Discrete Mathematics

Lecture Notes

on

Sets and Functions

M. Andrew Moshier

October 2014

Version: 0:0.0 (January 2, 2015) Format: print

Part I

Natural Numbers and Induction

Overview

The natural numbers constitute the fundamental structure of discrete mathematics. Moreover, they are the subject of some of your earliest mathematical experiences, when you first learned to count. In this part, we look carefully at how the natural numbers capture our basic intuitions about counting, how we build arithemetic from counting, and how we prove arithmetic laws using the basic structure of counting.

Lists also constitute a similar fundamental structure, whereby we can put things in a specific order. We investigate how lists are similar to natural numbers and develop some of the ways we will use lists later.

Lecture 1

The Natural Numbers

Goals

Lecture

- Present the natural numbers as comprising a structure suited to counting.
- Identify similar structures that can not properly represent counting.
- Rule out “bad” structures via postulates.

Study

- Gain facility in the course’s *successor* notation, including translating between successor notation and base 10 notation.
- Commit to memory the axioms of natural numbers.
- Demonstrate ability to recognize failures of the axioms.

The *natural numbers* have to do with counting: $0, 1, 2, 3, \ldots$. They do not include negatives or fractions or irrationals. In this lecture, the structure of natural numbers is the topic. To hone in on that structure, we look at structures similar to the natural numbers, but that fail to capture some basic aspects of counting. Bogus structures are ruled out by *postulates* (also known as *axioms*) that distinguish the structure of natural numbers from others.

1.1 The Basic Picture

Natural numbers are pictured like stepping stones in Figure 1.1.

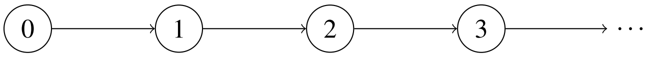


Figure 1.1: A picture of the natural numbers

Not all “stepping stone” pictures are acceptable. Figures 1.2, 1.3 and 1.4 illustrate three ways *not* to picture the natural numbers.

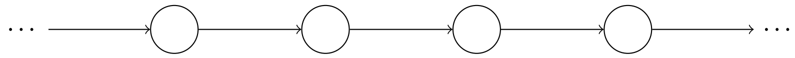


Figure 1.2: Nowhere to start

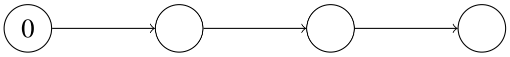


Figure 1.3: Nowhere to go

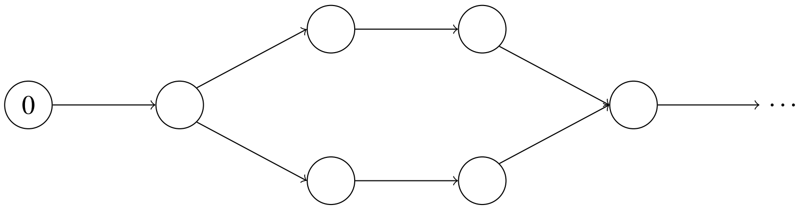


Figure 1.4: Forks in the path

These incorrect pictures can be ruled out by explaining the basic structure of counting.

Postulate 1.1

[Basic Structure] The natural numbers have the following basic structure.

- There is a special natural number. We denote this by 0.
- For any natural number n, there is a unique *next* natural number. We call this the *successor of* n. In these lectures, we denote the successor of n by $n^{\frown}$.

Exercises for Lecture 1

Convert the following to successor notation.

1. 9
2. 10
3. $4+3$
4. $n+4$

Convert the following to base 10 notation.

1. $0^{\frown\frown\frown\frown}$
2. $n^{\frown\frown\frown\frown\frown}$
3. $5^{\frown\frown}$
4. $0^{\frown}+0^{\frown\frown}$

1.2 Narrowing the possibilities

Figures 1.5 and 1.6 illustrate problems that Postulate **??** does not avoid.

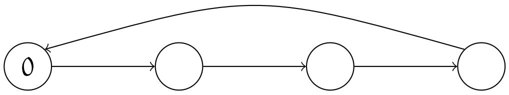


Figure 1.5: A strange way to count

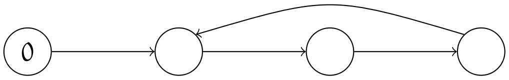


Figure 1.6: Another strange way to count

Exercises for Lecture 1

1. Explain, in one or two sentences each, why Figures 1.5 and 1.6 depict systems that agree with the Basic Vocabulary.

Figure 1.5 is flawed because 0 has a *predecessor*: a value n satisfying $0^{\curvearrowright\curvearrowright\curvearrowright\curvearrowright} = 0$. Figure 1.6 is flawed because an element has two distinct predecessors: $0^{\curvearrowright} = 0^{\curvearrowright\curvearrowright\curvearrowright\curvearrowright}$. We can insist that these flaws do not happen in the natural numbers. That is, we rule them out with axioms.

Postulate 1.2

[Nothing Precedes 0] For every natural number n, $n^{\curvearrowright} \neq 0$.

Postulate 1.3

[Predecessors are Unique] For any natural numbers m and n, if $m^{\curvearrowright} = n^{\curvearrowright}$ then $m = n$.

These postulates eliminate Figures 1.5, 1.6 and similar pictures. But there is still a subtle problem. Consider Figure 1.7.

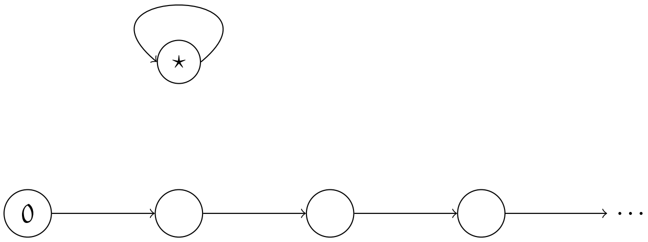


Figure 1.7: A model of the natural numbers?

This picture satisfies the first three postulates. Yet, it is not a picture of natural numbers because it has "extra stuff" in it ($\star$).

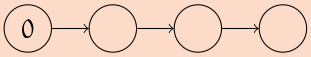
To rule out "extra stuff", we formulate our final postulate for natural numbers. We diagnose the problem as follows. Were we to erase the circle labelled $\star$ and any the arrows leading to and from it, the remaining part of Figure 1.7 would still live up to Postulate **??**. This is exactly what we mean by "extra stuff": elements that can be removed without violating the Postulate **??** (the essential structure). This leads to our last axiom.

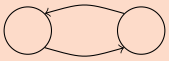
Postulate 1.4

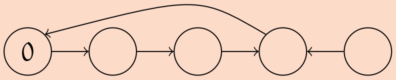
[The Axiom of Induction] No natural numbers can be removed without violating **??**.

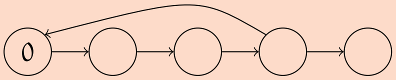
Exercises for Lecture 1

1. Each of the following pictures fails to satisfy either the one or more of our axioms. For each, explain which axioms are violated.

1. 

2. 

3. 

4. 

2. I have in mind a picture for the Basic Vocabulary **??** and that satisfies Axioms 1.2 and 1.3. Furthermore, in that picture, I have in mind and element n for which (a) $n \neq 0$ and (b) n has no predecessor (that is, $n \neq m^{\curvearrowright}$ for every m). Convince me that the picture fails to satisfy Axiom 1.4.

The latest exercise shows that in the natural numbers, if $n \neq 0$, then $n = m^{\curvearrowright}$ for some m. In other words, every non-zero natural number has a predecessor.

Lecture 2

Arithmetic

Goals

Lecture

- Present addition and multiplication via defining equations.
- Practice using the defining equations to calculate sums and products.

Study

- Understand addition and multiplication as characterized by defining equations.
- Be able to explain how addition and multiplication relate to counting.
- Exhibit competence in calculating sums and products from the defining equations.

Adding and multiplying arise from counting. In this section, we explore how to define them purely in terms of counting.

2.1 Basic Arithmetic Operations

Definition 2.1

[Arithmetic Operations] The *sum* of two natural numbers, m and n, is a natural number (denoted by $m + n$). For every natural number m, the following are true:

$$\begin{aligned} m + 0 &= m \\ m + k^{\frown} &= (m + k)^{\frown} \qquad \text{for any natural number } k \end{aligned}$$

The *product* of two natural numbers, m and n, is a natural number (denoted by $m \cdot n$). For every natural number m, the following are true:

$$\begin{aligned} m \cdot 0 &= 0 \\ m \cdot k^{\frown} &= m + (m \cdot k) \qquad \text{for any natural number } k \end{aligned}$$

A moment's thought about arithmetic should convince you that these equations are reasonable. Certainly $m + 0 = m$ and $m \cdot 0 = 0$ should be true for any m. The second equation for $+$ can be read as saying "to add m to the successor of k, simply add m to k, then take the successor." The second equation for $\cdot$ can be read as saying "to multiply m by the successor of k, simply multiply m by k, and add m to the result."

The Axiom of Induction ensures that there are indeed unique operations $+$ and $\cdot$ that satisfy the equations. A proof of this fact is not particularly illuminating right now, so let us agree to take it for granted.

Example 2.1

Do the defining equations for addition really explain how to add? Let's use them to calculate $4 + 3$:

$$\begin{aligned}
4 + 3 &= 4 + 0^{\curvearrowright\curvearrowright\curvearrowright} && \text{[3 abbreviates } 0^{\curvearrowright\curvearrowright\curvearrowright}\text{]}\\
&= 4^{\curvearrowright} + 0^{\curvearrowright\curvearrowright} && [m + k^{\curvearrowright} = m^{\curvearrowright} + k]\\
&= 4^{\curvearrowright\curvearrowright} + 0^{\curvearrowright} && \text{[Same reason]}\\
&= 4^{\curvearrowright\curvearrowright\curvearrowright} + 0 && \text{[Same reason]}\\
&= 4^{\curvearrowright\curvearrowright\curvearrowright} && [m + 0 = m]\\
&= (0^{\curvearrowright\curvearrowright\curvearrowright\curvearrowright})^{\curvearrowright\curvearrowright\curvearrowright} && \text{[4 abbreviates } 0^{\curvearrowright\curvearrowright\curvearrowright\curvearrowright}\text{]}\\
&= 0^{\curvearrowright\curvearrowright\curvearrowright\curvearrowright\curvearrowright\curvearrowright\curvearrowright} && \text{[Remove unneeded parentheses]}\\
&= 7 && \text{[7 abbreviates } 0^{\curvearrowright\curvearrowright\curvearrowright\curvearrowright\curvearrowright\curvearrowright\curvearrowright}\text{]}
\end{aligned}$$

Example 2.2

A product can be calculated similarly. Consider $2 \cdot 2$.

$$\begin{aligned}
2 \cdot 2 &= 2 \cdot 0^{\curvearrowright\curvearrowright} && \text{[2 abbreviates } 0^{\curvearrowright\curvearrowright}\text{]}\\
&= 2 + (2 \cdot 0^{\curvearrowright}) && [m \cdot k^{\curvearrowright} = m + (m \cdot k)]\\
&= 2 + (2 + (2 \cdot 0)) && \text{[Same reason]}\\
&= 2 + (2 + 0) && [m \cdot 0 = 0]\\
&= 2 + 2 && [m + 0 = m]\\
&= 2 + 0^{\curvearrowright\curvearrowright} && \text{[2 abbreviates } 0^{\curvearrowright\curvearrowright}\text{]}\\
&= 2^{\curvearrowright} + 0^{\curvearrowright} && [m + k^{\curvearrowright} = m^{\curvearrowright} + k]\\
&= 2^{\curvearrowright\curvearrowright} + 0 && \text{[Same reason]}\\
&= 2^{\curvearrowright\curvearrowright} && [m + 0 = m]\\
&= (0^{\curvearrowright\curvearrowright})^{\curvearrowright\curvearrowright} && \text{[2 abbreviates } 0^{\curvearrowright\curvearrowright}\text{]}\\
&= 0^{\curvearrowright\curvearrowright\curvearrowright\curvearrowright} && \text{[Remove unnecessary parentheses]}\\
&= 4 && \text{[4 abbreviates } 0^{\curvearrowright\curvearrowright\curvearrowright\curvearrowright}\text{]}
\end{aligned}$$

We certainly will not want to calculate this way in real life. After all, it took twelve steps just to figure $2 \cdot 2 = 4$. But these examples and the following exercises show how addition and multiplication are closely tied to simple counting.

Exercises for Lecture 2

1. Calculate these sums, following the previous example to write each step of your calculation explicitly. Include the reason for each step (as in the previous example).Take care to lay out the chain of equalities correctly, and do not skip any steps.

    1. $2 + 4$
    2. $4 + 2$
    3. $3 + (3 + 1)$
    4. $(3 + 3) + 1$
    5. $0 + 3$

2. Notice that it takes more steps to calculate $2 + 4$ than $4 + 2$, even though we know they will produce the same answer. Explain why.

3. Calculate the following values, writing each step explicity.

    1. $2 \cdot 3$
    2. $0 \cdot 2$
    3. $2 \cdot (2 \cdot 2)$
    4. $3 \cdot (2 + 1)$
    5. $(3 \cdot 2) + (3 \cdot 1)$

4. Write a definition of exponentiation via defining equations. Follow the pattern of definition I have written for addition and multiplication.

Lecture 3

Laws of Arithmetic

Goals

Lecture

- Present the most common Laws of Arithmetic for natural numbers.
- Explain the method of *proof by simple induction*
- Prove a representative sample of the laws by simple induction.

Study

- Become familiar with the common names for the Laws of Arithmetic.
- Pay particular attention to the Laws of Positivity and Cancellativity (they may be the least familiar to you).
- Demonstrate the ability to identify the main parts of a proof by simple induction.
- Demonstrate the ability to construct the parts of a proof by simple induction.
- Prove the remaining laws for yourself.

Before working the last exercises, you knew that $3 \cdot (2+1)$ and $3 \cdot 2 + 3 \cdot 1$ would come out the same because of a law of arithmetic known as *distributivity*. Addition and multiplication satisfy several other laws.

3.1 Basic Laws

The following list summarizes several useful laws of arithmetic on the natural numbers. They are organized to emphasize similarities between addition and multiplication.

Laws

For any natural numbers, m, n and p:

Associativity	$m+(n+p)=(m+n)+p$ $m\cdot(n\cdot p)=(m\cdot n)\cdot p$	**Commutativity**	$m+n=n+m$ $m\cdot n=n\cdot m$

Identity	$m+0=m$
	$m\cdot 1=m$
		Positivity	if $m+n=0$ then $m=0$
			if $m\cdot n=1$ then $m=1$

Cancellativity	if $m+p=n+p$ then $m=n$
	if $m\cdot p^{\frown}=n\cdot p^{\frown}$ then $m=n$		

Distributivity	$m\cdot(n+p)=(m\cdot n)+(m\cdot p)$		
Case Distinction	if $m\neq 0$ then $m=k^{\frown}$ for some k		

Most of these laws are familiar and are listed with their common names. The Law of Case Distinction was the subject of Lecture 8 Exercise 2.. *Go back and look at that exercise again.* The Law of Positivity for multiplication is not a common name, but I have used it to emphasize the analogies between addition and multiplication. Also Case Distinction does not really have a common name. I made that up.

3.2 Inductive Proofs

Suppose we wish to prove that every natural number has some property. For example, let us suppose we wish to prove that every natural number is *mimsy*. I have no idea what a mimsy number is, but let us try to prove this anyway. We could try proving that 0 is mimsy, 1 is mimsy, 2 is mimsy, and so on. But this won't work because our proof will never end. In fact, it is not so obvious that we, humans with finite minds, can ever prove that some property is true for *all* natural numbers, since it seems to involve checking infinitely many individual cases.

The Axiom of Induction provides a way forward in spite of our limitations. Suppose we were to show that the mimsy natural numbers all by themselves constitute a picture of Signature **??**. Then there could not be any natural numbers left out, for otherwise, we could erase all the non-mimsy natural numbers and still have a picture of **??**. This is exactly what the Axiom of Induction forbids: we can not erase *anything* without breaking the signature.

So to prove that all natural numbers are mimsy, we simply need to prove that

- 0 is mimsy, and
- for all natural numbers k, if k is mimsy so is $k^{\frown}$.

From these, we conclude that the mimsy natural numbers by themselves form a picture of **??**. So the Axiom of Induction ensures that all natural numbers are mimsy.

To make inductive proofs easier to understand, we often write them using a three step outline, as illustrated here.

- [Basis] Prove that 0 is mimsy.
- [Inductive Hypothesis] Assume that k is mimsy.
- [Inductive Step] Prove that $k^{\frown}$ is mimsy. [You may use the assumption that k is mimsy in this part of the proof.]

More practical examples are next.

Proposition 3.1

Addition is associative.

Proof: We need to show that $m + (n + p) = (m + n) + p$ for all m, n and p. Let us suppose that m and n are fixed values (not known to us). We now prove that the values p for which $m + (n + p) = (m + n) + p$ holds form a picture of **??**.

- [Basis] $m + (n + 0) = m + n = (m + n) + 0$. Both steps are due to the defining equations of $+$.
- [Inductive Hypothesis] Assume $m + (n + k) = (m + n) + k$.
- [Inductive Step] We must show that $m + (n + k^{\curvearrowright}) = (m + n) + k^{\curvearrowright}$.

$$\begin{aligned} m + (n + k^{\curvearrowright}) &= m + (n + k)^{\curvearrowright} && \text{[Def. of +]} \\ &= (m + (n + k))^{\curvearrowright} && \text{[Same]} \\ &= ((m + n) + k)^{\curvearrowright} && \text{[Inductive Hypothesis]} \\ &= (m + n) + k^{\curvearrowright} && \text{[Def. of +]} \end{aligned}$$

Therefore (by the Axiom of Induction), $m + (n + p) = (m + n) + p$ holds for all p. Since the argument does not depend on any extra assumptions about m and n, it holds for all m and n. □

We say this proof is *by induction on* p to emphasize that the variable p is the focus of attention. The other variables are not directly involved in the structure of the proof.

The remainder of this section further illustrates the technique of simple arithmetic induction via proofs of other laws of arithmetic.

Proposition 3.2

0 is the identity for addition.

Proof: We must prove that $m + 0 = m = 0 + m$ for all m. The first equality is true by the definition of $+$. But the second equality, $m = 0 + m$, is not explicitly one of the defining facts about $+$. So we proceed by induction on m.

- [Basis] $0 + 0 = 0$ is true by definition of $+$.
- [Inductive Hypothesis] Assume $0 + k = k$.
- [Inductive Step] We must show that $0 + k^{\curvearrowright} = k^{\curvearrowright}$.

$$\begin{aligned} 0 + k^{\curvearrowright} &= (0 + k)^{\curvearrowright} && \text{[Def. of +]} \\ &= k^{\curvearrowright} && \text{[Inductive hypothesis]} \end{aligned}$$

Therefore, $0 + m = m$ holds for all m. □

To prove that addition is commutative, we need an additional fact about how successor and addition interact.

Lemma 3.1

For any m and n, $(m + n)^{\curvearrowright} = m^{\curvearrowright} + n$.

Proof: By induction on n:

- [Basis]

$$\begin{aligned}(m+0)^{\curvearrowright} &= m^{\curvearrowright} && \text{[Def. of +]}\\ &= m^{\curvearrowright}+0 && \text{[Def. of +]}\end{aligned}$$

- [Inductive Hypothesis] Assume $(m+k)^{\curvearrowright} = m^{\curvearrowright} + k$ for some k.
- [Inductive Step] We must show that $(m+k^{\curvearrowright})^{\curvearrowright} = m^{\curvearrowright} + k^{\curvearrowright}$.

$$\begin{aligned}(m+k^{\curvearrowright})^{\curvearrowright} &= ((m+k)^{\curvearrowright})^{\curvearrowright} && \text{[Def. of +]}\\ &= (m^{\curvearrowright}+k)^{\curvearrowright} && \text{[Inductive Hypothesis]}\\ &= m^{\curvearrowright}+k^{\curvearrowright} && \text{[Def. of +]}\end{aligned}$$

So $(m+n)^{\curvearrowright} = m^{\curvearrowright} + n$. Because the proof does not depend on any assumption about m, it is valid for all m. □

Proposition 3.3

Addition is commutative.

