usual ideas and constructions about sets can be described with functions between sets. This mini-course will also be a gentle reminder of many concepts that are needed in the rest of the text. Throughout this book, we will point back to equations, diagrams, and ideas found in this section.

Sets and Operations

Definition 1.4.2. A **set** is a collection of elements. If S is a set and x is an element of S, we write $x \in S$. If x is not an element of S we write $x \notin S$.

Example 1.4.3. We will deal with both infinite sets and finite sets. Some of the most important infinite sets of numbers are

- The natural numbers, $\mathbf{N} = \{0, 1, 2, 3, \ldots\}$.
- The integers, $\mathbf{Z} = \{\ldots, -3, -2, -1, 0, 1, 2, 3, \ldots\}$.
- The rational numbers, $\mathbf{Q} = \{\frac{m}{n} : m \text{ and } n \text{ in } \mathbf{Z} \text{ and } n \neq 0\}$.
- The real numbers, $\mathbf{R}$, that is, all numbers on the real number line.
- The complex numbers, $\mathbf{C} = \{a + bi : a \text{ and } b \text{ in } \mathbf{R}\}$.

□

We begin by discussing several operations on sets. Let S and T be sets. If s is in S and t is in T, we write an **ordered pair** of the elements as (s, t). The set of all ordered pairs is called the **Cartesian product of sets** S and T

$$S \times T = \{(s, t) \colon s \in S, t \in T\}. \tag{1.4}$$

Example 1.4.4. If $Pants$ = {black, blue1, blue2, gray} is the set of pants that you own, and $Shirts$ = {white, blue, orange} is the set of shirts that you own, then the set of *Outfits* is

$$Pants \times Shirts = \left\{ \begin{matrix} \text{(black, white), (black, blue), (black, orange),} \\ \text{(blue1, white), (blue1, blue), (blue1, orange),} \\ \text{(blue2, white), (blue2, blue), (blue2, orange),} \\ \text{(gray, white), (gray, blue), (gray, orange)} \end{matrix} \right\} \tag{1.5}$$

(True scholarly category theorists do not care if their clothes fail to match!) □

Technical Point 1.4.5. The most important aspect of an ordered pair is its order. In contrast, sets are just collections, and as such, do not have a preferred order. The set $\{s, t\}$ is considered to be the same set as $\{t, s\}$. In contrast, the pair (s, t) is not considered the same as (t, s). Hence, we cannot simply use two curly brackets to describe ordered pairs. There are other ways of describing an ordered pair of elements from S and T. For example, we could write them as $\langle s, t\rangle$ or $\{s, t, \{s\}\}$ (where we collect the elements but indicate the first element by putting it into a set by itself), or even $\{s, t, \{t\}\}$ (where we indicate the second element by putting it into a set by itself). There is nothing special about the notation (s, t). (We will see this again in the beginning of Section 3.1) ♡

The most interesting property about a finite set is the number of elements in such a set. For every finite set S, we write $|S|$ to denote the number of elements in S.If there are m elements in S and n elements in T, then there are mn elements in $S \times T$. In symbols we write this as

$$|S \times T| = |S| \cdot |T|. \tag{1.6}$$

Exercise 1.4.6. How many elements are in $Pants$? How many elements are in $Shirts$? How many elements in *Outfits*? Show that the above formula works.

1.4.6 4, 3, $12 = 4 \cdot 3$.

■

We can generalize the notion of ordered pairs to **ordered triples**, **ordered 4-tuples**, **ordered 5-tuples**, etc. If there are n sets, $S_1, S_2, \ldots, S_n$, then an **ordered n-tuple** is written as $(s_1, s_2, \ldots, s_n)$ where s_i is in S_i. The set of all n-tuples is $S_1 \times S_2 \times \cdots \times S_n$. The number of n-tuples follows a generalization of Equation (1.6):

$$|S_1 \times S_2 \times \cdots \times S_n| = |S_1| \cdot |S_2| \cdots |S_n|. \tag{1.7}$$

Exercise 1.4.7. In addition to pants and shirts, an outfit might consist of a hat, socks, and shoes. How many outfits are there if there are m hats, n pairs of socks, p pairs of shoes, q pants and r shirts?

1.4.7 $m \cdot n \cdot p \cdot q \cdot r$.

■

Another operation performed on sets is the union. Let S and T be sets. The **union** of S and T is the set $S \bigcup T$ which contains those elements that are in S or in T.

$$S \bigcup T = \{x \colon x \in S \text{ or } x \in T\}. \tag{1.8}$$

It is important to notice that if there is some element that is in both S and in T then it will occur only once in $S \bigcup T$. This is because when dealing with a set, repetition does not matter. The set $\{a, b, c, b\}$ is considered the same set as $\{a, b, c\}$.

A related operation is the disjoint union. Given sets S and T, one forms the **disjoint union** $S \amalg T$ which contains the elements from S and T but considers elements that are in both sets as different elements. One way this is done is by tagging every element with extra information that says which set it comes from. This way an element that is both in S and in T would be considered two different elements. For example, if $S = \{a, b, c, x, y\}$ and $T = \{q, w, b, x, e, r\}$, then

$$S \amalg T = \{(a, 0), (b, 0), (c, 0), (x, 0), (y, 0), (q, 1), (w, 1), (b, 1), (x, 1), (e, 1), (r, 1)\}, \tag{1.9}$$

where the elements of S are tagged with a 0 and the elements of T are tagged with a 1. In general for sets S and T, we have

$$S \amalg T = (S \times \{0\}) \bigcup (T \times \{1\}). \tag{1.10}$$

The formula for the number of elements in the disjoint union is $|S \amalg T| = |S| + |T|$.

Exercise 1.4.8. When does the union of two sets have the same number of elements as the disjoint union of those same sets?

1.4.8 When the two sets have nothing in common, i.e., when the intersection of the two sets is empty.

■

Functions

The central idea of this mini-course is the notion of functions between sets and how they determine properties of sets.

Important Categorical Idea 1.4.9. Not Things, But Morphisms Between Things. In category theory, whenever we have a notion (for example, a set), the immediate next task is to consider how these notions relate to each other (for example, functions are ways for sets to relate to each other.) As we have stated, category theory is not about "things," but about how "things" relate to "things." Relations between objects are usually described by morphisms or functions between the objects.

Following this rule, once we describe the morphisms between the objects we must immediately ask what is between the morphisms. Usually the answer will be other morphisms. The computer scientist might protest that this recursive procedure will lead into an infinite loop. It will! We will see this in Section 9.4 when we describe infinite levels of morphisms in higher-dimensional category theory. Such structures are a reflection of this important idea that is at the center of category theory. ○

Definition 1.4.10. Let S and T be sets. A **function** f from S to T, written $f\colon S \longrightarrow T$ is an assignment to every element of S an element of T. The value of f on the element s is written as $f(s)$ ("f of s"). If $f(s) = t$ we write $s \longmapsto t$ ("s maps to t").

It is important to understand the difference between the symbol $\longrightarrow$ and the symbol $\longmapsto$. The symbol $\longrightarrow$ goes between two sets. It describes a function from one set to another. In contrast, the symbol $\longmapsto$ goes from an element in the first set to an element in the second set. It describes how the function is defined.

Example 1.4.11. For every set S, there is an **identity function** $id_S : S \longrightarrow S$ which takes every element to itself. In symbols it is defined as $id_S(s) = s$ or $s \longmapsto s$. □

Following Important Categorical Idea 1.4.1, let us look at how morphisms to a set, determine properties of a set. Functions can be used as a way of describing or choosing elements of a set. Consider a one-element set, $\{*\}$. (There are many one-element sets such as $\{a\}$, $\{b\}$, {Bill}, etc.) For a set S, a function $f\colon \{*\} \longrightarrow S$ picks out one element of S. The single element $*$ goes to the selected element s in S. In symbols, $f(*) = s$ or $* \longmapsto s$.

Example 1.4.12. Let S be the set {Jack, Jill, Joane, June, Joe, John}. The element Joe in S can be described as a function $f\colon \{*\} \longrightarrow S$ where $f(*) = \text{Joe}$. We might want to distinguish this function by calling it $f_{\text{Joe}}\colon \{*\} \longrightarrow S$. There will be other functions like $f_{\text{Jill}}\colon \{*\} \longrightarrow S$ where $f_{\text{Jill}}(*) = \text{Jill}$. For this set of six elements, there are six different functions from $\{*\}$ to S. □

If we are interested in choosing two elements of S, we can look at functions from a two-element set to S. So $f\colon \{0, 1\} \longrightarrow S$ will choose two elements of S. The first element is $f(0)$ and the second element is $f(1)$. If $f(0) \neq f(1)$ then f will choose two *different* elements of S. Every such function chooses two elements of S. Functions from $\{a, b, c\}$ to S will choose three elements of S. This can go on: if we wanted to choose n elements of a set, we would look at functions of the form

$$\{1, 2, \ldots, n\} \longrightarrow S. \tag{1.11}$$

Definition 1.4.13. A set T is a **subset** of S if every element of T is an element of S. We write this as $T \subseteq S$. If T is a subset of S but not equal to S, we call T a **proper subset** and write $T \subsetneq S$ or $T \subset S$. This is the case when there is at least one element in S that is not in T. If T is a subset of S, there is an **inclusion function** that takes every element of T to its corresponding element of S which is written as $inc\colon T \hookrightarrow S$. There is a special set that has no elements called the **empty set** and denoted $\emptyset$. Since it is true that whatever is in $\emptyset$ (nothing) is in any other set, we have that the empty set is a subset of every set. Furthermore, for every set S, there is a unique function from the empty set to S. We sometimes denote this function as $!\colon \emptyset \longrightarrow S$.