Proof: We need to show that $m+n = n+m$ for all m and n. This time, the proof is by induction on m. Fix a value for n.

- [Basis] $0+n = n = n+0$ holds because of Fact 3.2 and the definition of $+$.
- [Inductive Hypothesis] Assume that $k+n = n+k$ for some k.
- [Inductive Step] We must show that $k^{\curvearrowright} + n = n + k^{\curvearrowright}$.

$$\begin{aligned}k^{\curvearrowright}+n &= (k+n)^{\curvearrowright} && \text{[Lemma 3.1]}\\ &= (n+k)^{\curvearrowright} && \text{[Inductive Hypothesis]}\\ &= n+k^{\curvearrowright} && \text{[Def. of +]}\end{aligned}$$

Therefore, $m+n = n+m$ for all m. Since this argument does not depend on any assumptions about n, it is valid for all n. □

Proposition 3.4

Addition is cancellative.

Proof: We need to prove that if $m+p = n+p$, then $m = n$. This proof is a little subtler than the previous ones. But notice that is still follows the same form.

The proof is by induction on p. Assume that m and n are some fixed natural numbers.

- [Basis] Suppose $m+0 = n+0$. Then immediately by definition of $+$, $m = n$.
- [Inductive Hypothesis] Assume that the following statement is true for some k: if $m+k = n+k$ then $m = n$.

- [Inductive Step] We must show that if $m + k^{\curvearrowright} = n + k^{\curvearrowright}$ then $m = n$. Suppose $m + k^{\curvearrowright} = n + k^{\curvearrowright}$ [call this (*) for reference]. Then

$$\begin{aligned} (m+k)^{\curvearrowright} &= m + k^{\curvearrowright} && \text{[Def. of $+$]} \\ &= n + k^{\curvearrowright} && \text{[By the supposition (*)]} \\ &= (n+k)^{\curvearrowright} && \text{[Definition of $+$]} \end{aligned}$$

  Hence, by Axiom 1.3 $m + k = n + k$. So by the Inductive Hypothesis, $m = n$.

Therefore, $m + p = n + p$ implies $m = n$ for all p. Since this argument does not depend on any assumptions regarding m and n, it is valid for all m and n. □

To prove that multiplication is commutative and cancellative, we will need the following technical facts (analogous to Lemmas 3.2 and 3.1).

Lemma 3.2

$0 \cdot n = 0$

Proof: The proof is by induction on n.

- [Basis] $0 \cdot 0 = 0$ by definition of $\cdot$.
- [Inductive Hypothesis] Assume that $0 \cdot k = 0$ for some k.
- [Inductive Step] We must show that $0 \cdot k^{\curvearrowright} = 0$.

$$\begin{aligned} 0 \cdot k^{\curvearrowright} &= 0 + 0 \cdot k && \text{[Definition of $\cdot$]} \\ &= 0 + 0 && \text{[Inductive Hypothesis]} \\ &= 0 && \text{[Definition of $+$]} \end{aligned}$$

□

Lemma 3.3

$m^{\curvearrowright} \cdot n = m \cdot n + n$

Proof: The proof is by induction on n.

- [Basis] $m^{\curvearrowright} \cdot 0 = 0 = 0 + 0 = m \cdot 0 + 0$ all follow from the definitions of $+$ and $\cdot$.
- [Inductive Hypothesis] Assume that $m^{\curvearrowright} \cdot k = m \cdot k + k$ for some k.
- [Inductive Step] We must show that $m^{\curvearrowright} \cdot k^{\curvearrowright} = m \cdot k^{\curvearrowright} + k^{\curvearrowright}$.

$$\begin{aligned}
m^{\curvearrowright} \cdot k^{\curvearrowright} &= m^{\curvearrowright} + m^{\curvearrowright} \cdot k && \text{[Def. of $\cdot$]} \\
&= (m + m^{\curvearrowright} \cdot k)^{\curvearrowright} && \text{[Lemma 3.1]} \\
&= (m + (m \cdot k + k))^{\curvearrowright} && \text{[Inductive Hypothesis]} \\
&= ((m + m \cdot k) + k)^{\curvearrowright} && \text{[Associativity of $+$]} \\
&= (m \cdot k^{\curvearrowright} + k)^{\curvearrowright} && \text{[Def. of $\cdot$]} \\
&= m \cdot k^{\curvearrowright} + k^{\curvearrowright} && \text{[Def. of $+$]}
\end{aligned}$$

□

Other laws are left as exercises.

Exercises for Lecture 3

1. Prove that 1 is the identity for multiplication. That is $1 \cdot m = m = m \cdot 1$.
2. Prove that multiplication distributes over addition $[m \cdot (n + p) = m \cdot n + m \cdot p]$ by induction on p. You can use the fact that addition is associative and commutative because we have already proved those.
    1. Prove the basis: $m \cdot (n + 0) = m \cdot n + m \cdot 0$.
    2. Write the inductive hypothesis.
    3. Prove the inductive step: $m \cdot (n + k^{\curvearrowright}) = m \cdot n + m \cdot k^{\curvearrowright}$
3. Prove that multiplication is associative $[m \cdot (n \cdot p) = (m \cdot n) \cdot p]$ by induction on p.
    1. Prove the basis: $m \cdot (n \cdot 0) = (m \cdot n) \cdot 0$.
    2. Write the inductive hypothesis.
    3. Prove the Inductive Step: $m \cdot (n \cdot k^{\curvearrowright}) = (m \cdot n) \cdot k^{\curvearrowright}$. Hint: Use the Law of Distribution, which you just proved.
4. Prove that multiplication is commutative. Hint: Use the two facts we proved right before these exercises.

For the record, we also prove the cancellation law for multiplication. This is a bit harder than the exercises, but you should try to find your own proof before looking at the following.

Proposition 3.5

If $m \cdot p^{\curvearrowright} = n \cdot p^{\curvearrowright}$, *then* $m = n$.

Proof: The proof is by induction on n.

- [Basis] Suppose $m \cdot p^{\curvearrowright} = 0 \cdot p^{\curvearrowright}$. From Fact 3.2, $m + m \cdot p = m \cdot p^{\curvearrowright} = 0$. So the Law of Positivity for addition ensures that $m = 0$.
- [Inductive Hypothesis] Assume that for some k, the following is true: if $m \cdot p^{\curvearrowright} = k \cdot p^{\curvearrowright}$, then $m = k$.
- [Inductive Step] Suppose that $m \cdot p^{\curvearrowright} = k^{\curvearrowright} \cdot p^{\curvearrowright}$. Then

$$\begin{aligned} m \cdot p^{\curvearrowright} &= k^{\curvearrowright} \cdot p^{\curvearrowright} && \text{[By assumption]} \\ &= k \cdot p^{\curvearrowright} + p^{\curvearrowright} && \text{[Lemma 3.3]} \\ &= (k \cdot p^{\curvearrowright} + p)^{\curvearrowright} && \text{[Definition of } +\text{]} \\ &\neq 0 && \text{[Axiom Nat 1.2]} \end{aligned}$$

  Consequently, $m \neq 0$, for otherwise we would have $m \cdot p^{\curvearrowright} = 0$. Since m is not equal to 0, it is equal to some successor (by the Case Distinction Law). Let j be the predecessor of m, so that $j^{\curvearrowright} = m$. Then

$$\begin{aligned} j \cdot p^{\curvearrowright} + p^{\curvearrowright} &= m \cdot p^{\curvearrowright} && \text{[Lemma 3.3]} \\ &= k^{\curvearrowright} \cdot p^{\curvearrowright} && \text{[By supposition]} \\ &= k \cdot p^{\curvearrowright} + p^{\curvearrowright} && \text{[Lemma 3.3]} \end{aligned}$$

Because addition is cancellative, $j \cdot p^{\curvearrowright} = k \cdot p^{\curvearrowright}$. Now, the Inductive Hypothesis ensures that $j = k$. Hence $m = j^{\curvearrowright} = k^{\curvearrowright}$.

□

Lecture 4

Lists

Natural numbers constitute an important example of something more general, where objects are built up from simpler ones. The Axiom of Induction captures the idea of building "up" and provides an important method for proving facts about natural numbers.

In this lecture, we develop an analogous way to think about *lists*.

Goals

Lecture Goals

- Introduce a formal counterpart to the informal concept of a list
- Emphasize the close analogy between lists and natural numbers
- Introduce basic operations on lists.

Study Goals

- Demonstrate facility with basic list manipulation including calculating length and concatenation of lists.

4.1 Lists

In this section, we concentrate on the fundamental concept of *lists*. The idea is really meant to be the familiar one, so a list of "to do" items is a list. The alphabetized names on a class roster is a list. We will write lists using square brackets. So for example, $[2, 3, 5, 7]$ is the list of the prime numbers less than 10 in ascending order. For lists, we expect the order to matter. So $[7, 5, 3, 2]$ is a different list.

Something that occurs on a list is called an *item* of the list. We can even specify where it is. So we can talk about the "first", "second" item, and so on, assuming the list has enough items.

Because we have already agreed that natural numbers begin with 0, it turns out to make many things easier if we change the way we talk about items on a list to gibe with the natural numbers. So instead of refering to the "first" item, we might call it the "initial" item. Furthermore, we will number them to start with 0. What I mean is that if $L = [2, 3, 5, 7]$, we will write L_0, L_1, L_2, L_3 for the elements 2, 3, 5, 7, respectively. In short, the "initial" item is indexed by the "initial" natural number 0. The next item after that is indexed by next natural number, $0^\frown$, and so on.

Like natural numbers, lists can be built up by starting with an empty list and incrementally adding items. We have choices for how we might formalize the idea. We will follow a standard that has developed in computer science. Clearly, since we use square brackets to punctuate lists, the empty list should be written as $[\,]$. To add an item to a list, we will conventionally put it on the front.

Given the list $[x, y, z]$, we may build a new list with initial item w and the given list as the rest, resulting in $[w, x, y, z]$. The operation of *prepending* an item to a list is denoted by a colon (:). So $w : [x, y, z]$ *is* the list $[w, x, y, z]$.

The empty list, together with prepending items, gives us a way to construct any list we want.

Example 4.1

Here are some examples.

- $5 : 6 : [4, 5]$ is the same as $5 : [6, 4, 5]$, which is the same as $[5, 6, 4, 5]$.
- $[]$ is the empty list
- $1 : []$ is the same as $[1]$
- $1 : 2 : 3 : 4 : []$ is the same as $[1, 2, 3, 4]$.

Notice that every list is either empty ($[]$) or not. If not, it has the form $x : L$ where x is the initial item and L is the rest of the list. This suggests a signature for lists, not so different from the signature for natural numbers.

Postulate 4.1

[Basic Structure of Lists] Lists have the following basic structure.

- There is a special list, which we call *the empty list* and denote by $[]$.
- For any thing x and any list L, there is another list, obtained by *prepending* x to L. We denote the result by $x : L$.

As with the natural numbers, we need to think about axioms that prevent strange behavior. These are exactly analogous to the axioms of natural numbers. First, $[]$ can not be obtained by adding a new initial item to another list. So

Postulate 4.2

For any list L and any thing x, $[] \neq x : L$.

Likewise, a list that is not empty can only be built one way.

Postulate 4.3

For any things x and y and lists L and M, if $x : L = y : M$, then $x = y$ and $L = M$.

For example, if I tell you that $[2, 3, 4, 5] = x : L$, then you know immediately that $x = 2$ and $L = [3, 4, 5]$.

Finally, lists need an induction axiom that ensures that all lists are built up from $[]$.

Postulate 4.4

[The Axiom of List Induction] No lists can be removed without violating Postulate 4.1.

This axiom justifies conducting proofs about all lists by a scheme almost identical to simple arithmetic induction. That is, to prove some property is true about all lists, it is enough to show

- [Basis] The property is true about $[]$.
- [Inductive Hypothesis] Assume that the property is true for from list K.

- [Inductive Step] Prove that for any thing x, the property is true about $x : K$. [You may use the assumption about K in this part of the proof.]

Operations on lists can now also be defined by schemes similar to how we defined addition and multiplication on natural numbers. For example, every list has a length. Writing $\mathsf{len}(L)$ for the length of a list, $\mathsf{len}([2, 3, 4]) = 3$. A precise definition is now easy to formulate.

Definition 4.1

For a list L. the *length* of L, denoted by $\mathsf{len}(L)$, is the natural number. This satisfies the following equalities.

$$
\begin{aligned}
\mathsf{len}([\,]) &= 0\\
\mathsf{len}(x : L) &= len(L)^{\curvearrowright}
\end{aligned}
$$

Example 4.2

$$
\begin{aligned}
\mathsf{len}([2,3,4]) &= \mathsf{len}(2 : [3,4])\\
&= \mathsf{len}([3,4])^{\curvearrowright}\\
&= \mathsf{len}(3 : [4])^{\curvearrowright}\\
&= \mathsf{len}([4])^{\curvearrowright\curvearrowright}\\
&= \mathsf{len}(4 : [\,])^{\curvearrowright\curvearrowright}\\
&= \mathsf{len}(,])^{\curvearrowright\curvearrowright\curvearrowright}\\
&= 0^{\curvearrowright\curvearrowright\curvearrowright}\\
&= 3
\end{aligned}
$$

Another common operation on lists is *concatenation*: $[2, 3, 4] \otimes [4, 1, 3] = [2, 3, 4, 4, 1, 3]$, whereby the two lists are simply glued together in their original orders. This is defined precisely by the following.

Definition 4.2

For lists L and M, their *concatenation*, denoted by $L \otimes M$, is a list. For all lists M, the following are true.

$$
\begin{aligned}
[\,] \otimes M &= M\\
(x : K) \otimes M &= x : (K \otimes M) \qquad \text{for any thing } x \text{ and any list } K
\end{aligned}
$$

Example 4.3

To calculate $[4,5,2,1] \otimes [3,4,1]$, we can follow a method similar to arithmetic:

$$\begin{aligned}
[4,5,2,1] \otimes [3,4,1] &= (4:5:2:1:[\,]) \otimes [3,4,1] && [[4,5,2,1] \text{ abbreviates } 4:5:2:1:[\,]] \\
&= 4:((5:2:1:[\,]) \otimes [3,4,1]) && [\text{Def. of } \otimes] \\
&= 4:5:((2:1:[\,]) \otimes [3,4,1]) && [\text{Same}] \\
&= 4:5:2:((1:[\,]) \otimes [3,4,1]) && [\text{Same}] \\
&= 4:5:2:1:([\,] \otimes [3,4,1]) && [\text{Same}] \\
&= 4:5:2:1:[3,4,1] && [\text{Same}] \\
&= [4,5,2,1,3,4,1] && [\text{Abbreviation}]
\end{aligned}$$

Now we can prove some useful facts about lists.

Lemma 4.1

On lists, $[\,]$ is the identity for $\otimes$,

Proof: By definition $[\,] \otimes L = L$ always true. But $[\,]$ must also satisfy $L \otimes [\,] = L$ always. We can proceed by induction on L. The proof should look familiar (see the proof of Lemma 3.2).

- [Basis] $[\,] \otimes [\,] = [\,]$ is true by definition of $\otimes$.
- [Inductive Hypothesis] Assume $K \otimes [\,] = K$ for some list K.
- [Inductive Step] Suppose x is some thing. We need to show that $(x:K) \otimes [\,] = x:K$.

$$\begin{aligned}
(x:K) \otimes [\,] &= x:(K \otimes [\,]) && [\text{by definition of } \otimes] \\
&= x:K && [\text{by the Inductive Hypothesis}]
\end{aligned}$$

Thus (by the Axiom of List Induction), the lists for which $L \otimes [\,] = L$ constitute all lists. □

Lemma 4.2

On lists, $\otimes$ is associative.

Proof: We prove $L \otimes (M \otimes N) = (L \otimes M) \otimes N$ using induction on L. This should look familiar. It is almost identicial to the proofs that addition and multiplication are associative.

- [Basis] $[\,] \otimes (M \otimes N) = M \otimes N = ([\,] \otimes M) \otimes N$. Both steps are by the definition of $\otimes$.
- [Inductive hypothesis] Suppose $K \otimes (M \otimes N) = (K \otimes M) \otimes N$ for some particular list K.
- [Inductive step]

$$\begin{aligned}
(x:K) \otimes (M \otimes N) &= x:(K \otimes (M \otimes N)) && \text{Def. of } \otimes \\
&= x:((K \otimes M) \otimes N) && \text{Inductive Hypothesis} \\
&= (x:(K \otimes M)) \otimes N && \text{Def. of } \otimes \\
&= ((x:K) \otimes M) \otimes N && \text{Def. of } \otimes
\end{aligned}$$

So $L \otimes (M \otimes N) = (L \otimes M) \otimes N$ is true for all L. Since the proof does not depend on any special propertis of M and N (except that they are both lists), the result is true for all lists M and N. □

Here is another nice fact that we can prove by induction relating length to concatenation.

Lemma 4.3

For any lists L and M, $\mathsf{len}(L \otimes M) = \mathsf{len}(L) + \mathsf{len}(M)$.

Proof: [This claim is probably fairly obvious to you. Nevertheless, to illustrate the technique of list induction again, we prove it explicitly.]

- [Basis] $\mathsf{len}([\,]) + \mathsf{len}(M) = 0 + \mathsf{len}(M) = \mathsf{len}(M) = \mathsf{len}([\,] \otimes M)$. These are by definition of $\otimes$ and $+$.
- [Inductive Hypothesis] Suppose $\mathsf{len}(K \otimes M) = \mathsf{len}(K) + \mathsf{len}(M)$ holds for some particular list K.
- [Inductive Step]

$$\begin{aligned}
\mathsf{len}((x : K) \otimes M) &= \mathsf{len}(x : (K \otimes M)) && \text{Def. of } \otimes \\
&= \mathsf{len}(K \otimes M)^{\curvearrowright} && \text{Def. of } \mathsf{len} \\
&= (\mathsf{len}(K) + \mathsf{len}(M))^{\curvearrowright} && \text{Inductive Hypothesis} \\
&= \mathsf{len}(K)^{\curvearrowright} + \mathsf{len}(M) && \text{Lemma 3.1} \\
&= \mathsf{len}(x : K) + \mathsf{len}(M) && \text{Def. of } \mathsf{len}
\end{aligned}$$

□

Often we will use a list somewhat informally without all the punctuation. For example, we might say "Consider a list $a_0, a_1, \ldots, a_{n-1}$ of real numbers." If we do not intend to use the list itself for anything special, but only want to think about the numbers a_0 through a_n, then there is no need to be formal about it. Also, there is no harm in writing something like this: a_5, a_6, a_7, a_8, where the indices start at 5. The default is to start at 0, but that is merely a convention.

Lemma 4.4

$\otimes$ is cancellative on the left and on the right. That is,

- $L \otimes M = L \otimes N$ implies $M = N$; and
- $L \otimes N = M \otimes N$ implies $L = M$.

Proof: Exercise. □

4.2 List Itemization

In a list L, the items are in order. So we can refer to items by their position in the list. There are two standards in mathematics for doing this. Either we start counting from 1 or from 0. Although it may seem unintuitive at first to start from 0 (meaning that the "initial" item of a list is item number 0), this actually makes many calculations simpler. For that reason, most programming languages use this convention for a lists and arrays. So I will consistently start with 0.

The idea can be made precise as follows.

Definition 4.3

Suppose L is a list and $i < \mathsf{len}(L)$. Then L_i is an item on the list defined as follows.

$$
\begin{aligned}
[\,]_i & \text{is never defined because } 0 \not< \mathsf{len}([\,]) \\
(x : L)_0 &= x \\
(x : L)_{k\frown} &= L_k & \text{provided that } L_k \text{ is defined}
\end{aligned}
$$

This is a precise way of explaining that in a list, for example $L = [a, b, c, d, e]$, we can refer to an item by its *index*, so that $L_0 = a$, $L_1 = b$ and so on, up to $L_4 = e$. Notice that L_k is undefined if $k \geq \mathsf{len}(L)$.

Example 4.4

Suppose $L = [a, b, c, d, e]$. We can calculate L_3 explicitly step by step.

$$
\begin{aligned}
L_3 &= [a, b, c, d, e]_3 \\
&= (a : b : c : d : e : [\,])_{0\frown\frown\frown} \\
&= (b : c : d : e : [\,])_{0\frown\frown} \\
&= (c : d : e : [\,])_{0\frown} \\
&= (d : e : [\,])_0 \\
&= d
\end{aligned}
$$

Of course, this is just a very careful (you might even say fussy) way to find item number 3 in the list. In every day use, we humans would not do this. We would simply count forward from the beginning of the list.

Exercises for Lecture 4

Suppose $L = [3, 2, 3, 3, 5]$ and $M = [0, 1, 2, 3, 4, 5]$. Calculate the following explicitly step by step.

1. $\mathsf{len}(L)$
2. L_4
3. $(L \otimes M)_9$

4.3 Lists of a Particular Type

We will commonly need to consider lists in which all elements are similar, such as a list consisting of natural numbers. For example, because we know how arithmetic operations work on natural numbers, we can also define operations on lists of natural numbers using arithmetic. Similar extensions are possible for other operations defined on other types of elements.

To illustrate, suppose L is a list of natural numbers. We can define the *sum* of items on the list in the obvious way, so that the sum of the list $[2, 3, 4]$ is $2 + 3 + 4 = 9$. We make this precise with the following.

Definition 4.4

For a list L of natural numbers, the *sum of* L, denoted by $\sum \mathsf{L}$, is a natural number, satisfying

$$\begin{aligned} \sum [\,] &= 0 \\ \sum \mathsf{m} : \mathsf{L} &= \mathsf{m} + \sum \mathsf{L} \qquad \text{for any natural number } \mathsf{m} \text{ and any list of natural numbers } \mathsf{L} \end{aligned}$$

We will introduce variations and extensions of this notation this later. For now, we look only at lists.

Exercises for Lecture 4

Prove using list induction that for any lists of natural numbers,

$$\sum \mathsf{L} + \sum \mathsf{M} = \sum (\mathsf{L} \otimes \mathsf{M})$$

Define the product of lists of natural numbers, following the pattern of our definition for $\sum \mathsf{L}$. The standard notation for a product is $\prod \mathsf{L}$. The result should be that $\prod [2, 3, 4]$ equals 24. Pay close attention the base case $\prod [\,]$.

Using your definition of products, prove by list induction that for any lists of natural numbers,

$$\prod \mathsf{L} \cdot \prod \mathsf{M} = \prod (\mathsf{L} \otimes \mathsf{M})$$

.

We can also consider lists of integers, lists of real numbers, and so on. We can even think about lists of lists. For example, $[[2, 3, 4], [4, 3, 2], [5]]$ is a list consisting of two items: $[2, 3, 4]$, $[4, 3, 2]$ and $[5]$. Written this using :, this is list is $[2, 3, 4] : [4, 3, 2] : [5] : [\,]$. Suppose we have a list of lists like this we can define the concatenation of all the items. For this example, the result should be $[2, 3, 4, 4, 3, 2, 5]$. The definition of this is exactly analogous to sums and products.

Definition 4.5

For a list $\mathcal{L}$ of lists, the *fold of* $\mathcal{L}$, denoted by $\bigotimes \mathcal{L}$, is a list, satisfying

$$\begin{aligned} \bigotimes [\,] &= [\,] \\ \sum \mathsf{M} : \mathcal{L} &= \mathsf{M} \otimes \otimes \mathcal{L} \qquad \text{for any list } \mathsf{M} \text{ and any list of lists } \mathcal{L} \end{aligned}$$

Compare the definitions of $\sum$, $\prod$ and $\otimes$. They differ only in terms of (i) what is the result for an empty list and (ii) what binary operation is used in the second equation.

Suppose we are given a list $\mathcal{L}$ of lists of natural numbers (like the example just above the latest definition). Then its fold is a list of natural numbers. So this can be summed. That is, $\bigotimes \mathcal{L}$ is a list of natural numbers, and $\sum (\otimes \mathcal{L})$ is a natural number. But we might also apply the summation operation to each list on $\mathcal{L}$ separately, resulting in another list of natural numbers. The idea of applying an operation to each element of a list is called "mapping". In this case, we intend to "map" the operation $\sum$ across lists of lists of natural numbers. Here is a suitable definition.

Definition 4.6

For a list $\mathcal{L}$ of lists of natural numbers, the *mapping of* $\sum$ *on* $\mathcal{L}$, denoted by $\mathsf{map}_{\sum}(\mathcal{L})$ is a list of natural numbers, satisfying

$$\bigotimes[\,] = [\,]$$

$$\sum \mathrm{M} : \mathcal{L} = () \sum \mathrm{M}) : \mathcal{L} \quad \text{for any list of natural numbers } \mathrm{M} \text{ and any list of lists of natural numbers } \mathcal{L}$$

Exercises for Lecture 4

Calculate $\sum(\otimes[[3,4,5],[6,3]])$.

Calculate $\mathsf{map}_{\sum}([[3,4,5],[6,3]])$.

Calculate $\sum(\mathsf{map}_{\sum}([[3,4,5],[6,3]]))$.

Prove that $\sum(\otimes\mathcal{L}) = \sum(\mathsf{map}_{\sum}(\mathcal{L}))$ for any list of lists of natural numbers $\mathcal{L}$.

4.4 Other Inductively Defined Collections

The structure of natural numbers and the structure of lists are very similar. This similarity can be exploited to develop a simple way of summarizing their properties.

For natural numbers, 0 and $n^{\curvearrowright}$ are the only ways to construct them. Operations like addition and multiplication are defined in terms of 0 and ${}^{\curvearrowright}$, so they do not contribute directly to the *construction* of natural numbers. So we refer to 0 and $\curvearrowright$ as *constructors*.

Axioms 1.2 and 1.3 spell out how these constructors behave. Namely, Axiom 1.2 captures the idea that the two constructors are entirely different from the other: $0 \neq n^{\curvearrowright}$. Axiom 1.3 captures the idea that ${}^{\curvearrowright}$ constructs distinct natural numbers from distinct natural numbers: $m^{\curvearrowright} = n^{\curvearrowright}$ implies $m = n$ (or equivalently, $m \neq n$ implies $m^{\curvearrowright} \neq n^{\curvearrowright}$).

So the basic ingredients of natural numbers are the constructors 0 and ${}^{\curvearrowright}$ with the understanding that (a) each produces different results and (b) from different ingredients, ${}^{\curvearrowright}$ produces different results. In fact, point (b) also applies to 0 trivially, because 0 does not use any ingredients.

We can summarize everything we want to say about natural numbers concisely in the following way.

Definition 4.7

The *natural numbers* are defined *inductively* by

$$n \coloneqq 0 \mid n^{\curvearrowright}$$

In this notation, the vertical bar separates the different constructors for natural numbers. The first constructor (0) does not depend on anything else. The second constructor depends on a natural number n and produces a new one $n^{\curvearrowright}$. So this gives a very concise description of the signature of natural numbers. Implicitly, this notation is meant to indicate that the two alternatives are completely distinct. This is Axiom 1.2. Also implicitly, the notation is meant indicate that $n^{\curvearrowright}$ produces distinct results from distinct n's. This is Axiom 1.3. By declaring saying that this defines natural numbers *inductively*, we also mean that no natural numbers can be removed without violating the signature.

Now let's consider lists. Again, there are two ways to construct lists. $[\,]$ and $x : \mathrm{L}$ for any thing x and any list L. Likewise, the constructrs are distinct, and $x : \mathrm{L} = y : \mathrm{M}$ is true if and only if both $x = y$ and $\mathrm{L} = \mathrm{M}$. So we can encapsulate the definition of lists similarly.

Definition 4.8

The *lists* are defined *inductively* by

$$L \coloneqq [\,] \mid x : L \qquad\qquad \text{for any thing } x$$

Notice that x can also be a list.

Later in the course, we will make these definitions, and many others like them, rigorous. For now, we just draw attention to the similarity between natural numbers and lists, and point out that proofs by induction work thanks to the structure of these definitions.

4.5 Binary Trees

Simple binary trees are structures that play a role in many parts of computer science and mathematics. Figure 4.1 illustrates an example. There are many variations on the basic idea, but we concentrate on the simplest version (where there is no extra structure).

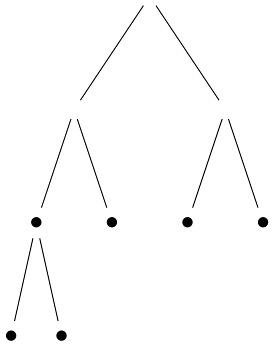


Figure 4.1: A simple binary tree

Such structures are built from *leaves* (denoted here by •) by "grafting" two smaller trees to form a larger one as pictured in Figure 4.2. Trees are usually depicted "upside down", so the *root* is at the top and the leaves are at the bottom. What can you do? It's tradition.

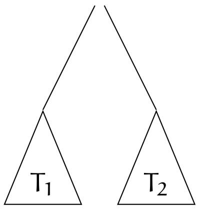


Figure 4.2: Constructing a tree from subtrees

In addition to drawing pictures of binary trees, we can use a linear notation (something that can we written in the midst of prose). If T_1 and T_2 are simple binary trees, we may denote the tree constructed by grafting T_1 on the left and T_2 on the right as $(T_1 \curlywedge T_2)$ (as in Figure 4.2. Thus we define simple binary trees, and two operations on them, as follows.

Definition 4.9

Simple binary trees are defined inductively by

$$\mathsf{T} \coloneqq \bullet \mid (\mathsf{T}_1 \curlywedge \mathsf{T}_2)$$

The *size* of a simple binary tree is a natural number defined by the equations

$$\begin{aligned} \mathsf{sz}(\bullet) &\coloneqq 0 \\ \mathsf{sz}(\mathsf{T}_1 \curlywedge \mathsf{T}_2) &\coloneqq 1 + \mathsf{sz}(\mathsf{T}_1) + \mathsf{sz}(\mathsf{T}_2) \end{aligned}$$

The *height* of a simply binary tree a natural number is defined by the equations

$$\begin{aligned} \mathsf{ht}(\bullet) &\coloneqq 0 \\ \mathsf{ht}(\mathsf{T}_1 \curlywedge \mathsf{T}_2) &\coloneqq 1 + \max(\mathsf{ht}(\mathsf{T}_1), \mathsf{ht}(\mathsf{T}_2)) \end{aligned}$$

where $\max(m, n)$ is the larger of the two numbers.

These definitions suggest a relation between the size and height of a tree.

Lemma 4.5

For any simple binary tree T,

$$\mathsf{ht}(\mathsf{T}) \leq \mathsf{sz}(\mathsf{T}) < 2^{\mathsf{ht}(\mathsf{T})}.$$

Proof: To prove this by *structural induction*, we must prove it for the basis (•) and that if it is true for some T_1 and T_2, then it is true for $(\mathsf{T}_1 \wedge \mathsf{T}_2)$.

- [Basis] $\mathsf{ht}(\bullet) = 0$, $\mathsf{sz}(\bullet) = 0$ and $2^{\mathsf{ht}(\bullet)} = 1$. So the claim is true for •.

- [Inductive Hypothesis] Suppose the inequalities holds for some T_1 and T_2.

- [Inductive Step] We must prove the two inequalities for $\mathsf{T} = (\mathsf{T}_1 \curlywedge \mathsf{T}_2)$. Without loss of generality, assume that $\mathsf{ht}(\mathsf{T}_1) \leq \mathsf{ht}(\mathsf{T}_2)$. That is, if this is not so, then we may swap T_1 for T_2 in the following.

$$\begin{aligned} \mathsf{ht}(\mathsf{T}) &= 1 + \max(\mathsf{ht}(\mathsf{T}_1), \mathsf{ht}(\mathsf{T}_2)) && \text{[Definition of ht]} \\ &\leq 1 + \mathsf{ht}(\mathsf{T}_1) + \mathsf{ht}(\mathsf{T}_2) && \text{[Arithmetic]} \\ &\leq 1 + \mathsf{sz}(\mathsf{T}_1) + \mathsf{sz}(\mathsf{T}_2) && \text{[Inductive Hypothesis]} \\ &= \mathsf{sz}(\mathsf{T}) && \text{[Definition of sz]} \end{aligned}$$

And

$$\begin{aligned} \mathsf{sz}(\mathsf{T}) &= 1 + \mathsf{sz}(\mathsf{T}_1) + \mathsf{sz}(\mathsf{T}_2) && \text{[Definition of sz]} \\ &\leq 1 + 2^{\mathsf{ht}(\mathsf{T}_1)} - 1) + (2^{\mathsf{ht}(\mathsf{T}_2)} - 1) && \text{[Inductive Hypothesis]} \\ &\leq 2 \cdot 2^{\mathsf{ht}(\mathsf{T}_2)} - 1 && \text{[Assumption that } \mathsf{ht}(\mathsf{T}_1) \leq \mathsf{ht}(\mathsf{T}_2)\text{]} \\ &= 2^{\mathsf{ht}(\mathsf{T}_2)+1} - 1 && \text{[Arithmetic]} \\ &= 2^{\mathsf{ht}(\mathsf{T})} - 1 && \text{[Definition of ht]} \end{aligned}$$

So $\mathsf{sz}(\mathsf{T}) < 2^{\mathsf{ht}(\mathsf{T})}$

□

Exercises for Lecture 4

Calculate the height and size of the following simple binary trees.

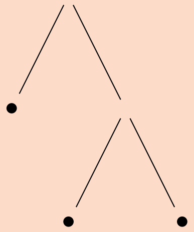
1. (• ⅄ •)
2. (• ⅄ (• ⅄ •))
3. ((• ⅄ •) ⅄ (• ⅄ (• ⅄ •)))
4. ((• ⅄ (• ⅄ •)) ∧ ((• ⅄ •) ⅄ (• ⅄ (• ⅄ •))))

Draw diagrams (similar to those in Figure 4.1) for the following simple binary trees.

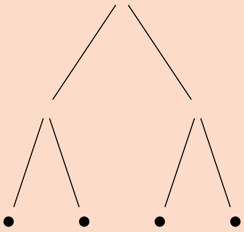
1. ((• ⅄ •) ⅄ (• ⅄ •))
2. (((• ⅄ •) ⅄ •) ⅄ ((• ⅄ •) ⅄ (• ⅄ (• ⅄ •))))

For each of the following diagrams, write the expression using • and ⅄ defining the same tree.

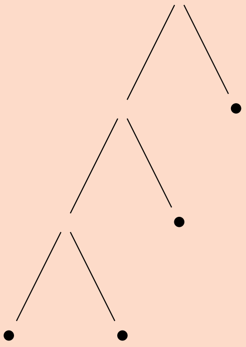
1.



2.



3.



We will study binary trees in more depth later in the course.

Part II

Basics of Sets and Functions

LECTURE 5

Sets

Goals

Lecture

- Describe informally the category of sets.
- Define list set notation.
- Introduce the idea of a subset.
- Introduce the axiom of extensionality for sets and some of its consequences.

Study

- Demonstrate ability to determine equality of sets.
- Develop facility in basic set theoretic notation.

Sets are the mathematician's way of thinking about *collections* of objects. Examples will be the set of natural numbers, the set of pairs of natural numbers, the set of real numbers, and so on.

An example is a set representing poker cards. We may denote it by

$$\begin{aligned}\mathsf{Deck} = \{&A\clubsuit, 2\clubsuit, 3\clubsuit, 4\clubsuit, 5\clubsuit, 6\clubsuit, 7\clubsuit, 8\clubsuit, 9\clubsuit, 10\clubsuit, J\clubsuit, Q\clubsuit, K\clubsuit,\\ &A\diamondsuit, 2\diamondsuit, 3\diamondsuit, 4\diamondsuit, 5\diamondsuit, 6\diamondsuit, 7\diamondsuit, 8\diamondsuit, 9\diamondsuit, 10\diamondsuit, J\diamondsuit, Q\diamondsuit, K\diamondsuit,\\ &A\spadesuit, 2\spadesuit, 3\spadesuit, 4\spadesuit, 5\spadesuit, 6\spadesuit, 7\spadesuit, 8\spadesuit, 9\spadesuit, 10\spadesuit, J\spadesuit, Q\spadesuit, K\spadesuit,\\ &A\heartsuit, 2\heartsuit, 3\heartsuit, 4\heartsuit, 5\heartsuit, 6\heartsuit, 7\heartsuit, 8\heartsuit, 9\heartsuit, 10\heartsuit, J\heartsuit, Q\heartsuit, K\heartsuit\}\end{aligned}$$

The elements are arranged here conveniently, but we could just as well have listed the cards in any "shuffled" order. The set of them would be the same.

Functions are the mathematicians way of thinking about attributes of the things in a collection, like "the color of", "the mass of", "the location of", "the father of", "the favorite book of the person to the left of" and so on. For our example of cards in a poker deck, "rank of" or "suit of" are two attributes. So we might write $\mathsf{rank}(A\diamondsuit) = A$ and $\mathsf{suit}(A\diamondsuit) = \diamondsuit$. In general, $\mathsf{rank}(x)$ and $\mathsf{suit}(x)$ pick out these attributes of a card x. The values these atrributes can take are also set $\mathsf{Rank} = \{A, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K\}$ and $\mathsf{Suit} = \{\clubsuit, \diamondsuit, \spadesuit, \heartsuit\}$.

The functions rank and suit capture some structure of the elements of Deck. For any rank r and any suit s, there is exactly one card c so that $\mathsf{rank}(c) = r$ and $\mathsf{suit}(c) = s$. For example, if $\mathsf{rank}(c) = 4$ and $\mathsf{suit}(c) = \spadesuit$, then we know exactly what c must be. So the two functions, in a sense, explain what a card is. We will use functions and sets to discuss more complicated structures, but the idea will be very similar to this simple example.

Taken together, sets and functions constitute a fundamental structure in contemporary mathematics called the *Category of Sets and Functions*. This is a slight lie. Actually, there are many different categories of sets and functions that differ in subtle ways. But for most mathematics, the differences are irrelevant. So in practice, it is safe to talk as if there is just one. The Category of Sets and Functions sometimes abbreviated as **Set**.

To understand sets and functions as they are used in every day mathematics, we need to answer some questions:

- What do we mean by saying that a set is a collection?
- What do we mean by saying that two sets are equal?
- What do we mean by saying that a function behaves like an attribute?
- What do we mean by saying that two functions are equal?
- How do we construct sets and functions?

The answers to these questions lead to some basic principles. We could be more formal and present these principles as *axioms*, but the word "axiom" has special connotation in mathematics that we do not need here. Nevertheless, everything we say in these lectures could be presented formally.

5.1 Set Basics

A set consists of things that are "in" the set. All other things are "not in" the set. In our running example, A♠ is in the set C, but 25 is not in C. Let us make the idea precise.

Principle 5.1

Basic Structure of Sets

A *set* is a mathematical entity X with the following feature. For any mathematical entity x, either x *is in* X or x *is not in* X. We write $x \in X$ if x is in X and $x \notin X$ if x is not in X.

The symbol $\in$ is used in mathematics exclusively to indicate membership in a set. You will not see it used in any other way.

For variety, all of the following phrases mean the same thing:

- x is in X
- x is an *element of* X
- x is a *member of* X
- X *contains* x
- x *belongs to* X

Principle 5.1 describes how we can talk about sets and elements, and how to use the notation of membership, but does not tell us that any sets actually exist. Our first remedy for this is to make room for finite sets.

Principle 5.2

Finite Sets

For any list $L = [a_0, \ldots, a_{n-1}]$, there is a set, denoted by $\{a_0, \ldots, a_{n-1}\}$, so that $x \in \{a_0, \ldots, a_{n-1}\}$ if and only if $x = a_i$ for some $i < n$. More precisely,

- $x \notin \{\}$ for any x ($\{\}$ is said to be *empty*);
- $x \in \{a_0, \ldots, a_n\}$ if and only if $x = a_0$ or $x \in \{a_1, \ldots, a_n\}$.

Example 5.1

Here are some examples of sets built from finite lists:

- $\{\}$ – an empty set;
- $\{1,2,5\}$ – a set consisting of three elements;
- $\{\{\}\}$ – a set consisting of one element, which is $\{\}$;
- $\{1,2,4,\{1,2\}\}$ – a set consisting of four elements, 1, 2, 4 and the set $\{1,2\}$.
- $\{4,5,\{\},[\,]\}$ – a set consisting of four elements. Note that the set $\{\}$ and the list $[\,]$ are not the same things.
- $\{1,2,3,4,3,2,1\}$ – a set consisting of four elements, listing an element twice is redundant.
- The sets Deck, Rank and Suit from the introduction.