Subsets of a set are of fundamental importance. We shall be interested in the collection of all subsets of a particular set.

Definition 1.4.14. For a set S, the set of all subsets of S is called the **powerset** of S and is denoted $\mathcal{P}(S)$. In other words,

$$\mathcal{P}(S) = \{T : T \text{ is a subset of } S\}. \tag{1.12}$$

Example 1.4.15. For the set $\{a\}$, the powerset is $\mathcal{P}(\{a\}) = \{\emptyset, \{a\}\}$. The powerset of a two element set $\{a, b\}$ is $\mathcal{P}(\{a, b\}) = \{\emptyset, \{a\}, \{b\}, \{a, b\}\}$. The powerset of a three-element set $\{a, b, c\}$ is $\mathcal{P}(\{a, b, c\}) = \{\emptyset, \{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\}\}$. Whenever we add an element to a set, we double the number of elements in the powerset. We have following rule: if S has n elements, then $\mathcal{P}(S)$ has 2^n elements. In symbols, $|S| = n$ implies $|\mathcal{P}(S)| = 2^n$. We can also write this as $|\mathcal{P}(S)| = 2^{|S|}$. □

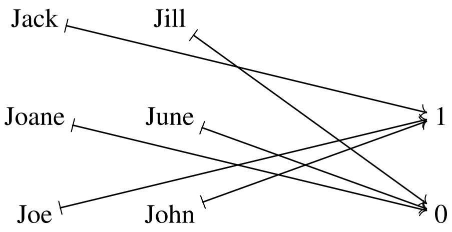
Functions can be used to describe subsets.

Definition 1.4.16. For any set S and subset $T \subseteq S$, there is an associated **characteristic function** $\chi_T\colon S \longrightarrow \{0, 1\}$. This function assigns either 1 or 0 to every element s of S. If s is in T, the characteristic function assigns a 1 to s, and if s is not in T, it assigns a 0 to s, i.e.,

$$\chi_T(s) = \begin{cases} 1 & : s \in T \\ 0 & : s \notin T. \end{cases} \tag{1.13}$$

(The Greek letter χ is pronounced "chi" and is supposed to remind you of the first syllable of "characteristic".) The function χ_T tells which elements of S are in T and which elements of S are not in T. Characteristic functions establish a correspondence between subsets of S and functions from S to $\{0, 1\}$.

Example 1.4.17. Let S be the set {Jack, Jill, Joane, June, Joe, John}. Consider the subset T = {Jack, Joe, John} of S that contains all the boys in S. This subset can be described by the function $\chi_T\colon S \longrightarrow \{0,1\}$ which can be visualized as



(1.14)

□

Exercise 1.4.18. For the sets of numbers, we know that $\mathbf{N} \subsetneq \mathbf{Z} \subsetneq \mathbf{Q} \subsetneq \mathbf{R} \subsetneq \mathbf{C}$. Give the characteristic function for each of these proper subsets.

1.4.18 Let us just focus on the subset $\mathbf{Q} \subsetneq \mathbf{R}$. The characteristic function $\chi_{\mathbf{Q}}\colon \mathbf{R} \longrightarrow \{0,1\}$ is defined as follows:

$$\chi_{\mathbf{Q}}(r) = \begin{cases} 1 & : r \text{ is rational} \\ 0 & : r \text{ is not rational.} \end{cases}$$

The others are done similarly.

■

A characteristic function assigns the elements of S to one of two possible values. There might be a need to assign one of many values to every element of S. For example, a function $S \longrightarrow \{a,b,c,d\}$ assigns to every element of S one of these letters which can stand for different ideas. In general, a function

$$S \longrightarrow \{1,2,\ldots,n\} \tag{1.15}$$

assigns every element of S one of n numbers. We can also assign to every element of S an element of $[0,1]$, the real interval between 0 and 1. Such a function may correspond to assigning a probability to every element.

Example 1.4.19. In school, every student usually has an associated grade point average (GPA). This is written as a function $Students \longrightarrow [0,4]$. □

If S is a set, then there is a function called the **diagonal function** $\Delta\colon S \longrightarrow S \times S$ which takes every element to an ordered pair of the same element. In symbols, for s in S we have

$$\Delta(s) = (s,s). \tag{1.16}$$

If $f\colon S \longrightarrow S'$ and $g\colon T \longrightarrow T'$ are functions then there exists a function $f \times g\colon S \times T \longrightarrow S' \times T'$ that takes an ordered pair of elements and applies f to the first element and g to the second. In symbols, the function is defined for elements s of S and t of T as

$$(f \times g)((s,t)) = (f(s), g(t)) \in S' \times T'. \tag{1.17}$$

In a sense, this process is a parallel process. The function f processes s while the function g processes t.

Exercise 1.4.20. Let $f\colon \mathbf{N} \longrightarrow \mathbf{R}$ be defined by $f(n) = \sqrt{n}$ and $g\colon \mathbf{R} \longrightarrow \mathbf{Z}$ be the ceiling function denoted as $g(r) = \lceil r \rceil$. (The ceiling functions outputs the least integer greater than or equal to the input.) What is $(f \times g)((5, -5.1))$?

1.4.20 $(\sqrt{5}, -5)$.

■

Definition 1.4.21. There are some special types of functions. We say $f\colon S \longrightarrow T$ is

- **one-to-one** or **injective** if different elements in S go to different elements in T. That is, for all s and s' in S, if $s \neq s'$ then $f(s) \neq f(s')$. Another way to say this is that if $f(s) = f(s')$, then it must be that $s = s'$. This means that if the function takes elements to the same output, the elements must have started off equal.
- **onto** or **surjective** if for every element t in T, there is an s in S such that $f(s) = t$.
- an **isomorphism** or a **one-to-one correspondence** or a **bijection** if f is one-to-one and onto. That is, for every element s of S there is a unique element t of T so that $f(s) = t$ and for every element t of T there is a unique element s of S so that $f(s) = t$. When there are sets S and T with an isomorphism between them, we say the sets are **isomorphic** and write this as $S \cong T$.

Isomorphism of sets is not the same idea as equality of sets. Consider a simple example: the set $\{x\}$ and the set $\{y\}$. Although these two sets have exactly the same number of elements, they are not equal. They are only isomorphic.

Exercise 1.4.22. Explain why two finite sets that have the same number of elements are isomorphic. ■

Exercise 1.4.23. Show that the Cartesian plane, $\mathbf{R} \times \mathbf{R}$, is isomorphic to the plane of complex numbers, $\mathbf{C}$.

1.4.23 The isomorphism will take a pair of real numbers $(r_1, r_2) \in \mathbf{R}\times\mathbf{R}$ to the complex number $r_1 + ir_2 \in \mathbf{C}$ where $i = \sqrt{-1}$.

■

One of the central ideas about sets is that given sets S and T we can form a set which consists of all functions from S to T. So while we looked at examples of particular functions, we will also be interested in the collection of *all* functions from one set to another set. This collection will have interesting structure. We call this collection a **set of functions** or a **function set** or a **Hom set** and we denote it as $Hom(S,T)$ or T^S. (The notation $Hom(S,T)$ comes from the word "**hom**omorphism" which is a vestige of the algebraic origins of the idea. The notation T^S is similar to exponentiation because the function set has similar properties to exponentiation.)

Exercise 1.4.24. Write down the set of all the functions from the set $\{a,b,c\}$ to the set $\{0,1\}$.

1.4.24 Each of the following lines is a function.

$f(a) = 0$	$f(b) = 0$	$f(c) = 0$
$f(a) = 0$	$f(b) = 0$	$f(c) = 1$
$f(a) = 0$	$f(b) = 1$	$f(c) = 0$
$f(a) = 0$	$f(b) = 1$	$f(c) = 1$
$f(a) = 1$	$f(b) = 0$	$f(c) = 0$
$f(a) = 1$	$f(b) = 0$	$f(c) = 1$
$f(a) = 1$	$f(b) = 1$	$f(c) = 0$
$f(a) = 1$	$f(b) = 1$	$f(c) = 1$

■

We saw that every element in a set S can be described as a function $\{*\} \longrightarrow S$. This correspondence between elements of S and functions from $\{*\}$ to S shows that

$$S \cong S^{\{*\}} = Hom(\{*\},S). \tag{1.18}$$

Using characteristic functions, we saw in Definition 1.4.16 that there is a correspondence between subsets of S and functions from S to $\{0,1\}$. This correspondence can be stated as

$$\mathcal{P}(S) \cong \{0,1\}^S = Hom(S,\{0,1\}). \tag{1.19}$$

We will denote the set $\{0,1\}$ as 2 and then write this as

$$\mathcal{P}(S) \cong 2^S = Hom(S,2). \tag{1.20}$$

Example 1.4.25. Consider the simple binary addition operation $+\colon \mathbf{N}\times\mathbf{N} \longrightarrow \mathbf{N}$. Let us write this function with its inputs clearly marked as follows

$$(\ ) + (\ )\colon \mathbf{N}\times\mathbf{N} \longrightarrow \mathbf{N}. \tag{1.21}$$

Now consider the function $(\ ) + 5\colon \mathbf{N} \longrightarrow \mathbf{N}$. This is a function with only one input. We could also make another function of one variable $(\ ) + 7\colon \mathbf{N} \longrightarrow \mathbf{N}$. In fact we can do this for any natural number. We can define a function that inputs a natural number and outputs a function from natural numbers to natural numbers. That is, there is a function $\Phi\colon \mathbf{N} \longrightarrow \mathit{Hom}(\mathbf{N},\mathbf{N})$ which is defined as $\Phi(n) = (\ ) + n$. The information described by the $(\ ) + (\ )$ function is the same as the information described by the Φ function.