The study of finite sets is surprisingly complex, and comprises a large part of the branch of mathematics called *combinatorics*. We will touch on the basics of combinatorics later in the course.

Various infinite sets of numbers also exist, but these follow from general principles we have not discussed yet. We do not try to justify anything for now. Instead, we introduce them informally along with the standard symbols we use to denote them.

Definition 5.1

The following sets are denoted by the special symbols:

$$\begin{aligned}
\mathbb{N} &= \text{the set of natural numbers}\\
\mathbb{Z} &= \text{the set of integers}\\
\mathbb{Q} &= \text{the set of rational numbers}\\
\mathbb{R} &= \text{the set of real numbers}\\
\mathbb{C} &= \text{the set of complex numbers}
\end{aligned}$$

Exercises for Lecture 5

1. Let $A = \{1,\{2,3\},4\}$. Determine which of the following assertions are true.
   1. $1 \in A$
   2. $2 \in A$
   3. $\{\} \in A$
   4. $\{2,3\} \in A$
   5. $A \in A$
2. In the following examples of sets with elements following a pattern, write an expression for the same set that makes the pattern clearer.
   1. $\{0,2,4,\ldots,100\}$
   2. $\{1,2,4,8,\ldots,256\}$

   3. $\{0, 1, 3, 6, 10, \ldots, 55\}$

3. $\bullet \in \mathsf{Suit}$ (the finite set defined above).

5.2 Subsets and Extensionality

A set is meant to be a collection: some things are in, some are not. That's all we can say. Unlike a list, a set has no "initial" element. For example, the set $\{1, 2, 3\}$ is the same as the set $\{2, 3, 1\}$, because both have the same elements. This is one important difference between lists and sets: $[1, 2, 3]$ and $[2, 3, 1]$ are *not* the same list because order matters in lists. To make this precise, we need to be clear about when sets are equal. To help, we introduce an important definition.

Definition 5.2

For sets X and Y, we say that X *is a subset of* Y provided that every element of X is an element of Y. We write this as $X \subseteq Y$, and say that X *is included in* Y. We may also write $Y \supseteq X$ to mean the same thing, and say that Y *is a superset of* B.

If X is *not* a subset of Y, we write $X \not\subseteq Y$. If $X \subseteq Y$ and $Y \not\subseteq X$, then X is called a *proper subset of* Y. To indicate that X is a *proper* subset of Y, we may write $X \subsetneq Y$. Some people write $X \subset Y$ for proper subsets, but we will never use that symbol.

To say $X \subseteq Y$ is exactly to say that for any x, if $x \in X$ then $x \in Y$. In plain English, we may translate it informally as "all Xs are Ys." For example, suppose P is the set of all professors, and H is the set of all human beings. Then $P \subseteq H$ is the (dubious) assertion that "all professors are human beings".

Example 5.2

Here are some examples and counter-examples of the subset relation.

- $\{1, 2, 3\} \subseteq \{0, 1, 2, 3\}$
- $\{\} \subseteq \{0\}$
- $X \subseteq X$ for any set X because, trivially, every element of X is an element of X
- $\{\} \subseteq X$ for any set X because every element of $\{\}$ (there are none) is an element of X
- $\{1, 2, 3\} \not\subseteq \{0, 2, 3\}$ because $1 \in \{1, 2, 3\}$ but $1 \notin \{0, 2, 3\}$
- $\{1, 2, 3\} \subseteq \{2, 3, 1\}$
- $\{\spadesuit\} \subseteq S$

Exercises for Lecture 5

For each of the following pairs of sets, determine whether or not the first is a subset of the second. Explain your answer.

1. $\{0, 1\}$ and $\{1, 0\}$
2. $\{a, b, c, d\}$ and $\{a, b, d, e, c\}$
3. $\{\}$ and $\{\{\}\}$
4. $\{0, 3, 6, 10\}$ and $\{10, 9, 8, 7, 6, 5, 4, 2, 1, 0\}$

We can summarize two useful properties of $\subseteq$ as follows.

- [Reflexivity] For any set X, $X \subseteq X$. We say $\subseteq$ is *reflexive*.
- [Transitivity] For any sets X, Y and Z, if $X \subseteq Y$ and $Y \subseteq Z$, then $X \subseteq Z$. We say $\subseteq$ is *transitive*.

Another familiar example of a reflexive, transitive relation is $\leq$ on the natural numbers. In fact there are many examples of reflexive, transitive relations throughout mathematics. The relation $\leq$ is also *anti-symmetric*, meaning that if $m \leq n$ and $n \leq m$ then $m = n$. Suppose $X \subseteq Y$ and $Y \subseteq X$. Then, by definition X and Y have exactly the same elements. By our understanding of sets as collections, X and Y must be equal. So we state this as another princple.

Principle 5.3

Set Extensionality For sets X and Y, if $X \subseteq Y$ and $Y \subseteq X$, then $X = Y$. In other words, $\subseteq$ is anti-symmetric.

Based on this, we can already establish a useful fact: there is exactly one empty set. To set the tone for what follows, we make this a formal claim.

Lemma 5.1

There is exactly one empty set.

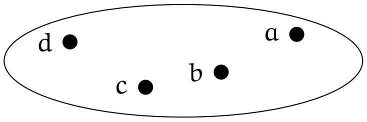
Proof: We have already noted that the set built from an empty list $\{\}$ has no elements. So there is at least one empty set.

Suppose E is a set with no elements. Then $E \subseteq \{\}$ because every element of E (there are none) is an element of $\{\}$. Similarly, $\{\} \subseteq E$ because every element of $\{\}$ (again, there are none) is an element of E. So by Principle 5.3 $E = \{\}$. □

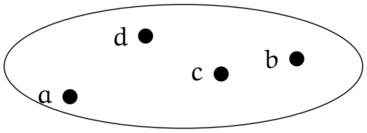
Definition 5.3

The set $\{\}$ is also denoted by $\emptyset$.

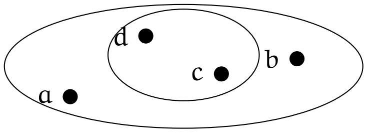
Set extensionality makes precise the idea that a set by itself does not have any structure other than what members it possesses. To emphasize this, sometimes it is useful to depict a set with elements scattered about something like



with the elements scattered about. Evidently, a re-arrangement of the elements does not change the depicted set. So



is the same set. Depicting a subset of a set is a simple matter of drawing a smaller boundary around some of the elements as in the following.



Exercises for Lecture 5

Draw depictions of the following sets

1. $\{1, 4, 5, 2, 3\}$
2. $\{1, 2, 3, \ldots, 23\}$
3. $\{a, b, c, d, e\}$ and $\{c, e, f, g\}$ on the same diagram
4. $\{a, e, b, c, e\}$ [sic]
5. $\{1, 3, 6, 7\}$ and $\{1, 3, 5, 6, 7, 9\}$ on the same diagram
6. $\{\textsc{f}, \textsc{t}$
7. $\{\bullet\}$
8. $\{\textsc{t}, \textsc{f}, 3, 5, 1, \bullet\}$

Lecture 6

Functions

Goals

Lecture

- Introduce basic structure of functions
- Define the identity functions and function composition
- Introduce internal diagrams of functions.

Study

- Be able to determine equality of functions
- Use internal diagrams to depict function composition

Functions (perhaps in your calculus courses) are often talked about as *operations*. For example,

$$f(x) = x^2$$

can be seen as an operation that transforms a number x into its square. But it can also be seen as an attribute (the "square of x"). The "operational" view is informal, and often useful. As we will see, though, it gets an important aspect of functions wrong because two entirely different operations may define the same function.

Informally, a function "takes" an element of a given set as input and "produces" an element of a given set as output. So the function f defined by $f(x) = x^2 + 2x + 1$ might "take" the natural number 2 and "produce" the natural number 10. That is, $f(3) = 3^2 + 1 = 10$. We begin by making this idea formal, introducing the vocabulary of functions.

Principle 6.1

Basic Structure of Functions

- For a set X and a set Y, there are things called *functions from* X *to* Y. We write $f\colon X \to Y$ or $A \xrightarrow{f} B$ to indicate that f is a function from X to Y.
- For $f\colon X \to Y$, the set X is called the *domain* of f and Y is called the *codomain* of f.
- For any function $f\colon X \to Y$ and any element $a \in X$, f and a determine an element of Y, written $f(a)$, and read "f of a".

A function may sometimes also be called a *map*, a *transformation*, or an *operation*. As we will see, however, *operation* is somewhat misleading, so we usually avoid it.

Often, a function $f\colon X \to Y$ is *defined* by a rule, just as they are in other parts of mathematics. We typically, write such rules by giving the function a name (very often f because we are lazy) and spelling out the rule at the same time. So we write things like

$$f(x) = x^2 + 4x + 2$$

to define a function $f\colon \mathbb{R} \to \mathbb{R}$ (recall that $\mathbb{R}$ is the set of real numbers). But sometimes it is useful to have a rule without giving it a name. To do that, we will use the "maps to" arrow $\mapsto$. So we may define the same function f by saying that f *is given by the rule*

$$x \mapsto x^2 + 4x + 2.$$

The rule $x \mapsto x^2 4x + 2$ is the same as the rule $y \mapsto y^2 + 4y + 2$. The variable only serves as a place holder, so its particular name does not matter.

There are two fundamental (trivial) types of rules that can be used to build functions.

Principle 6.2

Identities and Function Composition

- For any set X, there is a function $\mathsf{id}_X\colon X \to X$ defined by the rule $x \mapsto x$. This is called the *identity* function on A.
- For any two functions $f\colon X \to Y$ and $g\colon Y \to Z$, there is a function $g \circ f\colon X \to Z$ defined by the rule $x \mapsto g(f(x))$. This is called the *composition of* g *and* f (or sometimes g *following* f).

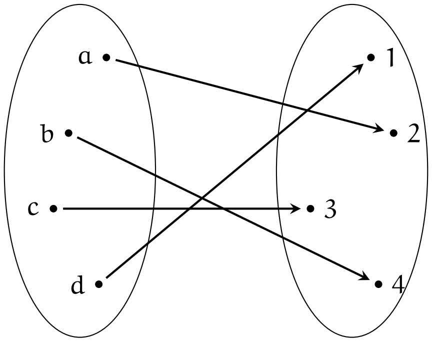
Notice that $g \circ f$ is only defined when the *domain* of g matches exactly the *codomain* of f.

Exercises for Lecture 6

1. Suppose $f\colon W \to X$, $g\colon X \to Y$ and $h\colon Y \to Z$ are functions. Then $h \circ (g \circ f)$ and $(h \circ g) \circ f$ are functions from W to Z. Do you think they are equal? Explain your answer in a few clearly written sentences.

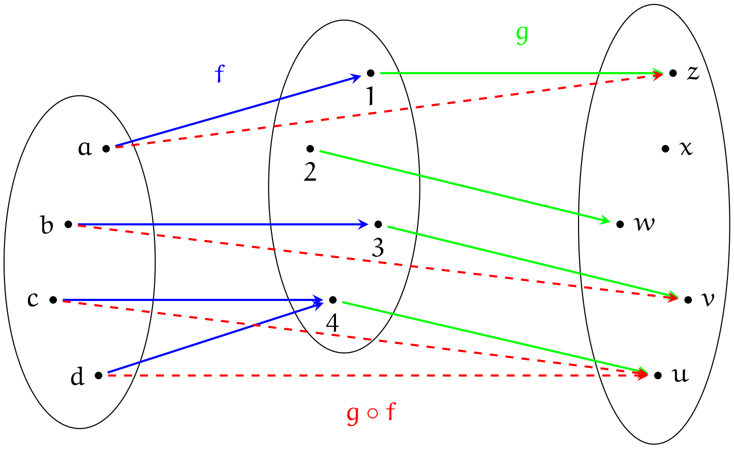
6.1 Internal Diagrams

To depict a function on small sets, we can use the internal diagrams of the last section with arrows indicating the input/output relationship. For example,



depicts a function from the set $\{a, b, c, d\}$ to the set $\{1, 2, 3, 4\}$.

Composition can also be illustrated using internal diagrams. For example,



Exercises for Lecture 6

Use internal diagrams for the following exercises.

1. Depict four different functions from the set $\{1, 2, 3\}$ to the set $\{\text{F}, \text{T}\}$. [Draw four different diagrams.]

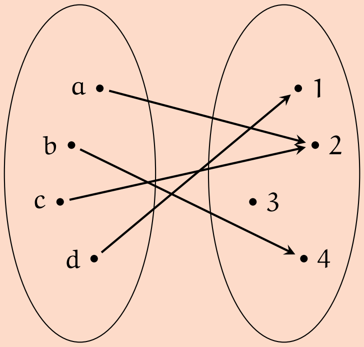
2. Depict all of the functions from $\{\bullet\}$ to $\{a, b, c\}$

3. Depict all of the functions from $\{a, b, c\}$ to $\{\bullet\}$

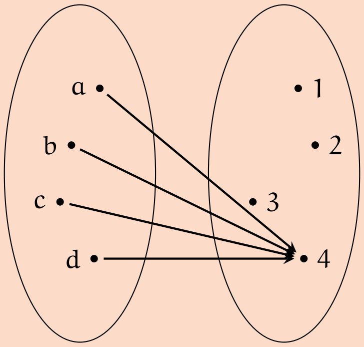
4. Are there any functions from $\{a, b\}$ to $\emptyset$?

5. Are there any functions from $\emptyset$ to $\{a, b\}$? If there are, how many?

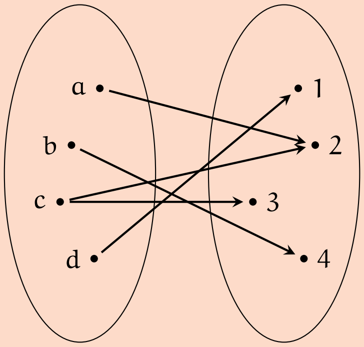
6. For each of the following diagrams, determine whether or not the diagram depicts a function. If not, explain why not.



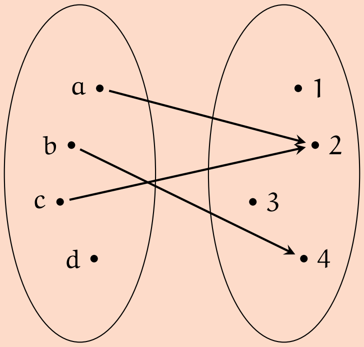
1.



3.



2.



4.

7. Let $A = \{1, 2, 3\}$. Let $B = \{a, b, c, e\}$ and let $C = \{\mathrm{F}, \mathrm{T}\}$. Depict some functions $f \colon A \to B$, $g \colon B \to C$, and $g \circ f$.

8. Think about how you might depict a function $h \colon A \to A$ using only one picture of the set A. Describe what you would do, and provide an example.

9. Suppose $f \colon \mathbb{R} \to \mathbb{R}$ is given by the rule $x \mapsto x^2$, suppose $g \colon \mathbb{R} \to \mathbb{R}$ is given by the rule $x \mapsto x - 1$. Write rules for $f \circ g$ and $g \circ f$ without using the symbols f and g. Explain whether or not it is the case that $f \circ g = g \circ f$.

6.2 Extensionality

As with sets, we need a way to say when two functions are equal. Consider an example. Recall that $\mathbb{N}$ denotes the set of natural numbers. Then define $f \colon \mathbb{N} \to \mathbb{N}$ and $g \colon \mathbb{N} \to \mathbb{N}$ by

$$f(n) = n^2 + 2n + 1$$
$$g(n) = (n + 1)^2$$

Evidently, for each $n \in \mathbb{N}$, it is true that $f(n) = g(n)$. So even though f and g are defined by different *operations*, the two functions yield the same results. This leads to a principle for equality of functions.

Principle 6.3

Equality of Functions For functions $f \colon X \to Y$ and $g \colon X \to Y$, if it is the case that $f(x) = g(x)$ for all $x \in X$, then $f = g$. Note that equality of functions only makes sense when the two functions share the same domain and the same codomain.

When we are not concerned about the detailed internals of sets, but only with how functions interact, then an individual function can be depicted very simply as $X \xrightarrow{f} Y$. So a composition of functions can be depicted as in

$$\begin{tikzcd}
X \arrow[r, "f"] \arrow[dr, "g \circ f"'] & Y \arrow[d, "g"] \\
 & Z
\end{tikzcd}$$

We do not really need to draw $g \circ f$ as a separate arrow because the *path* from X to Y to Z is already an implicit depiction of $g \circ f$. So the simpler diagram

$$\begin{tikzcd}
X \arrow[r, "f"] & Y \arrow[d, "g"] \\
 & Z
\end{tikzcd}$$

shows the same information, namely, that $f \colon X \to Y$ and $g \colon Y \to Z$ are functions and therefore, $g \circ f \colon X \to Z$ is too.

Now a diagram such as this

$$\begin{tikzcd}
W \arrow[r, "f"] \arrow[d, "h"'] & X \arrow[d, "g"] \\
Y \arrow[r, "k"'] & Z
\end{tikzcd}$$

depicts two composite functions $g \circ f$ and $k \circ h$, but $g \circ f$ and $k \circ k$ may not be equal. We say that the diagram *commutes* or that it is a *commutative diagram* if $g \circ f = k \circ h$. In other words, saying that a certain diagram commutes *is* an assertion that certain functions are equal.

Exercises for Lecture 6

1. For each of the following pairs of functions $\mathbb{N} \to \mathbb{N}$, determine whether they are equal and explain why or why not.

    1. $f(n) = 2n + 3$ and $g(m) = 2m + 3$
    2. $f(n) = 2^{n+1} - 1$ and $g(n) = \sum_{i=0}^{n} 2^i$
    3. $f(n) = n^2 + 5n + 6$ and $g(n) = (n + 3)(n + 2)$
    4. $f(n) = n^4 - 10n^3 + 35n^2 + 50n + 24$ and $g(n) = 24$

2. Let $\mathbb{R}$ denote the set of all real numbers. Let $f(x) = \tan(x)$. Explain why this does *not* define a function from $\mathbb{R}$ to $\mathbb{R}$.

3. Suppose the following functions exist: $f\colon W \to X$, $g\colon X \to Y$, $a\colon W \to Z$, $b\colon Y \to Z$. Draw a commutative diagram asserting that $b \circ g \circ f = a$.

4. Suppose the following functions exist: $f\colon C \to A$, $g\colon C \to B$, $h\colon C \to P$, $p\colon P \to A$ and $q\colon P \to B$. Draw a commutative diagram asserting that $f = p \circ h$ and $g = q \circ h$.

Lecture 7

Constructing Sets and Functions

Goals

Lecture

- Characterize and define
  - Pointer and constant functions
  - Solution sets
  - Characteristic functions
  - Products of sets
  - Exponents of sets

Study

- Be able to calculate membership in various constructed sets
- Learn to use universal constructions to define functions.

So far, we have thought mainly about informally defined sets and functions. To fill out our understanding of sets, we need to be able to build sets and functions for specific purposes and with specific structure in mind.

Three finite sets will play particularly important roles in this. We have already discussed the set $\emptyset$, consisting of no elements. We also need a designated set with one element and a designated set with two elements. We denote these by $\mathbb{1}$ and $\mathbb{2}$. It does not matter at all *what* the elements are because, as we will soon see, sets of the same size are interchangable.

For the time being, we merely need to agree on a fixed reference set with one element and a fixed reference set with two elements. The particular choices we make here will be clearer as we put them to use.

Definition 7.1

Let $\bullet$, $\textsc{f}$ and $\textsc{t}$ be fixed symbols. Then define

$$\begin{aligned}\mathbb{1} &= \{\bullet\}\\ \mathbb{2} &= \{\textsc{f}, \textsc{t}\}\end{aligned}$$

The single element of $\mathbb{1}$ is intended to look like a generic point in an internal diagram. The element $\textsc{t}$ is meant to indicate ‘True’ and $\textsc{f}$, ’False’.

7.1 Elements, Pointers and Constant Functions

Suppose we are told that $p\colon \mathbb{1} \to X$ is a function. Since $\bullet \in \mathbb{1}$, this function determines an element of X, namely $p(\bullet)$. A picture of the situation might be this:

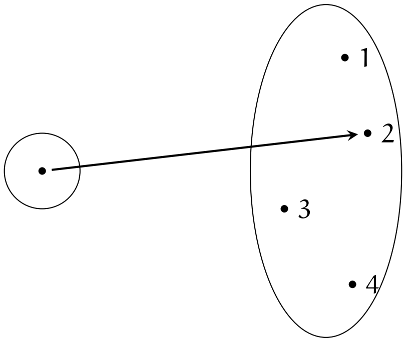


Figure 7.1: A function from $\mathbb{1}$ "points" to an element

Since $\bullet$ is the only element of $\mathbb{1}$, p can "point" only to a single element of X. So we might refer to a function $\mathbb{1} \xrightarrow{p} X$ as a *pointer* into X. Each pointer determines an element of X. And conversely, it should be possible to point to any element of X. This leads to a principle guaranteeing that certain (nearly trivial) functions exist.

Principle 7.1

Elements Determine Pointers

For any set X, and any $a \in X$, there is a function $\hat{a} \colon \mathbb{1} \to X$ given by the rule $x \mapsto a$.

Thus the function depicted in Figure 7.1 is $\hat{2}$. This principle simply asserts that elements of a set X and functions $\mathbb{1} \to X$ are interchangible: from $a \in X$ we get $\hat{a} \colon \mathbb{1} \to X$; from $p \colon \mathbb{1} \to X$. we get the element $p(\bullet)$.

Suppose $f \colon X \to \mathbb{1}$ and $g \colon X \to \mathbb{1}$ are functions, that is, their *codomain* is $\mathbb{1}$ instead their *domain*. Then $f(a) = \bullet = g(a)$ is true for every $a \in X$ because $\bullet$ is the only possible value. So $f = g$ by the Principle of Function Extensionality. In other words, there is at most one function from X to $\mathbb{1}$. But the rule $x \mapsto \bullet$ is as simple a rule as one can imagine. This leads to another definition and another principle.

Definition 7.2

A set T is *terminal* if it is the case that for any set X there is exactly one function from X to T.

Principle 7.2

The set $\mathbb{1}$ is a terminal set. We denote the unique function from X to $\mathbb{1}$ by $\diamond_X \colon X \to \mathbb{1}$.

The rule defining $\diamond_X$ is

$$x \mapsto \bullet.$$

Using $\diamond_X$ and $\hat{b}$ for an element $b \in Y$, we can now define a constant function. That is $\hat{b} \circ \diamond_X$ is a function from X to Y given by the rule $x \mapsto \hat{b}(\diamond_X(x)) = \hat{b}(\bullet) = b$. In short, this is the function sending all elements of X to the constant b. It will be convenient to have a standard name for this function.

Definition 7.3

For a sets X and Y, and element $b \in Y$, let $c_{X,b} = \hat{b} \circ \diamond X$. When X is obvious form context, we omit it.

Exercises for Lecture 7

1. Show that for any pointer $p\colon \mathbb{1} \to X$, it is the case that $\widehat{p(\bullet)} = p$.
2. Show that any set with exactly one element is a terminal set.
3. Show that any terminal set has exactly one element.
4. Suppose that $f\colon X \to Y$ is a function. Show that for every $a \in X$, $\widehat{f(a)} = f \circ \hat{a}$.

7.2 The Empty Set

For trivial reasons, there is at most one function from $\emptyset$ to X, for any set X. That is, if $f, g\colon \emptyset \to X$ are functions, then for each $x \in \emptyset$, $f(x) = g(x)$ because there are no x's to concern us. Hence by Principle **??**, $f = g$. The empty "rule" that tells us to do nothing specifies a function from $\emptyset$ to X. So for any set X, there is exactly one function from $\emptyset$ to X. Let's make that official.

Definition 7.4

An *initial set* is a set I so that for any set X, there is exactly one function from I to X.

Principle 7.3

The emptyset $\emptyset$ is an initial set. For a set X, the unique function from $\emptyset$ to X (given by the empty rule) may be denoted by $\Box_A : \emptyset \to X$.

Notice that a function $X \to \emptyset$ is impossible unless X is also empty. So $\emptyset$ is the only initial set.

Exercises for Lecture 7

1. How many functions are there from $\emptyset$ to $\{a, b, c, d\}$? Explain.
2. How many functions are there from $\{a, b, c, d\}$ to $emptyset$? Explain.
3. How many functions are there from $\mathbb{1}$ to $\{a, b, c, d\}$? Explain.
4. How many functions are there from $\{a, b, c, d\}$ to $\mathbb{1}$? Explain.

7.3 Solution Sets, Subsets, Characteristic Functions

Suppose we are given two functions that are "parallel": $f\colon X \to Y$ and $g\colon X \to Y$. To aid readability, we will write this as $X \overset{f}{\underset{g}{\rightrightarrows}} Y$. For some values $a \in X$, it might be the case that $f(a) = g(a)$. Let us call such a value a *particular solution to the equation* $f(x) = g(x)$.

It might be the case that there are no particular solutions to an equation $f(x) = g(x)$. For example, there are no natural numbers n such that $n + 1 = n$. On the other hand, there might be many particular solutions. For example, let $f(x) = x^3$ and let $g(x) = 6x^2 - 11x + 6$ both regarded as functions on the natural numbers. Then it is easy to check that 1, 2 and 3 solve the equation $f(x) = g(x)$. In fact, these three are the only particular solutions. We generalize as follows.

Definition 7.5

For two functions $X \overset{f}{\underset{g}{\rightrightarrows}} Y$, a *solution* is a function $S \xrightarrow{s} X$ so that $f \circ s = g \circ s$. Thus for example, if $a \in A$ is a particular solution then the pointer $\hat{a}$ is a solution.

For functions $A \overset{f}{\underset{g}{\rightrightarrows}} B$, an *equalizer* is a solution $E \xrightarrow{e} X$ so that for any solution $S \xrightarrow{s} X$, there is exactly one function $S \xrightarrow{h} E$ so that $e \circ h = k$.

Principle 7.4

For functions $X \overset{f}{\underset{g}{\rightrightarrows}} Y$, the collection of all particular solutions to the equation $f(x) = g(x)$ form a set, denoted by $\{x \in X \mid f(x) = g(x)\}$. The function $\{x \in X \mid f(x) = g(x)\} \xrightarrow{i} A$ given by the rule $x \mapsto x$ (called an *inclusion map*) is an equalizer for f and g.

If $S \xrightarrow{s} X$ is a solution (that is, $f \circ s = g \circ s$), then the function $C \xrightarrow{\check{s}} \{x \in A \mid f(x) = g(x)\}$ given by the rule

$$x \mapsto s(x)$$

is the unique function for which $s = i \circ \check{s}$.

This axiom tells us three main things. First, we can form a subset of X by specifying an equation $f(x) = g(x)$ for two functions $X \overset{f}{\underset{g}{\rightrightarrows}} Y$, and picking out the particular solutions. Second, a subset formed in this way "embeds" in the given set X by its inclusion map i. Third, for any solution s, the function into the set of particular solutions is defined by the same rule as s.

External diagrams help us understand equalizers. An equalizer is a solution

$$\begin{tikzcd}
E \arrow[r, "e"'] & X \arrow[r, shift left, "f"] \arrow[r, shift right, "g"'] & Y
\end{tikzcd}$$

so that if

$$\begin{tikzcd}
S \arrow[dr, "s"] & & \\
E \arrow[r, "e"'] & X \arrow[r, shift left, "f"] \arrow[r, shift right, "g"'] & Y
\end{tikzcd}$$

is also a solution ($f \circ s = g \circ s$), then there is exactly one function making

$$\begin{tikzcd}
S \arrow[d, "\check{s}"'] \arrow[dr, "s"] & & \\
E \arrow[r, "e"'] & X \arrow[r, shift left, "f"] \arrow[r, shift right, "g"'] & Y
\end{tikzcd}$$

commute.

Inverse Image of an Element

Suppose $c \in Y$ and $X \xrightarrow{f} Y$ is a function, then we can form the equalizer of f and the constant function $\hat{c} \circ \Diamond_X$. This is more easily written we $\{x \in X \mid f(x) = c\}$. Since it is common to pick out sets like this, special notation is in order.

Definition 7.6

For a function $X \xrightarrow{f} Y$, and an element $c \in Y$,

$$f^{-}(c) = \{x \in X \mid f(x) = c\}.$$

In this case, $f^{-}(c)$ is called the *inverse image of* c *with respect to* f.

A diagram can help us understand inverse images as well. Suppose $f\colon X \to Y$ and $c \in C$, then we can arrange a diagram

$$\begin{tikzcd}
 & \mathbb{1} \arrow[d, "\hat{c}"'] \\
X \arrow[r, "f"'] & Y
\end{tikzcd}$$

The inverse image is a subset of X with an inclusion map that makes the following diagram commute:

$$\begin{tikzcd}
f^{-}(c) \arrow[r, "\Diamond_{f^{-}(c)}"] \arrow[d, "i"'] & \mathbb{1} \arrow[d, "\hat{c}"] \\
A \arrow[r, "f"'] & C
\end{tikzcd}$$

For any other function $W \xrightarrow{g} X$ that makes following similar diagram commute:

$$\begin{tikzcd}
W \arrow[rrd, "\Diamond_W"] \arrow[rdd, "g"'] & & \\
 & f^{-}(c) \arrow[r, "\Diamond_{f^{-}(c)}"'] \arrow[d, "i"'] & \mathbb{1} \arrow[d, "\hat{c}"] \\
 & A \arrow[r, "f"'] & C
\end{tikzcd}$$

there is a unique function $B \xrightarrow{\breve{g}} f^{-}(c)$ making

$$\begin{tikzcd}
W \arrow[rrd, "\Diamond_W"] \arrow[rdd, "g"'] \arrow[rd, "\breve{g}"] & & \\
 & f^{-}(c) \arrow[r, "\Diamond_{f^{-}(c)}"'] \arrow[d, "i"'] & \mathbb{1} \arrow[d, "\hat{c}"] \\
 & X \arrow[r, "f"'] & Y
\end{tikzcd}$$

commute.

In Definition 7.6, the set $f^{-}(c)$ is a subset of X. It would be good to know that any subset of X can be described as an inverse image. This is where the set $\mathbb{2}$ plays a role.

Definition 7.7

Subset Classifier

A *pointed set* is a set P with a distinguished element $p \in P$.

A *subset classifier* is a set T with a distinguished element $t \in T$ so that for any set X and any subset $A \subseteq X$, there is exactly one function $k \colon X \longrightarrow T$ for which $A = k^{-}(t)$. That is, A is uniquely defined as the inverse image of t with respect to a function into T.

Principle 7.5

$\mathbb{2}$ and Characteristic Functions

The set $\mathbb{2}$ with the distinguished element T is a subset classifier. For subset $A \subseteq X$, the function corresponding to A, called the *characteristic function of* A, is denoted by κ_A. In other words, κ_A is the unique function for which $A = \kappa_A^{-}(\mathrm{T})$.

For $A \subseteq X$, the characteristic function is defined by the rule

$$x \longmapsto \begin{cases} \mathrm{T} & \text{if } x \in A \\ \mathrm{F} & \text{otherwise} \end{cases}$$

Just as Principle 7.1 asserts that elements of X and functions $\mathbb{1} \longrightarrow X$ are interchangeable, Principle 7.5 asserts that the subsets of X and the functions $X \longrightarrow \mathbb{2}$ are interchangeable.

Exercises for Lecture 7

1. Draw a depiction of $A = \{a, b, c, d, e, f, g\}$ and its subset $B = \{a, c, e, g\}$ in the same internal diagram. Now depict the characteristic map for B as a subset of A.

2. Define two functions $\mathbb{N} \overset{f}{\underset{g}{\rightrightarrows}} \mathbb{N}$ so that the set of particular solutions of $f(x) = g(x)$ is $\{1, 5\}$.

3. Consider the functions $f \colon \mathbb{R} \longrightarrow \mathbb{R}$ defined by $f(x) = \sin(x) + \cos(x)$, and $s \colon \mathbb{N} \longrightarrow \mathbb{R}$ defined by $s(n) = 2\pi n^2$. Is s a solution for the equation $f(x) = -1$? What is the set of all particular solutions?

4. Describe what a subset classifier is, using diagrams similar to the diagrams we have used to describe equalizers and inverse images. That is, for a start, we have a diagram

$$\begin{tikzcd} \mathbb{1} \arrow[d, "\hat{t}"] \\ T \end{tikzcd}$$

Now suppose we are given a subset $A \subseteq X$ with its inclusion map:

$$\begin{tikzcd}
A \arrow[r, "\diamond_A"] \arrow[d, "i"'] & \mathbb{1} \arrow[d, "\hat{t}"] \\
X & S
\end{tikzcd}$$

What additional function is required to exist? What properties is it required to have? [Hint: the result should be that A is an inverse image.]

7.4 Product Sets and Functions of Two Arguments

We should be able to deal with functions of more than one argument, such as a function $f(x,y) = x^2 + y^2$. To account for these, we take our cue from Descartes.

Descartes studied the geometric plane in terms of a coordinate system consisting of the so-called x-axis and y-axis (what we call cartesian coordinates in his honor). Once we have decided where to place the axes (as long as they do not run in parallel), a pair such as $(2,3)$ determines a point on the plane, and any point p in the plane determines a pair. So Descartes realized that we might as well just say that the plane actually *is* the collection of all pairs of real numbers. What makes this work is that points in the plane *project* onto the two axes in a universal way. Products of sets generalize this idea.

Definition 7.8

For sets X and Y, a *table* consists of two functions $X \xleftarrow{f} T \xrightarrow{g} Y$. Note that the two functions have the same domain. We may call the two functions *legs* of the table.

For sets X and Y, a *product of* X *and* Y is a table $X \xleftarrow{p} P \xrightarrow{q} Y$ so that for any table $X \xleftarrow{f} T \xrightarrow{g} Y$ there is exactly one function $T \xrightarrow{h} P$ for which $f = p \circ h$ and $g = q \circ h$. For a product, the legs p and q are called the *projections*.

Principle 7.6

For sets X and Y, the collection of all pairs (x,y) where $x \in X$ and $y \in Y$ is a set, denoted by $X \times Y$. The functions $X \xleftarrow{\pi_0} X \times Y \xrightarrow{\pi_1} Y$ given by the rules $(x,y) \mapsto x$ and $(x,y) \mapsto y$ are projections. For $X \xleftarrow{f} T \xrightarrow{g} Y$, the unique function required by the product may be denoted by $\langle f, g \rangle$.

For $X \xleftarrow{f} T \xrightarrow{g} Y$, the function $T \xrightarrow{\langle f,g \rangle} X \times Y$ is given by the rule $t \mapsto (f(t), g(t))$.

As with equalizers and inverse images, products can be described in terms of diagrams.

A product of X and Y is depicted as an external diagram

$$\begin{tikzcd}
X & P \arrow[l, "p"] \arrow[r, "q"'] & Y
\end{tikzcd}$$

so that for any other table over X and Y:

$$\begin{tikzcd}
& T \arrow[dl, "f"'] \arrow[dr, "g"] & \\
X & P \arrow[l, "p"] \arrow[r, "q"'] & Y
\end{tikzcd}$$
,

there is a unique function making

$$\begin{tikzcd}
& T \arrow[dl, "f"'] \arrow[dr, "g"] \arrow[d, "\langle f, g\rangle"'] & \\
X & P \arrow[l, "p"] \arrow[r, "q"'] & Y
\end{tikzcd}$$
,

commute.

Suppose we are given two unrelated functions $X \xrightarrow{f} Y$ and $A \xrightarrow{g} B$. We can form a single function from $X \times A \xrightarrow{f \times g} Y \times B$ by combining f and g "independently". That is, define $f \times g = \langle f \circ \pi_0, g \circ \pi_1 \rangle$. Calculating concretely in terms of elements $(f \times g)(x, y) = (f(x), g(y))$. So $f \times g$ acts on a pair (x, y) by applying f to x and unrelatedly applying g to y.

Products can by generalized to three, four or more sets. For example, given sets X, Y and Z, we might write $X \times Y \times Z$ for the set of triples (x, y, z) where $x \in X$, $y \in Y$ and $z \in Z$. Instead of two projections, this would have three projections $(x, y, z) \mapsto x$, and so on. It turns out, however, that binary products are enough because $X \times (Y \times Z)$ behaves just like $X \times Y \times Z$.

Exercises for Lecture 7

1. For the sets $A = \{a, b, c\}$ and $B = \{1, 2, 3, 4\}$, write out $A \times B$ and $B \times A$

2. What is $\emptyset \times A$?

3. Write out $\{4, a, 0\} \times 2$.

4. Describe in plain English what are the elements of $\mathbb{N} \times \mathbb{N}$.

5. Suppose A is a finite set with m elements and B is a finite set with n elements. How many elements are in $A \times B$?

6. Describe in plain English why it makes sense to refer to $A \times B$ as a "product."

7. For sets $A = \{a, b\}$, $B = \{0, 1, 2\}$ and $C = \{c, d\}$, calculate $A \times B \times C$, $A \times (B \times C)$ and $(A \times B) \times C$. Are these sets equal? If not, how do they differ.

8. Describe a model of the standard fifty-two card poker deck as a product of two sets.

7.5 Function Sets and Parametric Functions

A function from X to Y might depend on a parameter from another set P. For example, the function $\mathbb{R} \xrightarrow{f} \mathbb{R}$ given by the rule $x \mapsto \sin(x + c)$ depends on the constant c. There is a related function $\mathbb{R} \times \mathbb{R} \xrightarrow{g} \mathbb{R}$ given by $(c, x) \mapsto \sin(x + c)$. Though g describes the same behavior as f, it makes the parameter c explicit as another argument. The relation between f and g leads to the following definition.

Definition 7.9

For sets P, X and Y, a *parametric function from* X *to* Y is a function $P \times X \xrightarrow{g} Y$ for some set P. The set P will be called the *parameter set*.

Suppose $Q \times X \xrightarrow{f} Y$ is a parametric function with parameter set Q and $P \xrightarrow{k} Q$ is a function. Then another parametric function with parameter set P by composing: $f \circ (k \times \mathsf{id}_X)$. The function k acts like a *change of parameters* because it transforms the parametric function with parameters in Q into a parametric function with parameters in P. Specifically, $f \circ (k \times \mathsf{id}_X)$ is given by the rule $(c, a) \mapsto f(k(c), a)$.

An *evaluation map* for X and Y is a parametric function $F \times X \xrightarrow{a} Y$ with parameter set F, so that for any parametric function $P \times X \xrightarrow{g} Y$ there is exactly one change of parameters $h \colon P \to F$ so that $g = a \circ (h \times \mathsf{id}_X)$. In that case, F is called an *exponential with base* Y *and exponent* X.

Principle 7.7

For sets X and Y, the collection of all functions from X to Y, denoted by Y^X, is a set.

The rule $(f, x) \mapsto f(x)$ defines an evaluation map $Y^X \times X \xrightarrow{\mathsf{appl}} Y$.

For a parametric function $f \colon P \times X \to Y$, the unique function from P to Y^X determined by f does not have a completely standard name. Increasingly, mathematicians honor the twentieth century logician, Haskell Curry, by referring to this as ‘currying’. For these lectures, we follow that tradition and write $\mathsf{curry}[f]$ for the unique function satisfying $f = \mathsf{appl} \circ (\mathsf{curry}[f] \times \mathsf{id}_X)$.

Calculating how $P \xrightarrow{\mathsf{curry}[f]} Y^X$ must behave, we see that for any parameter $p \in P$, $\mathsf{curry}[f](p) \in Y^X$ is the function from X to Y given by the rule $x \mapsto f(p, x)$.

λ Notation

In defining $\mathsf{curry}[f]$, we needed to describe certain elements of Y^X. But elements of Y^X are functions. And a function typically is described by a rule. So it would be convnient to have a notation that permits us to describe the behavior of a function without giving the function a name. The logician Alonzo Church was interested in the fundamental idea of just what *is* a function. He proposed a notation for describing functions, writing things like $\lambda x.x^2$ to describe the function that squares its input. The Greek letter λ means nothing. It is used only as a marker to introduce a function. The “λ” notation is widely adopted in computer science. Indeed, it appears even in languages such as Python. We could make the λ notation formal (as did Church), but for our purposes informality is enough. We use this notation to describe elements of Y^X. Several examples will help to explain this.