Notice that what we said about + really applies to every function with two inputs. If $f\colon S \times T \longrightarrow U$ is a function from $S \times T$, then for every $t \in T$ there is a function $f(\ ,t)\colon S \longrightarrow U$. This shows that there is a function $f'\colon T \longrightarrow \mathit{Hom}(S,U)$. It is easy to see that the assignment described by any function f has the same information as the assignment described by the function f'. That is, f and f' relay the same information. □

This example brings to light the following important theorem about sets.

Theorem 1.4.26. For sets S, T and U there is an isomorphism

$$\mathit{Hom}(S \times T, U) \cong \mathit{Hom}(T, \mathit{Hom}(S,U)) \qquad \text{or} \qquad U^{S\times T} \cong (U^S)^T \tag{1.22}$$

★

Proof. To show that these two sets are isomorphic, consider $f\colon S \times T \longrightarrow U$. From this function let us define an $f'\colon T \longrightarrow \mathit{Hom}(S,U)$. For a t in T we have the function $f'(t)\colon S \longrightarrow U$ which is defined as follows: for s in S, let $f'(t)(s) = f(s,t)$. This function has the same information as f. Constructing f from f' is left to the reader. ♣

Let us count how many functions there are between two finite sets. Consider S with $|S| = m$ and T with $|T| = n$, and a function $f\colon S \longrightarrow T$. For each element s in S there are n possible values of $f(s)$ in T. For two elements in S there are $n \cdot n$ possibilities of choices in T. In total, there are $n \cdot n \cdot \cdots \cdot n$ (m times) possible maps. So

$$|\mathit{Hom}(S,T)| = |T^S| = n^m = |T|^{|S|}. \tag{1.23}$$

Remark 1.4.27. For three sets S, T, and U, we have

$$\begin{aligned}
|\mathit{Hom}(S \times T, U)| &= |U^{S\times T}| && \text{by definition or notation}\\
&= |U|^{|S\times T|} && \text{by Equation (1.23)}\\
&= |U|^{|S|\cdot|T|} && \text{by Equation (1.6)}\\
&= (|U|^{|S|})^{|T|} && \text{by arithmetic}\\
&= |\mathit{Hom}(S,U)|^{|T|} && \text{by Equation (1.23)}\\
&= |\mathit{Hom}(T, \mathit{Hom}(S,U))| && \text{by definition.}
\end{aligned}$$

Notice that the rule about exponentiation usually learned as children, $m^{(n\cdot p)} = (m^n)^p$ is expanded to a rule about sets and functions. ♠

Operations on Functions

Often we are going to take two functions and perform an operation to get another function. Three such operations are composition, extension, and lifting[1] Remarkably, many ideas about functions can be understood as operations in one of these three forms.

The simplest operation is **composition**. If there is a function $f\colon S \longrightarrow T$ and a function $g\colon T \longrightarrow U$, then the composite of them is a function $h = g \circ f\colon S \longrightarrow U$ that is defined on an s in S as $h(s) = g(f(s))$. We write these functions as

$$\begin{tikzcd}
S \arrow[rr, "f"] \arrow[dr, "h=g\circ f"'] & & T \arrow[dl, "g"] \\
& U. &
\end{tikzcd} \tag{1.24}$$

We say that f and g are **factors** of h or h **factors through** T. This diagram is called a **commutative diagram**. It means that if you start with any element s in S and you apply the functions f followed by g to go from S to U, you will get to the same resulting element as applying the function h to the element s. In detail, for all s, we have that $g(f(s)) = h(s)$. The diagram is "commutative" because we can go around the triangle this way or that way. There will be many such diagrams in the coming pages. In all the cases, start from any set and follow all the paths of composible functions to another set, and you will get the same element. Throughout this text, unless otherwise stated, all diagrams are commutative.

Example 1.4.28. Consider the set of real numbers, $\mathbf{R}$. Let a and b be real numbers. Consider the function $(\ )\cdot a\colon \mathbf{R} \longrightarrow \mathbf{R}$ that takes any real number and multiplies it by a. There is also a function $(\ )\cdot b\colon \mathbf{R} \longrightarrow \mathbf{R}$ that takes any real number and multiplies it by b. The composition of these two functions is the function $(\ )\cdot(a\cdot b)\colon \mathbf{R} \longrightarrow \mathbf{R}$ that takes any real number and multiplies it by $a \cdot b$ as in the following commutative diagram

$$\begin{tikzcd}
\mathbf{R} \arrow[rr, "(\ )\cdot a"] \arrow[dr, "(\ )\cdot(a\cdot b)"'] & & \mathbf{R} \arrow[dl, "(\ )\cdot b"] \\
& \mathbf{R}. &
\end{tikzcd} \tag{1.25}$$

One can make similar compositions with other arithmetic operations. □

[1] Technically, extension and lifting are not operations. Composition is an operation because if you take two composable functions, you will get a unique composiiton function. In contrast, given two functions of a certain type, their extension and their lifiting are not necessarily unique. There might be many extension and many liftings for two function. But we will use the word "operation" for these cases also.

Exercise 1.4.29. Show that function composition is associative. That is, let $f\colon S \longrightarrow T$, $g\colon T \longrightarrow U$ and $h\colon U \longrightarrow V$, and show that $h \circ (g \circ f) = (h \circ g) \circ f$.

1.4.29 There are two ways of associating the functions: $h \circ (g \circ f)$ and $(h \circ g) \circ f$. These two function are the same. On input s of S, both functions have the value $h(g(f(s)))$.

■

The fact that function composition is associative will be used many times throughout this text.

When dealing with the identity function, the input is the same as the output. This has an interesting consequence when dealing with composition. If you compose a function with an identity map, then you get the original function. In detail, for $f\colon S \longrightarrow T$, $id_S : S \longrightarrow S$, and $id_T : T \longrightarrow T$, we have

$$f \circ id_S = f \qquad \text{and} \qquad id_T \circ f = f. \tag{1.26}$$

We can see these equations as the following commutative diagram:

$$\begin{tikzcd}
S \arrow[rr, "id_S"] \arrow[dr, "f=f\circ id_S"'] & & S \arrow[dl, "f"'] \arrow[dr, "f=id_T\circ f"] & \\
& T \arrow[rr, "id_T"'] & & T.
\end{tikzcd} \tag{1.27}$$

Example 1.4.30. Evaluation of a function can be seen as composition. Let $f\colon S \longrightarrow T$ be a function and let an element be described by the function $g\colon \{*\} \longrightarrow S$. Then the value of f on the element that g chooses is the element that $f \circ g$ chooses, as in

$$\begin{tikzcd}
\{*\} \arrow[rr, "g"] \arrow[dr, "f\circ g"'] & & S \arrow[dl, "f"] \\
& T. &
\end{tikzcd} \tag{1.28}$$

The function $f \circ g\colon \{*\} \longrightarrow T$ picks out the output of f. If g performs the assignment $* \mapsto s$, then $f \circ g$ performs the assignment $* \mapsto f(s)$. □

Example 1.4.31. If $f\colon S \longrightarrow T$ and A is a subset of S with inclusion function $inc\colon A \hookrightarrow S$, then the restriction of f to A is the function $f|\colon A \longrightarrow T$ which is given as the composition

$$\begin{tikzcd}
A \arrow[rr, hook, "inc"] \arrow[dr, "f|=f\circ i"'] & & S \arrow[dl, "f"] \\
& T. &
\end{tikzcd} \tag{1.29}$$

□

Theorem 1.4.32. The three properties of functions that we saw in Definition 1.4.21 can be described with function composition. For non-empty sets S and T, the function $f\colon S \longrightarrow T$ is

- **one-to-one** if and only if there exists a $g\colon T \longrightarrow S$ such that $g \circ f = id_S$

$$\begin{tikzcd}
S \arrow[rr, "f"] \arrow[dr, "id_S"'] & & T \arrow[dl, "g"] \\
& S. &
\end{tikzcd} \tag{1.30}$$

  (Proof. The existence of a g such that the diagram commutes implies f is one-to-one. If $f(s) = f(s')$ then apply g to both sides of the equation and get $g(f(s)) = g(f(s'))$. But $g \circ f = id_S$ implies $s = s'$.
  If f is one-to one, then there exists a g such that the diagram commutes. Let $t \in T$. Assign $g(t)$ to be the unique s such that $f(s) = t$. If t is not an output of f then it does not matter what value you give to $g(t)$.)
- **onto** if and only if there exists a $g\colon T \longrightarrow S$ such that $f \circ g = id_T$

$$\begin{tikzcd}
T \arrow[rr, "g"] \arrow[dr, "id_T"'] & & S \arrow[dl, "f"] \\
& T. &
\end{tikzcd} \tag{1.31}$$

  (Proof. The existence of a g implies f is onto. The function f is onto because for any $t \in T$, the function g has $g(t) = s$ for some $s \in S$. This s gives an input to f whose output is t, i.e., $f(s) = f(g(t)) = id_T(t) = t$.
  Onto implies the existence of g such that the diagram commutes. Let $g(t)$ equal any s such that $f(s) = t$. There must be one such t because f is onto. This proof assumes the axiom of choice which we will meet later.)

- **isomorphism** or **one-to-one correspondence** if and only if there exists a $g\colon T \longrightarrow S$ such that $g \circ f = id_S$ and $f \circ g = id_T$. Or putting the previous two triangles together, we have

$$\begin{tikzcd}
S \arrow[rr, "f"] \arrow[dr, "id_S"'] & & T \arrow[dl, "g"] \arrow[dr, "id_T"] & \\
& S \arrow[rr, "f"'] & & T.
\end{tikzcd} \tag{1.32}$$

Another way to express this is the following diagram.

$$\begin{tikzcd}
S \arrow[r, bend left, "f"] \arrow[loop left, "id_S"] \arrow[loop left, looseness=12, "g\circ f \;="'] & T \arrow[l, bend left, "g"] \arrow[loop right, "id_T"'] \arrow[loop right, looseness=12, "= \; f\circ g"]
\end{tikzcd} \tag{1.33}$$

We will see variations of this diagram again and again.