Example 7.1

- For $f \in Y^X$ and $g \in Z^Y$, the composite function $g \circ f \in Z^X$ is $\lambda x.g(f(x))$.
- The element of $\mathbb{N}^{\mathbb{N}}$ defined by $\lambda x.x^{\curvearrowright}$ is the successor function.
- For any $a \in X$, the function $\hat{a} \in X^{\mathbb{1}}$ is $\lambda x.a$.
- For $X \xleftarrow{f} T \xrightarrow{g} Y$, the function $\langle f, g\rangle X \times Y^T$ is $\lambda x.(f(x), g(x))$.
- For a parametric function $P \times X \xrightarrow{f} Y$, the function $P \xrightarrow{\mathsf{curry}[f]} Y^X$ can be defined by the rule $p \mapsto \lambda x.f(p, x)$.
- For any $f \in Y^X$, $f = \lambda x.f(x) = \lambda y.f(y)$.
- The only element of $\mathbb{1}^X$ is $\lambda x.\bullet$.

Exercises for Lecture 7

1. For set $A = \{1, 2, 3\}$ and $B = \{a, b\}$
   1. draw internal diagrams corresponding to each element of B^A (there are eight of them);
   2. draw internal diagrams corresponding to each element of A^B (there are nine of them).
2. If X is a finite set with k elements, Y is a finite set with j elements, how many elements are there in the set Y^X?
3. Consider the function $\mathbb{N} \times \mathbb{N} \xrightarrow{f} \mathbb{N}$ defined by $f(m, n) = m^n$. What element of $\mathbb{N}^{\mathbb{N}}$ is $\mathsf{curry}[f](3)$? Use λ notation to describe it.
4. For the function min from $\mathbb{R} \times \mathbb{R}$ to $\mathbb{R}$ defined to mean the minimum of x and y, define $\mathsf{curry}[\mathrm{min}]$.
5. Use λ notation to describe the element of $\mathbb{N}^{\mathbb{N}}$ that quadruples the square of the input.

LECTURE 8

The Set of Natural Numbers

Goals

Lecture:

- Re-introduce the natural numbers as a set
- Introduce sequences and recursively defined sequences
- Relate recursion to proofs by induction

Study:

- Be able to define simple functions by recursion
- Be able to explain how induction and recursion are related

We have used $\mathbb{N}$ informally to denote the set of natural numbers. It is time that we make the structure of $\mathbb{N}$ explicit within our theory of sets and functions. It will turn out that $\mathbb{N}$ is also a universal construction.

Natural numbers provide a precise picture of counting and of putting things in an order. Now that we have sets and functions we can consider a function $a\colon \mathbb{N} \to A$ to be an *infinite sequence*: $a(0)$, $a(1)$, $a(2)$, ... When we do that, we sometimes write a_0, a_1, a_2, ... instead. Still a itself is just function from $\mathbb{N}$ to A. To emphasize the notation that a represents an infinite sequence, we sometimes also write $(a_i)_{i \in I}$.

Much of what we discuss in this lecture has the feel of computer programming. This is partly because natural numbers are the main objects of calculation. We want to understand, for example, how to define a function like $n \mapsto n!$ (n factorial) as a function from $\mathbb{N}$ to $\mathbb{N}$ by specifying how it is calculated. In particular, $0! = 1$ and $(n^\frown)! = n^\frown \cdot n!$ characterize factorial by spelling out how to calculate it. For example,

$$\begin{aligned}
4! &= 4 \cdot 3! \\
&= 4 \cdot (3 \cdot 2!) \\
&= 4 \cdot (3 \cdot (2 \cdot 1!)) \\
&= 4 \cdot (3 \cdot (2 \cdot (1 \cdot 0!))) \\
&= 4 \cdot (3 \cdot (2 \cdot (1 \cdot 1))) && = 24
\end{aligned}$$

It is quite common to think about a sequence in which a_{n+1} is functionally related to a_n. For example, in the sequence $1, 2, 4, 8, \ldots$, the initial entry is 1 and each successive entry is double its predecessor.

Indeed, if we know just those two facts – the initial entry is 1, and each subsequent entry is double its predecessor – then we know how the entire sequence behaves. Also, we know how to calculate the n^{th} entry, by recursion just like factorial.

The most basic sequence, of course, is 0, 1, 2, ... Its initial entry is 0 and each subsequent entry is the successor of its predecessor. So we think of the sequence that comprises $\mathbb{N}$ as a universal recursively defined sequence.

8.1 Sequences and Simple Recurrences

Let us make the informal word *sequence* official.

Definition 8.1

A *sequence in set* X is a function $a\colon \mathbb{N} \longrightarrow X$.

As we studied in previous lectures, the basic vocabulary of natural numbers is that (i) there is a starting natural number, 0, and (ii) for each natural number n there is a next one, $n^\frown$. To discuss successor in the language of sets and functions, we stipulate that successor is a function $\mathsf{suc}\colon \mathbb{N} \longrightarrow \mathbb{N}$ given by the rule $n \mapsto n^\frown$. So $\mathbb{N}$ is not just a set. It comes with functions $\mathbb{1} \xrightarrow{\hat{0}} \mathbb{N} \xleftarrow{\mathsf{suc}} \mathbb{N}$.

Suppose $\mathbb{1} \xrightarrow{\hat{b}} X \xleftarrow{r} X$ is a similar structure. Then we ought to be able to define a a sequence in X, so that $a_0 = b$, $a_1 = r(b)$, $a_2 = r(r(b))$, and so on. In general, a_k should be determined by starting with b and repeatedly applying r a total of k times.

Definition 8.2

A *simple recurrence* is a set with functions $\mathbb{1} \xrightarrow{\hat{b}} X \xleftarrow{r} X$. We will say the two functions $\hat{b}$ and r form a *recurrence on* X.

A *natural number set* is a simple recurrence $\mathbb{1} \xrightarrow{\hat{z}} N \xleftarrow{s} N$ so that for any other simple recurrence $\mathbb{1} \xrightarrow{\hat{b}} X \xleftarrow{r} X$, there is exactly one function $N \xrightarrow{f} X$ so that $f \circ \hat{z} = \hat{b}$ and $f \circ s = r \circ f$.

The principle we are interested in here is that simple recurrences determine sequences.

Principle 8.1

The collection of natural numbers is a set, denoted by $\mathbb{N}$. Moreover, $\mathbb{1} \xrightarrow{\hat{0}} \mathbb{N} \xleftarrow{\mathsf{suc}} \mathbb{N}$. makes $\mathbb{N}$ a natural numbers set.

From a simple recurrence $\mathbb{1} \xrightarrow{\hat{b}} X \xleftarrow{r} X$, the corresponding unique sequence in X may be denoted by $\mathsf{s\text{-}rec}[b, r]$. So $\mathbb{N} \xrightarrow{\mathsf{s\text{-}rec}[b,r]} X$ is characterized by

$$\begin{aligned}
\mathsf{s\text{-}rec}[b, r]_0 &= b \\
\mathsf{s\text{-}rec}[b, r]_{n^\frown} &= r(\mathsf{s\text{-}rec}[b, r]_n)
\end{aligned}$$

So every simple recurrence on a set X determines a sequence in X. On the other hand, it is not the case that every sequence is determined by a simple recurrence. Take for example, the sequence $0, 1, 0, 2, 0, 3, \ldots$. This can not be defined (at least not directly) by giving an initial entry and specifying successive entries based only on the predecessors. After all, the entries 1, 2, 3 and so on all have the same preceding entry.

8.2 Primitive Recursion

Evidently, addition, multiplication, factorial, and other familiar functions should be definable using Principle 8.1. But there are problems to overcome: Addition is not a sequence, at least not in an obvious way. And factorial is not obviously definable by a simple recurrence because we would need a function $r\colon \mathbb{N} \longrightarrow \mathbb{N}$ so that $n^\frown! = r(n!)$ for all n. If, in place of r, we could use a function that depends on n as well as on $n!$, we could define factorial recursively the usual way.

Putting things together, we consider a scheme that generalizes simple recursion to permit (i) dependence on a parameter not directly involved in the recursion, and (ii) dependence on n at each stage of the recursion.

Definition 8.3

A *primitive recurrence in* A *(with parameters in* C*)* consists of two functions $C \xrightarrow{b} A \xleftarrow{r} \mathbb{N} \times C \times A$.

A *parametric sequence in* A *with parameters in* C is a function $a \colon C \times \mathbb{N} \longrightarrow A$. For a parametric sequence, we may write $a_{c,n}$ instead of $a(c,n)$.

Theorem 8.1

For any primitive recurrence $C \xrightarrow{b} A \xleftarrow{r} C{\times}\mathbb{N}{\times}A$, there is a unique function $\mathsf{p\text{-}rec}[b,r] \colon C \times \mathbb{N} \longrightarrow A$ satisfying:

$$\begin{aligned} \mathsf{p\text{-}rec}[b,r]_{c,0} &= b(c) \\ \mathsf{p\text{-}rec}[b,r]_{c,k^\frown} &= r(c,k,\mathsf{p\text{-}rec}[b,r]_{c,k}) \end{aligned}$$

Proof: The proof of this requires additional ideas that are implict in the way subset classification works. We have all the principles we need, but a proof now would involve quite a long development. So we only sketch the rough idea.

We use simple recursion to define a sequence of "approximations" of the desired function. First, let $D_n = C \times \{0,\dots,n\}$ for each $n \in \mathbb{N}$. Then define $f_n \colon D_n \longrightarrow A$ for each $n \in \mathbb{N}$ by

$$\begin{aligned} f_0(c,0) &= b(c) \\ f_{k^\frown}(c,i) &= f_k(c,i) && \text{if } i \leq k \\ f_{k^\frown}(c,k^\frown) &= r(c,k,f_k(c,k))\text{otherwise}. \end{aligned}$$

Then define $f \colon C \times \mathbb{N} \longrightarrow A$ by $f(c,n) = f_n(c,n)$. It can be checked that f satisfies the desired equations, and that no other function does.

The technical part of the proof is mainly concerned with ensuring that the sequence of functions f_n is indeed definable and that f can be defined from those. □

Example 8.1

The "predecessor" function is defined by the scheme $\mathsf{pred}(0) = 0$ and $\mathsf{pred}(n^\frown) = n$. The primitive recurrence $\mathbb{1} \xrightarrow{\hat{0}} \mathbb{N} \xleftarrow{\pi_1} \mathbb{1} \times \mathbb{N} \times \mathbb{N}$ where π_1 is the projection $(x,y,z) \mapsto y$ determines a function $g \colon \mathbb{1} \times \mathbb{N} \longrightarrow \mathbb{N}$ given by $g(\bullet,0) = 0$ and $g(\bullet,n^\frown) = \pi_2(\bullet,n,g(\bullet,n)) = n$. Hence, we may define $\mathsf{pred}(n) = g(\bullet,n)$

Exercises for Lecture 8

1. Define addition $\mathbb{N} \times \mathbb{N} \xrightarrow{+} \mathbb{N}$ by primitive recursion. That is, find functions $\mathbb{N} \xrightarrow{b} \mathbb{N}$ and $\mathbb{N} \times \mathbb{N} \times \mathbb{N} \xrightarrow{r} \mathbb{N}$ so that
$$m + 0 = b(m)$$
$$m + n^\frown = r(m, n, m + n)$$
That way, $\mathsf{p\text{-}rec}[b, r]$ is addition.

2. Define multiplication $\mathbb{N} \times \mathbb{N} \xrightarrow{\cdot} \mathbb{N}$ by primitive recursion. You may use addition in defining the primitive recurrence.

3. Define the factorial function by primitive recursion. You may use multiplication in defining the primitive recurrence.

4. For a given function $f\colon \mathbb{N} \to \mathbb{N}$, find a primitive recurrence that defines the function $n \mapsto \sum_{i=0}^{n-1} f(i)$. That is, result will be the sequence $0, f(0), f(0) + f(1), f(0) + f(1) + f(2), \ldots$.

5. The operation of *monus* $m \dot{-} n$ is defined to be $m - n$ when $m \geq n$ and to be 0 otherwise. Define monus by primitive recursion.

6. Let $\mathbb{N} \xrightarrow{p} \mathbb{2}$ be a characteristic function. Show that bounded existential quantification is primitive recursive. That is, define the function $\exists_p^<\colon \mathbb{N} \to \mathbb{2}$ by $\exists_<^p(n) = \mathrm{T}$ iff there is at least one $k < n$ for which $p(k) = \mathrm{T}$. Show that this can be defined by a primitive recurrence. [Hint: $\exists_<^p(0) = \mathrm{F}$ because there are no natural numbers below 0. Now ask what value is $\exists_<^p(n^\frown)$ in terms of p and $\exists_<^p(n)$.]

8.3 Primitive Recursion and "for" loops

It is a theorem (we will not prove here) that there is a technical sense in which primitive recursive functions exactly the functions implementable, say in Python, by nested "for" loops. To get a taste of what this means, let us pretend that $P \xrightarrow{b} X$ and $P \times \mathbb{N} \times X \xrightarrow{r} X$ are somehow defined by Python functions. Then the following code implements $\mathsf{p\text{-}rec}[b, r]$.

```
def p_rec($b \in X^P$, $r \in X^{P \times \mathbb{N} \times X}$) $\in X^{P \times \mathbb{N}}$
    def f($p \in P$, $n \in \mathbb{N}$): $\in X$
        $x = b(p)$
        for i in range(n):
            $x = r(p, i, x)$
        return x
    return f
```

Conversely, suppose a function in Python is defined using only natural number variables and the following parts of Python:

- Assignment statements
- Increment statements: "n+=1"
- loops of the form "**for** i **in** range(n): ..."
- evaluations of functions that are defined similarly

The theorem we allude to claims that such a function is definable by primitive recursion. The theorem is not especially difficult, but it does involve a lot of careful checking. The point is that primitive recursion allows us to define a lot of the functions that we expect to be able to program in a standard programming language.

A question arises, however. If “for” loops are sufficient to program any primitive recursive function, why bother with other more complicated loops? One answer is that programming languages are not merely for computing functions. They are used to implement lots of behavior that is not so easily cast in terms of sets and functions. Another answer, though, is internal to set theory. Namely, there are computable functions on $\mathbb{N}$ which are not primitive recursive.

For each $n \in \mathbb{N}$, define the sets $P^n \subseteq \mathbb{N}^{\mathbb{N}^n}$ of n-*ary number theoretic primitive recursive functions* to be the smallest sets satisfying:

- $\hat{0} \in P^0$
- $\mathsf{suc} \in P^1$
- for each $k < n$, $\pi^n_k \in P^n$ where $\pi^n_k \in \mathbb{N}^{\mathbb{N}^n}$ is defined by $\pi^n_k(x_0, \ldots, x_{n-1}) = x_k$
- If $g \in P^n$ and for each $k < n$, $f_k \in P^m$, then $g \circ \langle f_0, \ldots, f_{n-1} \rangle \in P^m$
- if $b \in P^m$ and $h \in P^{m+2}$, then $\mathsf{p\text{-}rec}[b, h] \in P^{m+1}$

Consider the following sequence $A_0, A_1, \ldots$ in $\mathbb{N}^{\mathbb{N}}$ defined by

$$\begin{aligned} & A_0\mathsf{suc} \\ A_{k^\frown} &= \mathsf{p\text{-}rec}[\widehat{A_k(1)}, A_k \circ \pi^2_1] \end{aligned}$$

Putting this in terms of elements

$$\begin{aligned} A_0(m) &= m + 1 \\ A_{k^\frown}(0) &= A_k(1) \\ A_{k^\frown}(m^\frown) &= A_k(A_{k^\frown}(m)). \end{aligned}$$

Evidently, each individual function A_k is a number theoretic primitive recursive unary function. But the function $\mathsf{Ack}\colon \mathbb{N} \longrightarrow \mathbb{N}$ defined by $\mathsf{Ack}(n) = A_n(n)$ is not. The proof is quite ingeneous. Roughly, one shows that Ack grows faster than any number theoretic primitive recursive function.

To get an idea of how fast this function grows, $\mathsf{Ack}(0) = 1$, $\mathsf{Ack}(1) = 3$, $\mathsf{Ack}(3) = 61$, $\mathsf{Ack}(4) = 2^{2^{65536}} - 3$ a number vastly larger than the number of electrons in the visible universe.

8.4 Lists

For a set X, the lists consisting of elements from X also form a set. The structure of this set is similar to $\mathbb{N}$.

Definition 8.4

For a set X, a *simple* X-*recurrrence* is a set with two functions $\mathbb{1} \xrightarrow{\hat{b}} A \xleftarrow{r} X \times A$.

For a set X, an X-*list set* is a simple X-recurrence $\mathbb{1} \xrightarrow{\hat{e}} L \xleftarrow{c} X \times L$ so that for any simple A-recurrence $\mathbb{1} \xrightarrow{\hat{b}} A \xleftarrow{r} X \times A$ there is exactly one function $h\colon L \longrightarrow A$ so that

$$\begin{aligned} h \circ \hat{e} &= \hat{b} \\ h \circ c &= r \circ (\mathsf{id}_X \times h). \end{aligned}$$

Principle 8.2

The collection of lists with items drawn from X is a set, denoted by $\mathsf{List}[X]$. The functions $\mathbb{1} \xrightarrow{\widehat{[]}} \mathsf{List}[X] \xleftarrow{:} X \times \mathsf{List}[X]$ constitute an X-list set, where the function $X \times \mathsf{List}[X] \xrightarrow{:} \mathsf{List}[X]$ is the function given by the rule $(x, L) \mapsto x : L$.

For $\mathbb{1} \xrightarrow{\hat{b}} A \xleftarrow{r} X \times A$, we reuse our notation for simle recursive functions on $\mathbb{N}$ and write

$\mathsf{s\text{-}rec}[\hat{b}, r]$ for the unique function satisfying

$$\mathsf{s\text{-}rec}[\hat{b}, r]([\,]) = b$$
$$\mathsf{s\text{-}rec}[\hat{b}, r](x : L) = r(x, \mathsf{s\text{-}rec}[\hat{b}, r](L))$$

Example 8.2

$\mathbb{1} \xrightarrow{\hat{0}} \mathbb{N} \xleftarrow{+} \mathbb{N} \times \mathbb{N}$ determine a function $\mathsf{s\text{-}rec}[\hat{0}, +]$ from $\mathsf{List}(\mathbb{N})$ to $\mathbb{N}$. The function satifies $\mathsf{s\text{-}rec}[\hat{0}, +]([\,]) = 0$ and $\mathsf{s\text{-}rec}[\hat{0}, +](n : L) = n + \mathsf{s\text{-}rec}[\hat{0}, +](L)$. So $\mathsf{s\text{-}rec}[0, +]$ returns the sum of items in a list natural numbers. Earlier, we wrote this as $\sum L$. In other words, we *defined* $\Sigma = \mathsf{s\text{-}rec}[\hat{0}, +]$.

Like $\mathbb{N}$, $\mathsf{List}[A]$ supports a form of primitive recursion. The details can be worked out by the reader.

Exercises for Lecture 8

1. For a function $f\colon X \longrightarrow Y$, specify a simple X- recurrence that defines the function

$$\mathsf{map}[f]\colon \mathsf{List}[X] \longrightarrow \mathsf{List}[Y]$$

so that

$$\mathsf{map}[f]([a_0, a_1, \ldots, a_{n-1}]) = [f(a_0), f(a_1), \ldots, f(a_{n-1})]$$

2. For $\mathsf{map}[\cdot]$ as defined in the previous exercise, show that for any $f\colon X \longrightarrow Y$ and $g\colon Y \longrightarrow Z$, $\mathsf{map}[g \circ f] = \mathsf{map}[g] \circ \mathsf{map}[f]$.