★

A second operation of functions is an **extension**. In detail, if $f\colon R \longrightarrow T$ is a function and R is a subset of S with the inclusion function $inc\colon R \hookrightarrow S$, then an extension of f along inc is a function $\hat{f}\colon S \longrightarrow T$ such that the following commutes

$$\begin{tikzcd}
R \arrow[rr, hook, "inc"] \arrow[dr, "f"'] & & S \arrow[dl, "\hat{f}"] \\
& T. &
\end{tikzcd} \tag{1.34}$$

In English, $\hat{f}$ extends f to a larger domain.

Example 1.4.33. As a simple example, consider R to be a set of students and

$$f\colon R \longrightarrow \{A, B, C, D, F\} \tag{1.35}$$

assigns every student a grade. If some new students came into the class, the teacher would have to extend f to give grades to all the students (including the new ones) as $\hat{f}\colon S \longrightarrow \{A, B, C, D, F\}$. We want $\hat{f}$ to assign the same grades as f did for any of the original students. This is clear with the following commutative diagram:

$$\begin{tikzcd}
\{\text{original students}\} \arrow[rr, hook, "inc"] \arrow[dr, "f"'] & & \{\text{original and new students}\} \arrow[dl, "\hat{f}"] \\
& \{A, B, C, D, F\} &
\end{tikzcd} \tag{1.36}$$

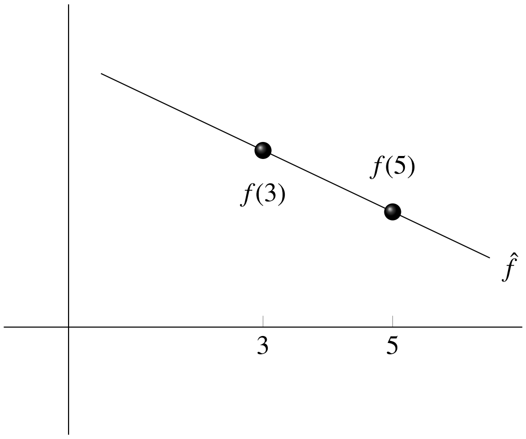
□

Example 1.4.34. Let $\{3, 5\}$ be a set of two real numbers. There is an obvious inclusion of the two real numbers into the set of all numbers $inc\colon \{3, 5\} \hookrightarrow \mathbf{R}$. Let $f\colon \{3, 5\} \longrightarrow \mathbf{R}$ be any function that picks two values. Then, there exists a function $\hat{f}\colon \mathbf{R} \longrightarrow \mathbf{R}$ that extends f.

$$\begin{tikzcd}
\{3, 5\} \arrow[rr, hook, "inc"] \arrow[dr, "f"'] & & \mathbf{R}. \arrow[dl, "\hat{f}"] \\
& \mathbf{R} &
\end{tikzcd} \tag{1.37}$$

This extension is another way of describing the simple idea that given any two points on the plane, there is a straight line that passes through both of them. This can be visualized as

(1.38)



In detail, the extended line is given by the following formula

$$\hat{f}(x) = mx+b = \frac{\Delta y}{\Delta x}x+b = \frac{f(5) - f(3)}{5 - 3}x+\frac{5f(3) - 3f(5)}{5 - 3} = \frac{f(5) - f(3)}{2}x+\frac{5f(3) - 3f(5)}{2}. \tag{1.39}$$

Be aware that this extension is one of many ways to extend f. There is nothing special about this extension other than it is usually taught in the first year of high school. □

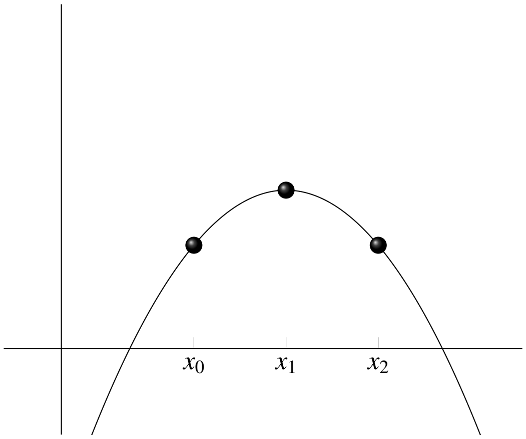
Thinking of a straight line as a function, the previous example of an extension can be ... extended...

Example 1.4.35. Let $\{x_0, x_1, x_2, \ldots, x_n\}$ be a set of $n+1$ different real numbers and let $inc\colon \{x_0, x_1, x_2, \ldots, x_n\} \hookrightarrow \mathbf{R}$ be the inclusion function. Every $f\colon \{x_0, x_1, x_2, \ldots, x_n\} \longrightarrow \mathbf{R}$ has an extension called $\hat{f}\colon \mathbf{R} \longrightarrow \mathbf{R}$ along inc which is a polynomial function of degree at most n.

$$\begin{tikzcd}
\{x_0, x_1, x_2, \ldots x_n\} \arrow[rr, hook, "inc"] \arrow[dr, "f"'] & & \mathbf{R} \arrow[dl, "\hat{f}"] \\
& \mathbf{R} &
\end{tikzcd} \tag{1.40}$$

This can be visualized as

(1.41)



The function $\hat{f}$ is called the "Lagrange interpolating polynomial" of the points described by f. (We will not use this in the text.) □

While extensions are usually about inclusion functions, we can also use the setup of an extension for functions that are not inclusion functions.

Example 1.4.36. A function $f\colon S \longrightarrow T$ is a **constant function** if it outputs the same value for any input. That means there is some $t_0 \in T$ such that for all $s \in S$ we have $f(s) = t_0$. We can describe a constant function by using the same notation of an extension but without the inclusion function. In particular, f is a constant function if there exists an "extension" $\hat{f}\colon \{*\} \longrightarrow T$ of f as in the diagram

$$\begin{tikzcd}
S \arrow[rr, "!"] \arrow[dr, "f"'] & & \{*\} \arrow[dl, "\hat{f}"] \\
& T &
\end{tikzcd} \tag{1.42}$$

where the function $!\colon S \longrightarrow \{*\}$ is the unique function that always outputs $*$, the only element it can output. Another way of saying this, is that f is a constant function if it can be written as a function that factors through $\{*\}$. □

Exercise 1.4.37. Show that if $id_S\colon S \longrightarrow S$ can be extended along the function $f\colon S \longrightarrow T$, then f is a one-to-one function.

1.4.37 This is essentially the content of Diagram (1.30).

■

The third operation of functions is a **lifting**. Consider an onto function $p\colon T \longrightarrow T'$. Let $f\colon S \longrightarrow T'$ be any function. A lifting of f along p is a function $\hat{f}\colon S \longrightarrow T$ that makes the following triangle commute

$$\begin{tikzcd}
S \arrow[rr, "\hat{f}"] \arrow[dr, "f"'] & & T \arrow[dl, "p"] \\
& T'. &
\end{tikzcd} \tag{1.43}$$

In a sense, we "lift " the map f from the target of p to the source of p. Here is the simplest example of lifting.

Example 1.4.38. Consider the following commutative diagram:

$$\begin{tikzcd}
\{*\} \arrow[rr, "\hat{f}"] \arrow[dr, "f"'] & & T \arrow[dl, "p"] \\
& T'. &
\end{tikzcd} \tag{1.44}$$

Here a function $f\colon \{*\} \longrightarrow T'$ picks out an element of T'. A lifting of f is $\hat{f}\colon \{*\} \longrightarrow T$ which picks out an element of T. The fact that the diagram is commutative, means that

p will take outputs of $\hat{f}$ to outputs of f. In other words, if f picked out $t_0 \in T'$, then $\hat{f}$ will pick an element in T that will map onto t_0 under p. There might be many liftings. The set of all possible elements that a lifting can pick is denoted $p^{-1}(t_0) \subseteq T$ which is called the "preimage" of p. □

Example 1.4.39. Here is a cute example of a lifting from the world of politics. Let T be the set of 320 million American citizens and let T' be the set of 50 states. The function p takes every citizen to the state they live in. Let S be a set of three elements such as $\{a, b, c\}$. The function $f\colon S \longrightarrow T'$ chooses three states. A lifting of f along p is a function $\hat{f}\colon S \longrightarrow T$ which will choose three citizens. Each of the three will be from the states f chooses. There are obviously many such liftings. □

Example 1.4.40. Let us build on the last example. Let T, T' and p be as in the last example. Let S be the set $\{a, b, c\} \times T'$, i.e., pairs of letters and states. The $f\colon S \longrightarrow T'$ function is defined as follows: $f(b, \text{New Jersey}) = \text{New Jersey}$, i.e., f takes a letter and a state and outputs the same state. Notice that for each state, there are three elements in S that go to that state. For example

$$f(a, \text{New Jersey}) = f(b, \text{New Jersey}) = f(c, \text{New Jersey}) = \text{New Jersey}.$$

A lifting of f along p is a function $\hat{f}\colon S \longrightarrow T$ which will choose three citizens from each state. There are many such liftings. □

Exercise 1.4.41. Show that if $id_T\colon T \longrightarrow T$ can be lifted along the function $f\colon S \longrightarrow T$, then f is an onto function. ■

One can see these three operations — composition, extension, and lifting — as three sides of a triangle:

$$\begin{tikzcd}
\bullet \arrow[rr, "\text{Lifting}"] \arrow[dr, "\text{Composition}"'] & & \bullet \arrow[dl, "\text{Extension}"] \\
& \bullet &
\end{tikzcd} \tag{1.45}$$

Each side uses the other two sides as the input to the operation. Composition will be used on every page of this text. We will see (especially in Section 9.2) that the extension and lifting operations are very important in many contexts besides sets and functions.