3. Define concatenation in $\mathsf{List}[A]$ by a simple A recurrence. [Hint: I am asking for a function from $\mathsf{List}[A] \times \mathsf{List}[A]$ to $\mathsf{List}[A]$, but simple A recurrence alone will not do the job, because that can only define a function $\mathsf{List}[A] \longrightarrow X$ for some X. Do this instead: (i) specify a simple A-recurrence to define a function $c\colon \mathsf{List}[A] \longrightarrow \mathsf{List}[A]^{\mathsf{List}[A]}$ for which $c(M)(L)$ is the concatetation of L followed by M, (ii) define $L \star M$ to be $c(M)(L)$.

4. Show that $\mathbb{N}$ constitutes a $\mathbb{1}$-list set. That is, describe two functions $\mathbb{1} \xrightarrow{\hat{e}} \mathbb{N} \xleftarrow{c} \mathbb{1} \times \mathbb{N}$ so that for any two functions $\mathbb{1} \xrightarrow{\hat{b}} A \xleftarrow{r} \mathbb{1} \times A$ there is a unique function $\mathbb{N} \xrightarrow{f} A$ satisfying

$$f(e) = b$$
$$f(c(x, y)) = s(x, f(y))$$

5. Write a scheme for "primitive X-recursion", specified by functions $P \xrightarrow{b} A \xleftarrow{r} P \times \mathsf{List}[X] \times A$. Write the equations involving b and r that should determine a unique function $P \times \mathsf{List}[X] \longrightarrow A$. You do not need to prove that your scheme actually defines a function.

Lecture 9

Powersets

Goals

In this lecture we look at the structure of the collection of all subsets of a given set. This structure is intimately related to how we *describe* subsets. The result of our investigation leads to *logic* as a means of reasoning about descriptions.

For this lecture, we concentrate on a fixed set U, which we will think of as our “universe”. For example, if we are currently mainly interested in natural numbers, U could be taken to be $\mathbb{N}$. If we are interested in poker, it could be Deck. If we are interested in functions on the reals, it could be $\mathbb{R}^{\mathbb{R}}$.

The exponential $\mathbb{2}^U$ is the set of all characteristic maps on U. Each $k \in \mathbb{2}^U$ corresponds to a subset of U by $k^{-1}(\mathrm{T}) = \{x \in U \mid k(x) = \mathrm{T}\}$. Vice versa, a subset $A \subseteq U$ determines a characteristic function $\kappa_A\colon U \to \mathbb{2}$ by the rule

$$x \mapsto \begin{cases} \mathrm{T} & \text{if } x \in A \\ \mathrm{F} & \text{otherwise} \end{cases}$$

So $\mathbb{2}^U$ is *representative* of the collection of all subsets of U. It is convenient also to suppose that the actual collection of subsets of a set U forms a set.

Principle 9.1

For any set U, the collection of subsets of U is a set, denoted by $\mathcal{P}(U)$ and called the *power set of* U. Moreover, there is a function $\ni_X\colon \mathcal{P}(U) \times U \to \mathbb{2}$ defined by

$$\ni_U(A, x) = \begin{cases} \mathrm{T}, & \text{if } x \in A \\ \mathrm{F} & \text{otherwise} \end{cases}$$

The function $\ni_U$ is an evaluation map, meaning that for any function $f\colon P \times U \to \mathbb{2}$, there is a unique function $f^\dagger\colon P \to \mathcal{P}(U)$ for which $f = \ni_U \circ (f^\dagger \times \mathsf{id}_U)$. The function $f^\dagger$ is given by the rule $p \mapsto \{x \in U \mid f(p, x) = \mathrm{T}\}$.

The function $\mathsf{curry}[\ni_U]\colon \mathcal{P}(U) \to \mathbb{2}^U$ is the unique function satisfying $\ni_U = \mathsf{appl}_{U,\mathbb{2}} \circ (\mathsf{curry}[\ni_U] \times \mathsf{id}_U)$. In particular, this must send $A \in \mathcal{P}(U)$ to its characteristic function. So this is none other than the rule $A \mapsto \kappa_A$. Also, $\mathsf{appl}^\dagger_{U,\mathbb{2}}\colon \mathbb{2}^U \to \mathcal{P}(U)$ is the unique function satisfying $\mathsf{appl}_{U,\mathbb{2}} = \ni_U \circ (\mathsf{appl}^\dagger_{U,\mathbb{2}} \times \mathsf{id}_U)$. This must send $k \in \mathbb{2}^U$ to the subset $k^-(\mathrm{T})$. So it is given by the rule $k \mapsto k^-(\mathrm{T})$. In this sense, elements of $\mathcal{P}(U)$ and elements of $\mathbb{2}^U$ are interchangible.

For a function $f\colon U \to V$, $\ni_V \circ (\mathsf{id}_{\mathcal{P}(U)} \times f)$ is a function from $\mathcal{P}(V) \times U$ to $\mathbb{2}$. So there is a unique function from $\mathcal{P}(V)$ to $\mathcal{P}(U)$ determined by f according to the above princple. This is called *inverse image*. We introduce notation for this.

Definition 9.1

For $U \xrightarrow{f} V$, let $\mathcal{P}(V) \xrightarrow{f^-} \mathcal{P}(U)$ denote the unique function given by the rule $B \mapsto \{x \in X \mid f(x) \in B\}$. That is, f^- is the unique function for which $\ni_V \circ (\mathsf{id}_{\mathcal{P}(V)} \times f) = \ni_U \circ (f^- \times \mathsf{id}_U)$. In terms of elements, f^{-} satisfies

$$x \in f^{-}(B) \text{ if and only if } f(x) \in B$$

for every $B \subseteq Y$.

This notation clashes slightly with our earlier definition of inverse image of an element, $f^{-}(b) = \{x \in U \mid f(x) = b\}$. But this is harmless because $f^{-}(b)$ in the earlier usage is the same as $f^{-}(\{b\})$ in the new usage.

Exercises for Lecture 9

1. For $f\colon \mathbb{N} \to \mathbb{N}$ defined by $f(n) = x^2$, what is $f^{-}(\{2, 3, 4, 5, 6, 7, 8, 9\})$?
2. For $\sin\colon \mathbb{R} \to \mathbb{R}$, what is $\sin^{-}(\{-1, 1\})$?
3. Show that for any two functions $f\colon X \to Y$ and $g\colon Y \to Z$, $(g \circ f)^{-} = f^{-} \circ g^{-}$.

9.1 Finitary Structure of $\mathcal{P}(U)$

Suppose A and B are subsets of U. Then it makes sense to consider the elements that A and B have in common. For example, for $A \subseteq \mathbb{N}$ being the set of even natural numbers and $B \subseteq \mathbb{N}$ the set of perfect squares, we might want to concentrate on the set of even, perfect squares, another subset of $\mathbb{N}$. In general, for $A \subseteq U$ and $B \subseteq U$, the elements in common constitute another subset of U. This is called the *intersection* and is denoted by $A \cap B$.

Likewise, we might consider merging A and B into a single set (in our example, the set of numbers that are either even or perfect squares). This is called the *union*.

Intersection is defined by “and”, union is defined by “or”. That is, $x \in A \cap B$ if and only if $x \in A$ *and* $x \in B$; $x \in A \cup B$ if and only if $x \in A$ *or* $x \in B$.

Related to intersection, there is a largest set C so that $C \cap A \subseteq B$. This is called the *residual*, and is denoted by $A \Rightarrow B$. This is defined by “implies”. That is, $x \in A \Rightarrow B$ if it is the case that $x \in A$ implies $x \in B$. Flipping things the other way, there is also a smallest set C so that $A \subseteq B \cup C$. This is called the *set difference* and is denoted by $A \setminus B$. This is defined by “but not”. So $x \in A \setminus B$ if and only if $x \in A$ but not $x \in B$.

These operations on subsets of U are closely related to the logic of propositions. Imagine that U consists of a “universe” of possible worlds. Then subsets of U are collections of worlds where certain things are true. For example, perhaps $P \subseteq U$ is the set of worlds in which pigs fly; $K \subseteq X$ is the set of worlds in which kittens smoke cigars. So $P \cap K$ is the set of worlds in which pigs fly *and* kittens smoke cigars. Likewise, $P \cup K$ is the set of worlds in which *Either* pigs fly *or* kittens smoke cigars. And $P \Rightarrow K$ is the set of worlds in which it is true that if pigs fly, *then* kittens smoke cigars. Finally, $P \setminus K$ is the set of worlds in which pigs fly, but kittens don’t smoke cigars.

Understanding the interaction of $\cap$, $\cup$ and $\Rightarrow$ is closely related to the logic of “and”, “or” and “implies”. If we add a sentence “False” that is never true and another one “True” that is always true, then the logic of “and”, “or” and “implies” form what is called a *Heyting algebra*. The other operation $\setminus$ is less commonly studied (perhaps because most languages to not have a single word meaning “but not”).

In fact, “implies” interacts with “False” in a useful way. It turns out that “P implies False” is essentially the same as saying “P is not true”. And if “P is not true” is not true, then “P” must be true. This oservation indicates that the Heyting algebra of subsets is actually a *Boolean algebra*.

The operation of set difference is not as familiar in a logical setting. It corresponds to“but not”, so $P \setminus K$ is the set of worlds in which pigs fly, but kittens do not smoke cigars.

Lemma 9.1

In the following, let U be a set, and $A, B \subseteq U$ be subsets.

- There is a subset of U, denoted by $A \cap B$, so that for every $C \subseteq U$, $C \subseteq A \cap B$ if and only if $C \subseteq A$ and $C \subseteq B$.
- There is a subset of U, denoted by $A \cup B$, so that for every $C \subseteq U$, $A \cup B \subseteq C$ if and only if $A \subseteq C$ and $B \subseteq C$.
- There is a subset of U, denoted by $A \Rightarrow B$, so that for every $C \subseteq U$, $C \subseteq A \Rightarrow B$ if and only if $C \cap A \subseteq B$.
- There is a subset of U, denoted by $A \setminus B$, so that for every $C \subseteq U$, $A \setminus B \subseteq C$ if and only if $A \subseteq B \cup C$.

Proof: A and B are determined by characteristic functions $\kappa_A \colon X \to 2$ and $\kappa_B \colon X \to 2$. So $\rangle \kappa_A, \kappa_B \langle$ is a function from X to 2×2. If we compose with a function $h\colon 2 \times 2 \to 2$, we have another characteristic function on X. So this determines another subset of X. So all of the above constructions amount to defining suitable functions $2 \times 2 \to 2$.

The four functions corresponding to the subset operations can be given by tables. In these tables, we read the first argument on the left, and the second argument on the top.

$\wedge$	F	T
F	F	F
T	F	T

$\vee$	F	T
F	F	T
T	T	T

$\to$	F	T
F	T	T
T	F	T

$-$	F	T
F	F	F
T	T	F

So for each $h \in \{\wedge, \vee, \to, -\}$, there is a subset $(h \circ \langle \kappa_A, \kappa_B \rangle)^{-}(\mathrm{T})$.

It is routine to check that for $h = \wedge$, the result is $A \cap B$; for $h = \vee$, the result is $A \cup B$; for $h = \to$, the result is $A \Rightarrow B$; and for $h = -$, the result is $A \setminus B$. □

This lemma, together with the fact that $\emptyset$ is the smallest element of $\mathcal{P}(X)$ and X is the largest, can be summarized by saying that $\cap$, $\cup$, $\Rightarrow$, $\emptyset$ and X make $\mathcal{P}(X)$ into what is known as a *Heyting algebra*. We spell out the axioms for Heyting algebras in the next section, but generally speaking these are the structures that correspond to a sort of minimal version of propositional logic in which "and", "or", "implies", "true" and "false" interact in natural ways. Likewise, "or", "and", "but not", "false" and "true" interact in natural ways to determine a co-Heyting algebra.

Additionally, "implies" and "false" interact in a stronger way, as do "but not" and "true". In particular, $\mathcal{P}(U)$ is a *Boolean algebra*. Let us abbreviate $A \Rightarrow \emptyset$ by writing A^* (read this informally as "not A). Then the Law of Double Negation which characterizes Boolean algebras, asserts that double negations do not change anything: $A^{**} = A$.

Lemma 9.2

In $\mathcal{P}(X)$, $A^{**} = A$.

Proof: Calculating the members of A^*, it is easy to check that for every element $x \in X$, either $x \in A$ or $x \in A^*$, but not both. So $x \in A^{**}$ if and only if $x \notin A^*$ if and only if $x \in A$.

Put differently, define the function $\neg\colon 2 \to 2$ by $\neg \mathrm{T} = \mathrm{F}$ and $\neg \mathrm{F} = \mathrm{T}$. Then A^* is defined by $(\neg \circ \kappa_A)^{-}(\mathrm{T})$. Obviously, $\neg \circ \neg = \mathsf{id}_2$. As with the other operations, it is now routine to check that $A^{**} = A$. □

One can define the term "Boolean algebra" to mean "Heyting algebra that satisfies the Law of Double

Negation." So $\mathcal{P}(X)$ is indeed a Boolean algebra.

Exercises for Lecture 9

1. An easy consequence of Double Negation is that $A \Rightarrow \emptyset = U \setminus A$. Prove it.

9.2 Laws of Finitary Set Operations

As we mentioned, $\mathcal{P}(X)$ forms a Boolean algebra. This means that $\cup$, cup, $\Rightarrow$, $\emptyset$ and X satisfy various laws. We spell the most important out here.

Laws

For any set U and any subsets A, B and C:

	Semilattice Laws
Associativity	$A \cap (B \cap C) = (A \cap B) \cap C$
	$A \cup (B \cup C) = (A \cup B) \cup C$
Commutativity	$A \cap B = B \cap A$
	$A \cup B = B \cup A$
Idempotency	$A \cap A = A$
	$A \cup A = A$
	Lattice Laws
Absorptivity	$A = (A \cap B) \cup A$
	$A = (A \cup B) \cap A)$
Ordering	$A = B \cap A$ if and only if $A \cup B = A$
	Bounded Lattice Laws
Identity	$A \subseteq A \cap U$
	$A \cup \emptyset \subseteq A$
	Heyting Algebra Law
Residuation	$A \cap B \subseteq C$ if and only if $A \subseteq B \Rightarrow C$
	co-Heyting Algebra Law
Co-Residuation	$A \setminus B \subseteq C$ if and only if $A \subseteq B \cup C$
	Boolean Algebra Law
Double Negation	$A^{**} \subseteq A$
	Distributive Lattice Laws
Distributivity	$A \cap (B \cup C) \subseteq (A \cap B) \cup (A \cap C)$
	$(A \cup B) \cap (A \cup C) \subseteq A \cup (B \cap C)$
	Other Boolean Laws
de Morgan's Laws	$(A \cap B)^* = A^* \cup B^*$
	$(A \cup B)^* = A^* \cap B^*$

Several remarks are in order.

- The Semilattice Laws describe how $\cap$ and $\cup$ behave without any interaction between the two. In fact, any binary operation that is associative, commutative and idempotent is called a *semilattice operation*.

- The Lattice Laws describe how $\cap$ and $\cup$ interact. The two Absorption Laws together are equivalent to the Ordering Law. For suppose $A = A \cup (A \cap B)$ holds for all A and B. Now suppose $X = Y \cap X$. Then $Y \cup X = Y \cup (X \cap Y) = Y$. Conversely, suppose $A = B \cap A$ implies $B \cup A = B$ for all A and B. Then $(X \cap Y) = (X \cap Y) \cap X$. So $X \cup (X \cap Y) = X$.

- The remaining laws are stated in terms of $\subseteq$ instead of equality. Becaause of the Ordering Laws, $A \subseteq B$ is equivalent to $A = B \cap A$ and also equivalent to $B = A \cup B$.

- The Bounded Lattice Laws indicate that U is the unit element for $\cap$ and $\emptyset$ is the unit element for $\cup$. It follows that $\emptyset$ is the smallest element of $\mathcal{P}(U)$ and U is the largest.

- The Residuation and Co-residuation Laws show that $A \Rightarrow B$ and $A \setminus B$ are defined as *duals* of one another.

- In the Double Negation Law recall that A^* is defined to be $A \Rightarrow \emptyset$. Since $A \cap \emptyset \subseteq A \Rightarrow \emptyset$, it follows from Residuation that $A \subseteq A^{**}$. So in fact, $A = A^{*}*$.

- If we defined $A^{\dagger} = U \setminus A$, then in any co-Heyting algebra, $A^{\dagger\dagger} \subseteq A$. In a Boolean algebra $A^{\dagger} = A^*$.

- With respect to Distributivity, in any lattice, the opposite inclusions hold: $A \cap B \subseteq A \cap (B \cup C)$ and $A \cap B \subseteq A \cap (B \cup C)$, so $(A \cap B) \cup (A \cap C) \subseteq A \cap (B \cup C)$. Similarly, for the opposite inclusion for the second Distributive Law.

- The Distributivity laws are equivalent to each other. Suppose $A \cap (B \cup C) \subseteq (A \cap B) \cup (A \cap C)$ is true for all A, B and C. Then $(X \cup Y) \cap (X \cup Z) \subseteq ((X \cup Y) \cap X) \cup ((X \cup Y) \cap Z) = X \cup ((X \cup Y) \cap Z) \subseteq X \cup ((X \cap Z) \cup (Y \cap Z)) = X \cup (Y \cap Z)$

- The Heyting (or co-Heyting) Law implies Distributivity. Obviously, $A \cap B \subseteq (A \cap B) \cup (A \cap C)$, so $B \subseteq A \Rightarrow ((A \cap B) \cup (A \cap C))$. Likewise $C \subseteq A \Rightarrow ((A \cap B) \cup (A \cap C))$. So $B \cup C \subseteq A \Rightarrow ((A \cap B) \cup (A \cap C))$. And again using Residuation, $A \cap (B \cup C) \subseteq (A \cap B) \cup (A \cap C)$.

Exercises for Lecture 9

In the following, assume that U is a set, and all other sets are subsets of U.

1. Prove, using only the Semilattice Laws, the Absorption Laws and the Bounded Lattice Laws, that $A \cap \emptyset = \emptyset$. Likewise, show that $A \cup U = U$.

2. Prove, using only the Semilattice and Lattice Laws, that the two Distribution Laws are equivalent. That is, show that if $A \cap (B \cup C) \subseteq (A \cap B) \cup (A \cap C)$ holds for all A, B, C, then so does $(A \cup B) \cap (A \cup C) \subseteq A \cup (B \cap C)$. Hint: Let $X = A \cup B$, $Y = A$ and $Z = C$. So $(A \cup B) \cap (A \cup C) = X \cap (Y \cup Z)$. Now use the first distributivity law, followed by absorption, then a second use of distributivity, and association and finally a second use of absorption.

3. Prove, using only the Semilattice, Lattice and Heyting Algebra Laws, the Distributivity Law (either one).

4. Prove, using only the Semilattice, Lattice Laws, Heyting Algebra and Boolean Algebra Laws, that the two de Morgan's Laws hold.

5. Prove that $A \Rightarrow B = A^* \cup B$, using any method.

6. Prove that $A \setminus B = A \cap B^*$, using any method.

9.3 Binary Relations

In many situations, we wish to consider how elements of a set are related to elements of another. For example, if S is set modelling the students at Chapman and M is a set modelling majors the university offers, then "majors in" is a relation between elements of S and elements of M. Perhaps "Jethro majors in Phrenology" is true, while "Aczel majors in Flim-Flam Studies" is not. For a more mathematical example, "is less than" is a relation between natural numbers and natural numbers. So for example "5 is less than 6" is true while "9 is less than 6" is not.

Notice that "majors in" can not be modelled obviously by a function because some students may not have a declared major and some students may actually be double majors. So we need a new idea. A binary relation between X and Y determines, for each $x \in X$ and each $y \in Y$, whether or not x is in the relation to y. There are several equivalent ways to model the general notion. But traditionally, the following is usually taken to be the official version.

Definition 9.2

A *relation between* X *and* Y is a subset $R \subseteq X \times Y$. We can write $x \mathrel{R} y$ instead of $(x, y) \in R$ to mimic familiar examples.

A *relation on* X is a relation between X and X.

Here are two common equivalent formulations.

- $R \subseteq X \times Y$ determines a characteristic function $\kappa_R \colon X \times Y \to 2$ so that $R = \{(x, y) \in X \times Y \mid \kappa_R(x, y) = \mathrm{T}\}$.
- R determines a function $R[-] \colon X \to \mathcal{P}(Y)$ so that $R = \{(x, y) \in X \times Y \mid y \in R[x]\}$.