The rest of this mini-course will be concerned with equivalence relations, graphs, and groups. These three subjects are structures based on sets. The eager reader might want to skip these pages and go directly to the beginning of the next chapter.

Equivalence Relations

We are not only interested in how a set is related to other sets. Sometimes the elements of a set are related to each other in interesting ways.

Definition 1.4.42. Let S be a set. A **relation** on S is a subset R of the set $S \times S$. The ordered pair (s_1, s_2) in R means s_1 is related to s_2.

Example 1.4.43. Let S be the set of citizens of the United States. Consider the following relations on this set.

- R_1 consists of those (s, t) where s and t are cousins.
- R_2 consists of those (s, t) where s is the same age or older than t.
- R_3 consists of those (s, t) where s and t live in the same state.
- R_4 consists of those (s, t) where s and t belong to the same political party.

□

The following three properties of a relation will characterize the notion of "sameness."

Definition 1.4.44. The relation $R \subseteq S \times S$ on a set S is

- **reflexive** if every element is related to itself: for all s in S, (s, s) is in R.
- **symmetric** whenever one element is related to another, then the other is related to the first: for all s and t in S, if (s, t) is in R, then (t, s) is in R.
- **transitive** whenever s is related to t and t is related to u, then s is related to u: for all s, t and u in S, if (s, t) is in R and (t, u) is in R, then (s, u) is in R.

Example 1.4.45. Let us look which properties are satisfied from the relations of Example 1.4.43.

- The cousin relation R_1 is not reflexive (no one is their own cousin); it is symmetric; but it is not transitive (x can be a cousin to y through y's mother's side and y can be a cousin to z through y's father's side. In this case x will, in general, not be cousins to z.)
- The older relation R_2 is reflexive (everyone is the same age as themselves), not symmetric (if x is older than y then y is not older than or the same age as x), and it is transitive.
- The state relation R_3 is reflexive, symmetric, and transitive.
- The political party relation R_4 is reflexive, symmetric, and transitive.

□

Many times a set of elements can be split up or partitioned into different subsets where each subset will have all the elements with a particular property. For example, the set of cars can be split up by color. So there will be the subset of blue cars, the subset of red cars, the subset of green cars, etc. The collection of all such subsets will form a set itself. Formally this can be said as follows.

Definition 1.4.46. A relation on a set is an **equivalence relation** if it is reflexive, symmetric, and transitive. We write such relations as $\sim \subseteq S \times S$ and write $r \sim s$ for $(r, s) \in \sim$. With an equivalence relation on the set S, we can describe disjoint subsets of S called **equivalence classes**. If s is an element of S, then the equivalence class of s is the set of all elements that are related to it:

$$[s] = \{r \in S : r \sim s\} \tag{1.46}$$

That is, $[s]$ is the set of all elements that are "the same" as s. For a given set S and an equivalence relation $\sim$ on S, we form a **quotient set** denoted $S/\sim$. The elements of $S/\sim$ are all the equivalence classes of elements in S. There is an obvious **quotient function** from S to $S/\sim$ that takes s to $[s]$.

Example 1.4.47. Let us examine the equivalence classes for the equivalence relations of Example 1.4.43.

- Each equivalence class for the relation R_3 consists of all the residents of a particular state. The quotient set contains the 50 equivalence classes corresponding to the 50 States (we are ignoring abnormalities like Guam and Washington D.C.). The quotient function takes every citizen to the state in which they reside.
- Each equivalence class for the relation R_4 consists of all the people belonging to a particular political party. The quotient set is a set whose elements correspond to political parties. The quotient function takes every citizen to the political party to which they belong (we are ignoring independents.)

□

Graphs

A directed graph is a common structure (based on sets) that has applications everywhere. Directed graphs also have many similarities to categories.

Definition 1.4.48. A **directed graph** $G = (V(G), A(G), src_G, trg_G)$ is

- a set of vertices, $V(G)$, and
- a set of arrows, $A(G)$.

Furthermore,

- every arrow has a source: there is a function $src_G \colon A(G) \longrightarrow V(G)$, and
- every arrow has a target: there is a function $trg_G \colon A(G) \longrightarrow V(G)$.

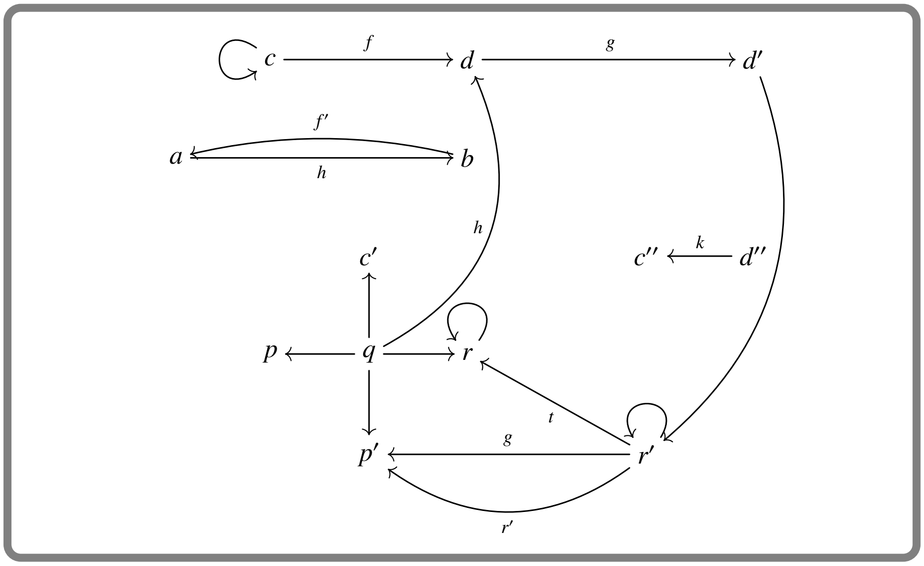


Figure 1.3: An example of a directed graph.

> If f is an element of $A(G)$ with $src_G(f) = x$ and $trg_G(f) = y$, we draw this arrow as
>
> $$x \xrightarrow{\quad f \quad} y\ . \tag{1.47}$$

An example of a graph is Figure 1.3.

Example 1.4.49. Graphs are everywhere.

- A street map can be thought of as a directed graph where the vertices are street corners and there is an arrow from one corner to the other if there is a one-way street between them. When there is a two-way street, we might write it like this

  $$* \rightleftarrows * \qquad \text{or} \qquad * \;\rule[0.5ex]{5em}{0.4pt}\; *. \tag{1.48}$$

  Such an arrow is called a "symmetric edge."
- An electrical circuit can be viewed as a directed graph. The vertices are the branching points and edges might have resistors, batteries, capacitors, diodes, etc. The arrows describe the direction of the flow of electricity.
- Computer networks can be seen as directed graphs where the vertices are computers and there is an arrow from one computer to another if there is a way for the first computer to communicate with the second.
- The billions of web pages in the World Wide Web form a directed graph. The vertices are the web pages and there is an arrow if there is a link from one web page to another.

- Facebook can be seen as a directed graph. Every personal Facebook account is a vertex, and there are arrows between two Facebook accounts if they are friends. Notice that if x is friends with y then y must be friends with x. So all the arrows are symmetric edges and the graph is called symmetric.
- All the people on Earth form a graph. The vertices are the people. There is an arrow from x to y if x knows y. (We are not being specific as to what it means to “know” someone.) There is an idea called “six degrees of separation,” which says that in this graph, you never need to traverse more than six arrows to get from any person to any other person. We are all connected!
- The collection of all sets and functions form a giant graph. In detail, the vertices are all sets. The arrows are functions from a set to another set. (This will be a motivating example of a category.)

□

A graph homomorphism is a way of mapping one graph to another. This will be similar to what happens when we talk about mapping one category to another category. Basically the vertices map to the vertices and the arrows map to the arrows but we insist that they match up well.

Definition 1.4.50. Let $G = (V(G), A(G), src_G, trg_G)$ and $G' = (V(G'), A(G'), src_{G'}, trg_{G'})$ be graphs. A **graph homomorphism** $H\colon G \longrightarrow G'$ consists of

- A function that assigns vertices to vertices, $H_V\colon V(G) \longrightarrow V(G')$.
- A function that assigns arrows to arrows, $H_A\colon A(G) \longrightarrow A(G')$.

These two maps must respect the source and target of each arrow. That means:

- For all f in $A(G)$, $H_V(src_G(f)) = src_{G'}(H_A(f))$.
- For all f in $A(G)$, $H_V(trg_G(f)) = trg_{G'}(H_A(f))$.

Saying that these axioms are satisfied is the same as saying that the following two squares commute:

$$\begin{tikzcd}
A(G) \arrow[r, "H_A"] \arrow[d, "src_G"'] & A(G') \arrow[d, "src_{G'}"] & & A(G) \arrow[r, "H_A"] \arrow[d, "trg_G"'] & A(G') \arrow[d, "trg_{G'}"] \\
V(G) \arrow[r, "H_V"'] & V(G') & & V(G) \arrow[r, "H_V"'] & V(G').
\end{tikzcd} \tag{1.49}$$

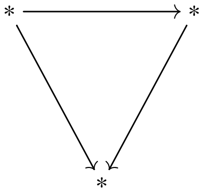
Another way to understand these requirements is to see what the maps H_V and H_A do to a single arrow f (that is, $\begin{tikzcd} f \arrow[r, maps to, squiggly] & H_A(f) \end{tikzcd}$. We use the wavy arrow so that the reader can see what is going on easier.)

$$\begin{tikzcd}
\underline{\textbf{Graph G}} & & \underline{\textbf{Graph G}'} \\
src_G(f) \arrow[rr, mapsto, squiggly, "H_V"] \arrow[dd, "f"'] & & H_V(src_G(f)) = src_{G'}(H_A(f)) \arrow[dd, "H_A(f)"] \\
{} \arrow[rr, mapsto, squiggly, "H_A"] & & {} \\
trg_G(f) \arrow[rr, mapsto, squiggly, "H_V"] & & H_V(trg_G(f)) = trg_{G'}(H_A(f))
\end{tikzcd} \tag{1.50}$$

Just as we can determine many properties of sets by examining functions, we can also determine many properties of graphs by examining graph homomorphisms from simple graphs.

Example 1.4.51.

- A vertex of a graph G can be described by a graph homomorphism from the one-vertex graph $(*)$ (no arrows) as follows $H\colon * \longrightarrow G$.
- A directed edge of a graph can be determined by a graph homomorphism from the graph $* \longrightarrow *$ to G.
- A triangle in a graph G can be determined by a graph homomorphism from the graph



(1.51)

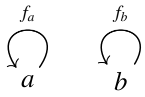
to the graph G.
- A path of length n in a graph G can be determined by a graph homomorphism from the "snake" graph

$$* \longrightarrow * \longrightarrow * \longrightarrow \cdots \longrightarrow * \tag{1.52}$$

of length n to G.

□

Exercise 1.4.52. Prove the following fact about a graph G. If there does *not* exist a graph homomorphism that is surjective on vertices from G to the graph



(1.53)

then G is weakly connected (for any two vertices, there is a sequence of edges in either direction between them.)

1.4.52 We prove the contrapositive. If G is not weakly connected, then G will be able to split the graph into two parts with no arrow from one part to the other part. There is then a function from the graph to this two-vertex graph, where the nodes of one part of the graph go to a, and all the arrows of that part go to the single arrow f_a. The nodes of the other part go to b and all the arrows of that part go to f_b. If the graph is weakly connected, then this partitioning would not be possible.

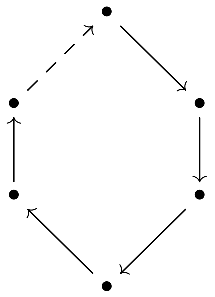
■

Exercise 1.4.53. Use graph homomorphisms to determine different types of paths in a graph.

- How do you describe a simple path in a graph (a simple path is a path that does not have repeated vertices)?
- What about a cycle of length n (a cycle is a path that starts and ends at the same vertex)?
- Do the same for a simple cycle of length n (a simple cycle is a cycle in which the only repeating vertex is the starting point which is the ending point).

1.4.53

- One-to-one graph homomorphisms from the "snake" graph, Diagram (1.52), to any graph will correspond to simple paths.
- A cycle can be described as a graph homomorphisms from "ring" graphs. These are graphs of the form



  where the dashed arrow is an ellipsis that possibly denotes many more arrows.
- One-to-one graph homomorphisms from the "ring" graphs will correspond to simple cycles.

■

Exercise 1.4.54. Show that the composite of graph homomorphisms is a graph homomorphism. Also show that the composition is an associative operation.

1.4.54 Let $H\colon G \longrightarrow G'$ and $H'\colon G' \longrightarrow G''$ be graph homomorphisms. Then $H' \circ H$: $G \longrightarrow G''$ is a graph homomorphism. The fact that $H' \circ H$ preserves the sources of arrows amounts to the commutativity of the following diagram

$$\begin{tikzcd}
A(G) \arrow[r, "H_A"] \arrow[d, "src_G"'] & A(G') \arrow[r, "H'_A"] \arrow[d, "src_{G'}"] & A(G'') \arrow[d, "src_{G''}"] \\
V(G) \arrow[r, "H_V"'] & V(G') \arrow[r, "H'_V"'] & V(G''),
\end{tikzcd}$$

which is assured because each square commutes. A similar argument must be made to show that $H' \circ H$ preserves targets. The proof of the associativity of the composition of graph homomorphisms comes from the fact that graph homomorphisms are functions and is similar to the solution to Exercise 1.4.29.

■

Exercise 1.4.55. Define the **identity graph homomorphism**, I_G, for any graph G. Show that if $H\colon G \longrightarrow G'$ is a graph homomorphism then $H \circ I_G = H$ and $I_{G'} \circ H = H$.

1.4.55 The identity graph homomorphism is defined as $I_{G,A}(x) = x$ and $I_{G,V}(f) = f$. This graph homomorphism preserves the sources and targets of the arrows for trivial reasons. The fact that it acts like a unit to composition is because it is essentially two identity functions.

■

Groups

Another important structure that is based on sets and related to categories is a group. It is nice to see the definition of a group from a function perspective.

First a discussion of operations. We all know what we mean by operations on numbers. If you take numbers x and y, you can perform the addition operation, $x + y$. You can also perform other operations like $x - y$ or $y \cdot x$. All of these are examples of binary operations. Operations are really just functions. For a given set, S, a **binary operation** is a function $f\colon S \times S \longrightarrow S$. For two elements s and s', we write the value of f as $f(s, s')$. A **unary operation** is a function that takes one element of S and outputs one element of S, i.e., $f\colon S \longrightarrow S$. An example of a unary operation is the inverse operation that takes x and returns x^{-1}. There are also **ternary operations** $f\colon S \times S \times S \longrightarrow S$ and n**-ary operations**

$$f\colon \underbrace{S \times S \times \cdots \times S}_{n \text{ times}} \longrightarrow S \tag{1.54}$$

for all natural numbers n. If $n = 0$, then we write the 0-ary product as the set with one element $\{*\}$ and a **0-ary operation** is written as $f\colon \{*\} \longrightarrow S$ which basically picks out an element of S. Such an operation describes an element that does not change, i.e., a constant.

Let us put this all together and give the formal definition of a group.

Definition 1.4.56. A **group** $(G, \star, e, (\ )^{-1})$ is a set G with the following three operations:

- A binary operation: a function $\star\colon G \times G \longrightarrow G$.
- An identity: there is a special element e in G called the identity of the group. This can be stated in a functional way: there is a 0-ary operation $u : \{*\} \longrightarrow G$ where $u(*) = e$.
- An inverse operation: a unary operation $(\ )^{-1} \colon G \longrightarrow G$

These three operations satisfy the following axioms:

- The binary operation is associative: for all x, y and z, we have $(x \star y) \star z = x \star (y \star z)$.
- The identity acts like a unit of the binary operation (like when you multiply a number with 1, the result does not change, i.e., $1 \cdot n = n$ hence 1 is a "unit"): for all x, $x \star e = x = e \star x$.
- Applying the binary operation to an element with its inverse gives the identity: for all x in G, $x \star x^{-1} = e = x^{-1} \star x$

Example 1.4.57. Here are some examples of groups.

- The additive integers: $(\mathbf{Z}, +, 0, -(\ ))$. Addition and negation are the usual operations.
- The additive real numbers: $(\mathbf{R}, +, 0, -(\ ))$. Addition and negation are the usual operations.
- The multiplicative positive reals: $(\mathbf{R}^+, \cdot, 1, (\ )^{-1})$ where $\mathbf{R}^+$ are the positive real numbers, the operation $\cdot$ is multiplication, and the function $(\ )^{-1}$ takes r to $\frac{1}{r}$.
- Clock arithmetic: $(\{0, 1, 2, 3, \ldots, 11\}, +, 0, -)$ where addition and subtraction is going around the clock. 0 is the unit because when you add 0 to any number you get back to the original number. Notice that we could have used another number besides 11. In fact, any non-negative integer would have worked.
- The **trivial group**: $(\{0\}, +, 0, -)$. This is the world's smallest group. It has only one element and the operations work as expected.

□

Parts of the three axioms of a group can be seen as commutative diagrams in Figure 1.4. Around each of the commutative diagrams are maps showing the values of the functions on elements.

Exercise 1.4.58. The second diagram in Figure 1.4 shows the $x = x \star e$ axiom. Give a commutative diagram for the $x = e \star x$ axiom.

1.4.58 It is essentially the same diagram but change the $G \times \{*\}$ to $\{*\} \times G$.

■

Exercise 1.4.59. The third diagram in Figure 1.4 shows the $e = x \star x^{-1}$ axiom. Give the commutative diagram for the $e = x^{-1} \star x$ axiom.

1.4.59 It is essentially the same diagram with the $id \times (\ )^{-1}$ map switched to $(\ )^{-1} \times id$.

■

Important Categorical Idea 1.4.60. Descriptions Using Morphisms. Many times, even when we have a nice, clear definition or description of a mathematical structure in terms of elements, we still desire a description in terms of functions or morphisms. The reason that a description using functions or morphisms is important is that once we have it, we can use it in many different categories. Whereas a description in terms of elements is good only in one context, a description in terms of functions or morphisms can be used in many different categories and contexts. For example, we will see this morphism definition of a group arise in other contexts besides sets and functions. ○

Just as a function is a way of mapping one set to another, and a graph homomorphism is a way of mapping one graph to another, a group homomorphism is a way of mapping one group to another.

Definition 1.4.61. Let $(G, \star, e, (\ )^{-1})$ and $(G', \star', e', (\ )'^{-1})$ be groups. A **group homomorphism** $f\colon (G, \star, e, (\ )^{-1}) \longrightarrow (G', \star', e', (\ )'^{-1})$ is a function $f\colon G \longrightarrow G'$ that satisfies the following two axioms

- The function respects the group operation: for all $x, y \in G$, $f(x \star y) = f(x) \star' f(y)$
- The function respects the unit: $f(e) = e'$

We can write these two requirements as the following two commutative diagrams.

$$\begin{tikzcd}
G \times G \arrow[r, "f \times f"] \arrow[d, "\star"'] & G' \times G' \arrow[d, "\star'"] \\
G \arrow[r, "f"'] & G'
\end{tikzcd}
\qquad
\begin{tikzcd}
& \{*\} \arrow[dl, "u"'] \arrow[dr, "u'"] & \\
G \arrow[rr, "f"'] & & G'
\end{tikzcd}
\tag{1.55}$$

Technical Point 1.4.62. We did not insist that f respect inverses. Do not worry about it. It is true without saying it because it is a consequence of the other two axioms. First notice that in any group, x^{-1} is the unique inverse of the element x. To see this, imagine that x has two inverses y and y'. Consider the following sequence of equalities

$$\begin{aligned}
y &= y \star e && \text{by the unit axiom} \\
&= y \star (x \star y') && \text{because } y' \text{ is the inverse of } x \\
&= (y \star x) \star y' && \text{by the associativity axiom} \\
&= e \star y' && \text{because } y \text{ is the inverse of } x \\
&= y' && \text{by the unit axiom}
\end{aligned}$$

This shows that $y = y'$. Now let us use this fact to show that inverses are preserved by group homomorphisms. First consider

$$e' = f(e) = f(x \star x^{-1}) = f(x) \star' f(x^{-1}). \tag{1.56}$$

This shows that the inverse of $f(x)$ is $f(x^{-1})$. Since inverses are unique, we proved $f(x)^{-1} = f(x^{-1})$. ♡

Example 1.4.63. Here are some examples of group homomorphisms.

- There is always a unique group homomorphism from any group to the trivial group where every element of the group goes to 0 of the trivial group.
- There is always a unique group homomorphism from the trivial group to any group in which the 0 of the trivial group goes to the identity of the group.
- There is an inclusion group homomorphism $inc\colon \mathbf{Z} \longrightarrow \mathbf{R}$.
- There is a group homomorphism $\mathbf{Z} \longrightarrow \{0, 1, 2, 3, \ldots, 11\}$ that takes every whole number x and sends it to the remainder when x is divided by 12.
- Let b be some positive real number called the "base". There is a exponential function
$$b^{(\ )}\colon (\mathbf{R}, +, 0, -) \longrightarrow (\mathbf{R}^+, \cdot, 1, (\ )^{-1}) \tag{1.57}$$
that takes a real number r and sends it to b^r. The two requirements to be a group homomorphism turn out to mean that $b^{r+r'} = b^r \cdot b^{r'}$ ($b^{(\ )}$ takes addition to multiplication) and $b^0 = 1$.

- There is a logarithm function (that is the inverse of the exponential function)

$$Log_b \colon (\mathbf{R}^+, \cdot, 1, (\ )^{-1}) \longrightarrow (\mathbf{R}, +, 0, -). \tag{1.58}$$

  The function Log_b takes a positive real number r to $Log_b(r)$. The requirements to be a group homomorphism are the well-known facts that $Log_b(r \cdot r') = Log_b(r) + Log_b(r')$ (Log_b takes multiplication to addition) and $Log_b(1) = 0$.

□

Exercise 1.4.64. Show that the composite of group homomorphisms is a group homomorphism. Also show that the composition is an associative operation.

1.4.64 Let $f \colon G \longrightarrow G'$ and $f' \colon G' \longrightarrow G''$ be group homomorphisms. Then $(f' \circ f)(x) = f'(f(x))$. To show that composition preserves the group operations: for $x, y \in G$

$$(f' \circ f)(x \star y) = f'(f(x \star y)) = f'(f(x) \star' f(y)) = f'(f(x)) \star'' f'(f(y)) = (f' \circ f)(x) \star'' (f' \circ f)(y)$$

and

$$(f' \circ f)(e) = f'(f(e)) = f'(e') = e''.$$

■

Exercise 1.4.65. Define the **identity group homomorphism**, id_G, for every group $(G, \star, 0, (\ )^{-1})$. Show that if $f \colon (G, \star, e, (\ )^{-1}) \longrightarrow (G', \star', e', (\ )'^{-1})$ is a group homomorphism then $f \circ id_G = f$ and $id_{G'} \circ f = f$.

1.4.65 The identity trivially respects the group operations. The last part follows from the fact that group homomorphisms are simply functions (that satisfy certain properties.)

■

Suggestions for Further Study

Most of the material found in this section can be found in any discrete mathematics or finite mathematics textbook e.g. [226, 225]. This material can also be found in many pre-calculus textbooks.

The importance of looking at functions between sets is central to all of category theory. This is also stressed by two books coauthored by F. William Lawvere, one of the leaders of category theory. Together with Robert Rosebrugh, Lawvere wrote *Sets for Mathematics* [165] and together with Stephen H. Schanuel he wrote *Conceptual Mathematics* [166].

The novice can find basic group theory in any introduction to modern algebra or abstract algebra, e.g., [15, 79].

This idea that most of the operations on functions can be seen as compositions, extensions and liftings (as in Diagram (1.45)) was taken from [265] where much of category theory is built from these operations. We will see more of extensions and liftings in Section 9.2.

If you really want to learn more about categorical thinking, roll up your sleeves and let us get to the rest of this book!

$$\begin{tikzcd}
x,y,z \arrow[rrrr, mapsto] \arrow[ddd, mapsto] & & & & x\bigstar y,z \arrow[ddd, mapsto] \\
& G\times G\times G \arrow[r, "\star\times id"] \arrow[d, "id\times\star"'] & G\times G \arrow[d, "\star"] & & \\
& G\times G \arrow[r, "\star"'] & G & & \\
x,y\bigstar z \arrow[rrr, mapsto] & & & x\bigstar(y\bigstar z) \arrow[r, equal] & (x\bigstar y)\bigstar z
\end{tikzcd}$$

$$\begin{tikzcd}
x \arrow[rrr, mapsto] \arrow[ddd, mapsto] & & & x,* \arrow[ddd, mapsto] \\
& G \arrow[r, "\cong"] \arrow[d, "id"'] & G\times\{*\} \arrow[d, "id\times u"] & \\
& G & G\times G \arrow[l, "\star"] & \\
x \arrow[r, equal] & x\bigstar e & & x,e \arrow[ll, mapsto]
\end{tikzcd}$$

$$\begin{tikzcd}
x \arrow[rrr, mapsto] \arrow[dddd, mapsto] & & & x,x \arrow[dd, mapsto] \\
& G \arrow[r, "\Delta"] \arrow[dd] & G\times G \arrow[d, "id\times(\ )^{-1}"] & \\
& & G\times G \arrow[d, "\star"] & x,x^{-1} \arrow[dd, mapsto] \\
& \{*\} \arrow[r, "u"'] & G & \\
* \arrow[rr, mapsto] & & e \arrow[r, equal] & x\bigstar x^{-1}.
\end{tikzcd}$$

Figure 1.4: Commutative diagrams of some of the axioms of a group

Chapter 2

Categories

A good stack of examples, as large as possible, is indispensable for a thorough understanding of any concept, and when I want to learn something new, I make it my first job to build one.
Paul Halmos
[100], page 63.

With the ideas of set and functions in hand, we move on to the world of categories. Section 2.1 begins with a formal definition of categories. It then proceeds to form a giant stack of examples of categories from all over. In Section 2.2 we discuss some simple properties of morphisms in categories. We elaborate on some simple categories related to a category in Section 2.3. The chapter ends with Section 2.4, which is a mini-course that teaches the basics of linear algebra. The study of linear algebra is essentially an in-depth exploration of the category of vector spaces.

2.1 Basic Definitions and Examples

Before formally defining a category, let us summarize what we saw in Section 1.4 concerning sets and functions. The collection of sets and functions form a category. By carefully examining this collection, we will see what is needed in the definition of a category.

Example 2.1.1. Consider the collection of all sets. There are functions between sets. If f is a function from set S to set T, then we write it as $f\colon S \longrightarrow T$. We call S the **domain** of f and T the **codomain** of f. Certain functions can be composed: for $f\colon S \longrightarrow T$ and $g\colon T \longrightarrow U$, there exists a function $g \circ f\colon S \longrightarrow U$ which is defined for s in S as $(g \circ f)(s) = g(f(s))$. This composition operation is associative, which means that for $f\colon S \longrightarrow T$, $g\colon T \longrightarrow U$, and $h\colon U \longrightarrow V$, both ways of associating the functions $h \circ (g \circ f)$ and $(h \circ g) \circ f$ are equal to the function described as follows

$$s \mapsto f(s) \mapsto g(f(s)) \mapsto h(g(f(s))). \tag{2.1}$$

That is, $h \circ (g \circ f) = (h \circ g) \circ f$ and on s of S this function has the value $h(g(f(s)))$. For every set S, there is a function $id_S : S \longrightarrow S$, which is called the **identity function** and is defined for s in S as $id_S(s) = s$. These identity functions have the following properties: for all $f\colon S \longrightarrow T$, it is true that $f \circ id_S = f$ and $id_T \circ f = f$. The collection of sets and functions form a category called $\mathbb{S}\mathrm{et}$. This category is easy to understand, and we use it to hone our ideas about many structures of category theory. □

Now for the formal definition of a category.

Definition 2.1.2. A **category** $\mathbb{A}$ is a collection of **objects** $Ob(\mathbb{A})$ and a collection of **morphisms** $Mor(\mathbb{A})$ which has the following structure:

- Every morphism has an object associated to it called its domain: there is a function $dom_{\mathbb{A}} : Mor(\mathbb{A}) \longrightarrow Ob(\mathbb{A})$.
- Every morphism has an object associated to it called its codomain: there is a function $cod_{\mathbb{A}} : Mor(\mathbb{A}) \longrightarrow Ob(\mathbb{A})$. We write

$$f\colon a \longrightarrow b \qquad \text{or} \qquad a \xrightarrow{\quad f \quad} b \tag{2.2}$$

  for the fact that $dom_{\mathbb{A}}(f) = a$ and $cod_{\mathbb{A}}(f) = b$.
- Adjoining morphisms can be composed: if $f\colon a \longrightarrow b$ and $g\colon b \longrightarrow c$, then there is an associated morphism $g \circ f : a \longrightarrow c$. We can write these morphisms as

$$\begin{tikzcd} a \arrow[r, "f"] \arrow[rr, bend left, "g\circ f"] & b \arrow[r, "g"] & c. \end{tikzcd} \tag{2.3}$$

- Every object has an identity morphism: there is a function $ident_{\mathbb{A}} : Ob(\mathbb{A}) \longrightarrow Mor(\mathbb{A})$. We denote the identity of a as $id_a : a \longrightarrow a$ or

$$\begin{tikzcd} a. \arrow[loop above, "id_a"] \end{tikzcd} \tag{2.4}$$

This structure must satisfy the following two axioms:

- Composition is associative: given $f\colon a \longrightarrow b$, $g\colon b \longrightarrow c$, and $h\colon c \longrightarrow d$, the two ways of composing these maps are equal:

$$h \circ (g \circ f) = (h \circ g) \circ f, \tag{2.5}$$

  i.e., they are the same map from a to d.
- Composition with the identity does not change the morphism: for any $f\colon a \longrightarrow b$ the composition with id_a is f, i.e., $f \circ id_a = f$, and composition with id_b is also f, i.e., $id_b \circ f = f$.

Basic Examples

We will go through many examples. One might feel overwhelmed by all the examples. You do not have to "get it" at the first reading. The point is that all of the examples have the same feel to them. There are objects, there are morphisms, and they have to satisfy certain properties. Press on!

Example 2.1.3. Let us mention three examples of categories that we already saw in this text. Although we did not call them categories, the text and exercises showed that they each have the structure of a category.

- Sets and functions form the category $\mathbb{Set}$.
- Directed graphs and graph homomorphisms give us $\mathbb{Graph}$.
- Groups and group homomorphisms make up $\mathbb{Group}$.

□

The definition of a category is a "mouthful" that has many parts to it. There are several important comments concerning this definition.

- We called the elements of $Mor(\mathbb{A})$ the "morphisms" of the category. We will also interchangeably use the words **maps** and **arrows**.
- $Ob(\mathbb{A})$ and $Mor(\mathbb{A})$ are called "collections" rather than the more set-theoretical "sets" or "classes". The reason for this is because we do not want to get bogged down in the language of set theory. If you know the language of set theory, then realize that sometimes our objects and morphisms will be sets and sometimes proper classes (classes that are not sets). Often we will not specify which and just use the word "collection."
- It is important to notice that, if we have morphisms $f\colon a \longrightarrow b$ and $g\colon b \longrightarrow c$, then we write the composition as $g \circ f$ rather than $f \circ g$. We do this because in many categories the morphisms will be types of functions. When we apply the composition of functions, it looks like $g(f(\ \ ))$ which is notationally closer to $g \circ f$ than $f \circ g$. As we get more and more used to the language we will write gf rather than $g \circ f$.
- Another way of seeing the definition of a category is to discuss collections of morphisms. In the previous chapter we saw that for sets S and T we can look at the collection of all set functions from S to T. Now we look at the set of all morphisms between two objects. For objects a and b in category $\mathbb{A}$, there is a collection of *all* the morphisms from a to b which we write $Hom_{\mathbb{A}}(a, b)$. The name "*Hom*" comes from the word "homomorphism" which is a vestige of the algebraic background of category theory. We call these collections **Hom sets** even though the collections might not be sets. Composition in the category in terms of the Hom sets becomes the operation

$$\circ\colon Hom_{\mathbb{A}}(b,c) \times Hom_{\mathbb{A}}(a,b) \longrightarrow Hom_{\mathbb{A}}(a,c) \qquad (2.6)$$

$$(g, f) \longmapsto g \circ f.$$

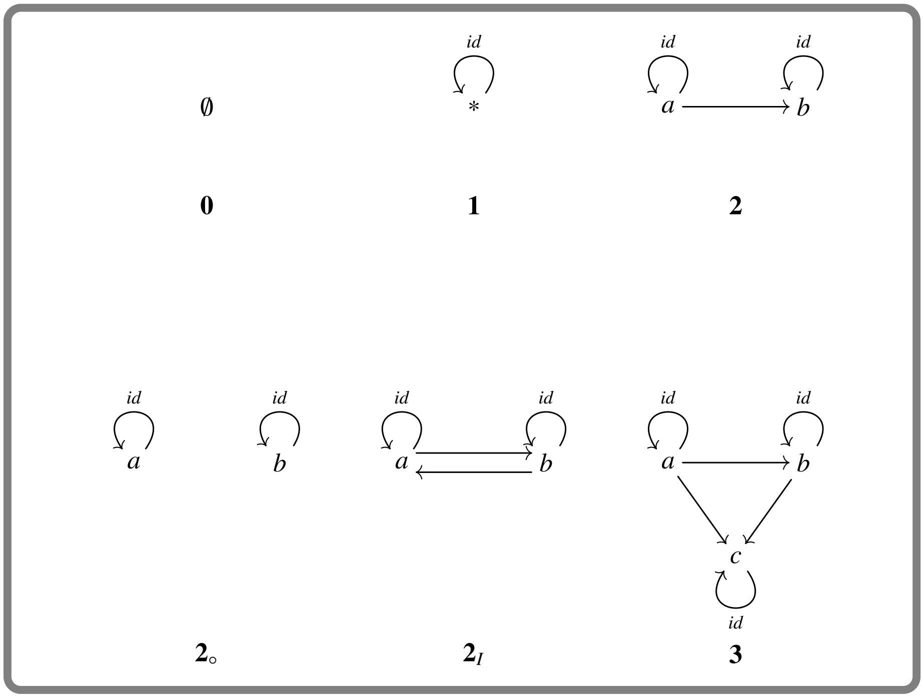


Figure 2.1: Several finite categories.

This means that we take an $f\colon a \longrightarrow b$ and a $g\colon b \longrightarrow c$, and return $g \circ f\colon a \longrightarrow c$.
We refer to mappings between Hom sets as functions even though the Hom sets might be a proper class. The fact that every element a of $\mathbb{A}$ has an identity element means that there is a special morphism in $Hom_{\mathbb{A}}(a,a)$ that satisfies the properties stated. As time goes on, and it is obvious what category we mean, we will drop the subscript and write $Hom(a,b)$.

The categories $\mathbb{Set}$, $\mathbb{Graph}$, and $\mathbb{Group}$ each have collections of objects and morphisms that are infinite. Let us, however, look at some examples of categories with finite collections of objects and morphisms.

Example 2.1.4. These **finite categories** are depicted in Figure 2.1. In detail,

- **0**, the empty category, has no objects and no morphisms. All the axioms of being a category are trivially true.
- **1** has one object and the single identity morphism on that object.
- **2** has two objects and three morphisms. Two of the morphisms are identity morphisms on the two objects and the third morphism goes from one object to the other.
- **2**$_\circ$ has the two objects and the two identity maps but does not have the non-

identity morphism.

- $\mathbf{2}_I$ is like **2** but there are two non-identity morphisms, and their compositions are the identity morphisms. In total, it has two objects and four morphisms.
- **3** is a category with three objects, three identity morphisms, and three non-identity morphisms. The non-identity morphisms form a commutative triangle.

Although these categories may seem trivial, they will be very useful. They provide easy examples to explore concepts and have important roles as we explore category theory. □

Example 2.1.5. Not only does the collection of all sets form a category, but each individual set can also be thought of as having the structure of a category. Let S be a set (it can be finite, infinite, or even a proper class, if you understand the jargon of set theory.) We form the category $d(S)$ where the objects are the elements of S, and the only morphisms are identity morphisms. The composition operation can only compose identity maps with themselves. We call a category with only identity morphisms a **discrete category**. For example the set $S = \{a, b, c, d\}$ becomes the category:

$$\begin{tikzcd} a \arrow[loop above, "id_a"] & b \arrow[loop above, "id_b"] \\ c \arrow[loop above, "id_c"] & d. \arrow[loop above, "id_d"] \end{tikzcd} \tag{2.7}$$

□

Let us go through more examples of categories.

Example from Computers

Here is an example of a category for someone who appreciates computer science.

Example 2.1.6. The category of computable functions $\mathbb{CompFunc}$ is central for computer science. A function is **computable** if there exists a computer program that can tell a computer how to describe the function. That means, there is a computer program (written in some programming language) and if $f(x) = y$ then when x is entered into the computer as input, the program will output y. Computable functions take certain forms of data as input and return certain forms of data as output. The kind of input and output is called a **type**. A type is a class of data. Computers deal with types like *Nat* (natural numbers), *Int* (integers), *Real*, *Bool* (Boolean), *String*, etc. The objects of $\mathbb{CompFunc}$ are sequences (or products) of types. For example, $Int \times Bool \times Bool \times Real$. Given two sequences of types, a morphism of this category will be a computable function from the first sequence of types to the second sequence of types. A typical computable function might look like

$$f\colon Int \times String \times Bool \longrightarrow Bool \times Real \times Real \times Nat. \tag{2.8}$$