The ability to move between these can be helpful. So it is worth practicing.

Exercises for Lecture 9

Let $r \colon \mathbb{N} \times \mathbb{N} \to 2$ be the function defined by

$$r(m, n) = \begin{cases} \mathrm{T} & \text{if } m \geq n^2 \\ \mathrm{F} & \text{otherwise.} \end{cases}$$

Consider the relation $R = r^{-}(\mathrm{T})$. Explain your answers for the following.

1. Is it the case that $3 \mathrel{R} 2$?
2. Is it the case that $9 \mathrel{R} 3$?
3. What is $R[5]$?
4. What is $\mathsf{curry}[r]$?

We will discuss the structure of relations in more detail in Lecture **??**.

9.4 Quantifiers and Completeness

Consider how we might try to define the set of perfect square natural numbers. These are the natural numbers of the form m^2 for some natural number m. So the first few are 0, 1, 4, 9, 16 and so on. We would be right to define this set by $\{n \in \mathbb{N} \mid \text{for some } m \in \mathbb{N},\ m^2 = n\}$. To make sense of this, we need to understand what "for some $m \in \mathbb{N}, \ldots$" means formally. Just as we abbreviated "and" with symbol $\wedge$ and "or" with $\vee$, we will abbreviate "for some $m \in \mathbb{N}, \ldots$" as $\exists m \in \mathbb{N}, \ldots$.

Lemma 9.3

For sets W and U and relation $R \subseteq W \times U$, there is a subset of U, denoted by $\{x \in U \mid \exists w \in W. w \mathrel{R} x\}$ so that for all $C \subseteq U$,

$$\{x \in U \mid \exists w \in W. w \mathrel{R} x\} \subseteq C \text{ if and only if } R \subseteq W \times C.$$

Also, there is a subset of U, denoted by $\{x \in U \mid \forall w \in W, w \mathrel{R} x\}$ so that for all $C \subseteq U$,

$$C \subseteq \{x \in U \mid \forall w \in W, w \mathrel{R} x\} \text{ if and only if } W \times C \subseteq R.$$

Proof: The relation R determines a function r from U to $\mathcal{P}(W)$ by $x \mapsto \{w \in W \mid w \mathrel{R} x\}$. Then

$$\{x \in U \mid \exists w \in W, w \mathrel{R} x\} = r^{-}(\mathcal{P}(W) \setminus \{\emptyset\})$$
$$\{x \in U \mid \forall w \in W, w \mathrel{R} x\} = r^{-}(\{W\})$$

Proving that these sets satisify the desired conditions is technical, but routine.

Concretely, $\{x \in U \mid \exists w \in W, w \mathrel{R} x\}$ consists of those $x \in U$ so that $w \mathrel{R} x$ for some $w \in W$; $\{x \in U \mid \forall w \in W, w \mathrel{R} x\}$ consists of those $x \in U$ so that $w \mathrel{R} x$ for all $w \in W$. This justifies our notation: $\exists w \in W, w \mathrel{R} x$ is read as "there *exists* $w \in W$ so that $w \mathrel{R} x$;" $\forall w \in W, w \mathrel{R} x$" is read as "for *all* $w \in W$, $w \mathrel{R} x$." □

Expressions using $\exists$ and $\forall$ can be nested, so for example suppose we wish to define the set of all functions $\mathbb{R} \to \mathbb{R}$ that are continuous at a. We can write

$$\{f \in \mathbb{R}^{\mathbb{R}} \mid \forall \epsilon \in \mathbb{R}^{+} \exists \delta \in \mathbb{R}^{+}, B(a, \delta) \subseteq f^{-}(B(f(a), \epsilon))\}$$

where $\mathbb{R}^{+} = \{x \in \mathbb{R} \mid 0 < x\}$ – the set of positive real numbers – and $B(b, \gamma) = \{x \in \mathbb{R} \mid b - \gamma < x < b + \gamma\}$ – the set of real numbers in the interval $(b - \gamma, b + \gamma)$. So a function f belong to this set if and only if for every $\epsilon > 0$ there is a $\delta > 0$ so that for all $x \in \mathbb{R}$, if $a - \delta < x < a + \delta$ then $f(a) - \epsilon < f(x) < f(a) + \epsilon$. The reader who is familiar with Calculus will recognize that this is the precise definition of *continuity at* a.

We can use $\exists$ and $\forall$ to generalize union and intersection to arbitrary collections.

Definition 9.3

Let $A \colon I \to \mathcal{P}(U)$ be a function into the powerset of U. We write A_i instead of $A(i)$ to emphasize that each A_i is a set. Then define

$$\bigcup_{i \in I} A_i = \{x \in X \mid \exists i \in I . x \in A_i\}$$

According to Lemma **??**, $\bigcup_{i \in I} A_i$ is again a subset of U, generalizing $\cup$ to the union of arbitrary families of subsets of U, rather than just two. In this sense, $\mathcal{P}(U)$ is *complete*. That is, the union of any family of subsets of U exists. This justifes saying that $\mathcal{P}(U)$ is a *complete* Boolean algebra.

Exercises for Lecture 9

In the following, consider the family $A \colon I \to \mathcal{P}(U)$.

1. Show that $\bigcup_{i \in I} A_i \subseteq C$ if and only if $A_k \subseteq C$ holds every $k \in I$.
2. Define $\bigcap_{i \in I} A_i$ in analogy with $\bigcup_{i \in I} A_i$.
3. For $I = \emptyset$, what is $\bigcup_{i \in I} A_i$?
4. For $I = \emptyset$, what is $\bigcap_{i \in I} A_i$?

9.5 Atomicity

The complete Boolean algebras $\mathcal{P}(U)$ have one more feature that characterizes powersets. They are *atomic*. This means, roughly, that all subsets are built from the simplest ones.

An *atom* of a Boolean algebra is an element with the property that there is nothing strictly between the smallest element and it. A singleton subset $\{x\} \subseteq U$ is an atom of $\mathcal{P}(U)$ because there are no other subsets lying between $\emptyset$ and $\{x\}$

Lemma 9.4

For any set U, the rule $x \mapsto \{x\}$ determines a function from U to $\mathcal{P}(U)$.

Proof: Recall that the *diagonal* relation on U is $\Delta_U = \{(x, y) \in U \times U \mid x = y\}$. Let $s \colon U \to \mathcal{P}(U)$ be the unique function for which $\in_U \circ (s \circ \mathsf{id}_U) = \kappa_{\Delta_U}$. In other words $\ni_U(s(x), y) = \top$ if and only if $x = y$. Since $\ni_U(s(x), y) = \top$ if and only if $y \in s(x)$, it is the case that $y \in s(x)$ if and only if $x = y$. So $s(x) = \{x\}$. □

Now every subset of U is obtained as a union of singletons: $A = \bigcup_{x \in A}\{x\}$. For a complete Boolean algebra, this is what is meant by saying that $\mathcal{P}(U)$ is a complete *atomic* Boolean algebra. Although we do not investigate this here, any complete atomic Boolean algebra has the same structure as $\mathcal{P}(U)$ for some U.

The structure of $\mathcal{P}(U)$ is even preserved by inverse images.

Lemma 9.5

For any function $f \colon X \to Y$, any $B \colon I \to \mathcal{P}(Y)$, it is the case that $f^{-}(\bigcup_{i \in I} B_i) = \bigcup_{i \in I} f^{-1}(B_i)$ and $f^{-1}(\bigcap_{i \in I} B_i) = \bigcap_{i \in I} f^{-1}(B_i)$.

Proof: $x \in f^{-}(\bigcup_{i \in I} B_i)$ if and only if $f(x) \in \bigcup_{i \in I} B_i$ if and only if $f(x) \in B_k$ for some $k \in I$ if and only if $x \in f^{-}(B_k)$ for some $k \in I$ if and only if $x \in \bigcup_{i \in I} f^{-}(B_i)$. The proof for $\bigcap$ is similar with “for some …” replaced by “for all …". □

Exercises for Lecture 9

1. Write out $\mathcal{P}(\{a, b, c\})$
2. Write out $\mathcal{P}(\emptyset)$
3. Is it the case that $\emptyset \in \mathcal{P}(A)$ for any set A? Explain.
4. Write out $\mathcal{P}(2 \times 2)$ and $\mathcal{P}(\mathcal{P}(2))$. Pay attention to writing them in a systematic way, so that it is clear you have actually listed everything.
5. I claim that $\mathcal{P}(\emptyset)$ is a terminal set (Definition 7.2). Justify the claim.
6. I claim that $\mathcal{P}(\emptyset) \in \mathcal{P}(\mathcal{P}(\emptyset))$ is a subset classifier (Definition 7.7). Justify the claim.

9.6 Forward images

For a function $f \colon A \to B$ the function $f^{-} \colon \mathcal{P}(C) \to \mathcal{P}(B)$ has what is known as an *upper adjoint*.

Definition 9.4

For $f\colon X \longrightarrow Y$, define $f^{+}\colon \mathcal{P}(X) \longrightarrow \mathcal{P}(Y)$ by the rule $A \mapsto \{y \in B \mid \exists x \in A.f(x) = y\}$. The subset $f^{+}(A) \subseteq Y$ is called the *forward image of* A *with respect to* f.

The important fact about f^{+} is that is related to f^{-}.

Lemma 9.6

For any $A \subseteq X$ and $B \subseteq Y$, $f^{+}(A) \subseteq B$ if and only if $A \subseteq f^{-}(B)$.

Proof: Suppose $f^{+}(A) \subseteq B$. For $x \in A$, $f(x) \in f^{+}(A)$. So $f(x) \in B$. By definition, this means $x \in f^{-}(B)$. This shows that $A \subseteq f^{-}(B)$. Conversely, suppose $A \subseteq f^{-}(B)$. For $y \in f^{+}(A)$, there is some $x \in A$ so that $f(x) = y$. So there is some $x \in f^{-}(B)$ so that $f(x) = y$. Thus $y = f(x) \in B$. This shows that $f^{+}(A) \subseteq B$ □

There is also a lower adjoint of f^{-}, but as it is less commonly used, we do not investigate it here.

The important features of f^{+} are easily checked. First, f^{+} preserves atoms. That is, $f^{+}(\{x\}) = \{f(x)\}$. Second, f^{+} preserves all unions So $f^{+}(\bigcup_{i \in I} A_i) = \bigcup_{i \in I} f^{+}(A_i)$. Note that f^{+} does not necessarily preserve intersections.

Exercises for Lecture 9

1. Define a function $f\colon X \longrightarrow Y$ and two subsets $A, B \subseteq X$ so that $f(X) \cap f(Y) \neq f(X \cap Y)$. Try to find the smallest example you can.

2. Prove that for any function $f\colon X \longrightarrow Y$ and any $A \subseteq X$, the inclusion $A \subseteq f^{-}(f^{+}(A))$ holds.

3. Prove that for any function $f\colon X \longrightarrow Y$ and any $B \subseteq Y$, the inclusion $f^{+}(f^{-}(B)) \subseteq B$ holds.

LECTURE 10

Additional Constructions

Goals

Many other constructions can be built up using the principles we have discussed. In this lecture, we consider some of the most useful and most general.

10.1 Unions

Although not strictly needed for most purposes, mathematicians generally agree that sets, no matter how they are related, can be merged into a single set with elements taken from the originals. That is, union ($\bigcup$) is meaningful for any set of sets. We take this as an additional principle.

Principle 10.1

Suppose $\mathcal{X}$ is a set and each element of $\mathcal{X}$ is a set (so $\mathcal{X}$ is a set of sets). Then there is a set U so that $\mathcal{X} \subseteq \mathcal{P}(U)$ and for any set Z, if $\mathcal{X} \subseteq \mathcal{P}(Z)$, then $U \subseteq Z$.

Using this principle, the set U is actually the union $\bigcup_{X \in \mathcal{X}} X$. That is, $x \in \bigcup_{X \in \mathcal{X}} X$ holds if any only if $x \in X$ for some $X \in \mathcal{X}$.

10.2 Monomorphisms and Epimorphisms

As we know, the fact that addition is cancellative ($m + p = n + p$ implies $m = n$) is quite useful. Likewise, multiplication is amost always cancellative ($m \cdot p = n \cdot p$ implies $m = n$ provided $p \neq 0$). Even list concatenation is cancellative. But function composition is not. At least not generally. It will turn out though that certain functions can indeed by cancelled on the left and others can be cancelled on the right. The role of these special functions is akin to the role that non-0's play in multiplication.

Definition 10.1

A function $f\colon X \to Y$ is called

- a *monomorphism* if it is the case that for every $W \underset{k}{\overset{h}{\rightrightarrows}} X$, if $f \circ h = f \circ k$ then $h = k$;
- an *epimorphism* if it is the case that for every $Y \underset{k}{\overset{h}{\rightrightarrows}} Z$, if $h \circ f = k \circ f$ then $h = k$.

The terms *mono-* and *epi-* refer to behavior that we explore in this Lecture. For now, just commit them to memory: monomorphisms *cancel on the left* of $\circ$, epimorphisms *cancel on the right*.

Example 10.1

- For any subset $A \subseteq X$, the inclusion function $A \xrightarrow{i} X$ is a monomorphism.
- For sets X and Y, the projection functions $X \times Y \xrightarrow{\pi_0} X$ and $X \times Y \xrightarrow{\pi_1} Y$ are epimorphisms.
- The functions $X \xrightarrow{\diamond_X} \mathbb{1}$ are almost always epimorphisms. The only except is when $X = \emptyset$.
- All pointers $\mathbb{1} \xrightarrow{\hat{a}} X$ for $a \in X$ are monomorphisms.
- For any set X, id_X is both a monomorphism and an epimorphism.

Exercises for Lecture 10

1. Show that if $X \xrightarrow{f} Y$ and $Y \xrightarrow{g} Z$ are epimorphisms, then so is $g \circ f$.
2. Show that if $X \xrightarrow{f} Y$ and $Y \xrightarrow{g} Z$ are monomorphisms, then so is $g \circ f$.
3. Show that if $X \xrightarrow{f} Y$ and $X \xrightarrow{Y}$ satisfy $g \circ f = \mathsf{id}_X$, then g is an epimorphism and f is a monomorphism.

The two lemmas spell out equivalent internal behavior.

Definition 10.2

A function $X \xrightarrow{f} Y$ is called

- *one-to-one* or *injective* or *an injection* if it is the case that for every $b \in Y$ there is at most one $a \in X$ so that $f(a) = b$;
- *onto* or *surjective* or *a surjection* if it is the case that for every $b \in Y$ there is at least one $a \in X$ so that $f(a) = b$.

The property of being injective can also be stated by saying that if $f(a_0) = f(a_1)$ then $a_0 = a_1$.

Lemma 10.1

A function is a monomorphism if and only if it is an injection.

Proof: Assume $X \xrightarrow{m} Y$ is a monomorphism. Consider $a_0, a_1 \in X$ so that $m(a_0) = m(a_1)$. Then $m \circ \widehat{a_0} = m \circ \widehat{a_1}$. Since m is a monomorphism $\widehat{a_0} = \widehat{a_1}$. Hence $a_0 = a_1$.

Assume $X \xrightarrow{m} Y$ is not a monomorphism. So there must be some set W and functions $W \overset{h}{\underset{k}{\rightrightarrows}} X$ so that $m \circ h = m \circ k$ but $h \neq k$. Since $h \neq k$, there must be an element $w \in W$ so that $h(w) \neq k(w)$. Let $a_0 = h(w)$ and $a_1 = k(w)$. These witness that m is not an injection because $m(a_0) = m(a_1)$ but $a_0 \neq a_1$. □

Lemma 10.2

A function is an epimorphism if and only if it is a surjection.

Proof: Assume $X \xrightarrow{e} Y$ is not onto. So there is some $b \in Y$ for which $e(x) \neq y$ is the case for all $x \in X$. Define $Y \overset{h}{\underset{k}{\rightrightarrows}} \{0, 1, 2\}$ by

$$h(y) = \begin{cases} 0 & \text{if } y = b \\ 2 & \text{otherwise} \end{cases}$$

$$k(y) = \begin{cases} 1 & \text{if } y = b \\ 2 & \text{otherwise.} \end{cases}$$

It is easy to verify that $h \circ e = k \circ e$, but obviously $h \neq k$. So e is not an epimorphism.

Assume $X \xrightarrow{e} Y$ is onto. Suppose $Y \overset{h}{\underset{k}{\rightrightarrows}} Z$ satisfy $h \circ e = k \circ e$. For any $y \in Y$, there is some $x \in X$ so that $e(x) = y$. So $h(y) = h(e(x)) = k(e(x)) = k(y)$. By function extensionality, $h = k$. □

Subsets permit us to concentrate attention on elements with special properties. For an informal example, let $[-1, 1] \subseteq \mathbb{R}$ be the set of real numbers satisfying $-1 \leq x \leq 1$. Then the trigonometric function $\sin\colon \mathbb{R} \to \mathbb{R}$ has the property that $\sin(x) \in [-1, 1]$ is true for every $x \in \mathbb{R}$. So this suggests that sin can be "co-restricted" to the smaller codomain of $[-1, 1]$. In fact, Principle 7.5 guarantees this. We need, however, to generalize the idea to situations not covered by the principle as stated.

Consider the two functions depicted in Figure **??**. The function m is a monomorphism. The two functions are related by having the same codomain. Moreover, the elements of the codomain that are reached by the function f are also reached by m ($f(x) = m(d)$, $f(y) = m(a)$ and $f(z) = m(a)$). So it makes sense to define a function as indicated in 10.2. The situations in which this should be possible can be characterized internally via the behavior of functions on elements, or externally via their behavior with other functions.

Monomorphisms act like subsets, in the sense that if $A \subseteq X$, then the inclusion function $A \xrightarrow{i} X$ is a monomorphism. Our next principle stipulates that monomorphisms behave like subsets, also in the sense that they determine characteristic functions.

Principle 10.2

For a monomorphism $X \xrightarrow{m} Y$ there is a unique function $Y \xrightarrow{\kappa_m} \mathbb{2}$ so that m is an equalizer for the equation $\kappa_m(y) = \mathrm{T}$. In other words, $\kappa_m \circ m = c_{\mathrm{T}}$ and if $W \xrightarrow{f} Y$ is a function satisfying $\kappa_m \circ f = c_{\mathrm{T}}$, then there is a unique function $W \xrightarrow{m \backslash f} X$ so that $f = (m \backslash f) \circ m$. We say that f *factors through* m *uniquely*.

For a monomorphism $X \xrightarrow{m} Y$, the characteristic function κ_m is defined by

$$\kappa_m(y) = \begin{cases} \mathrm{T} & \text{if } y = m(x) \text{ for some } x \in X \\ \mathrm{F} & \text{otherwise.} \end{cases}$$

Suppose $W \xrightarrow{f} Y$ is a function satisfying $\kappa_m \circ f = c_{\mathrm{T}}$. Then $(m \backslash f)$ satisfies $(m \backslash f)(w) = x$ if and only if $f(w) = m(x)$.

In Figure 10.1, the function f satisfies $\kappa_m \circ f = c_{\mathrm{T}}$, so there is indeed a unique function $(m \backslash f)$ as indicated in Figure 10.2.

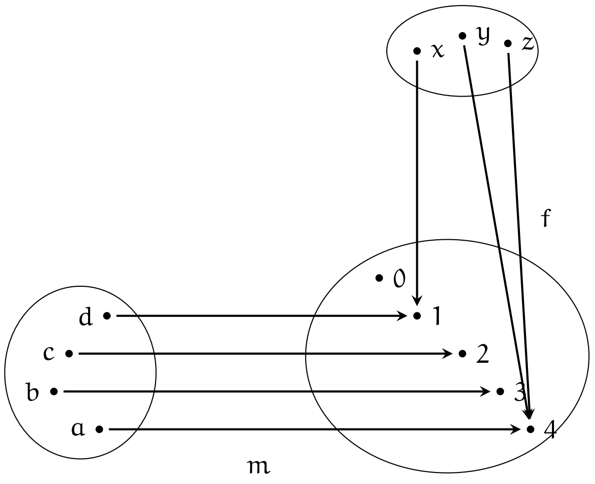


Figure 10.1: Function f and m with a common codomain

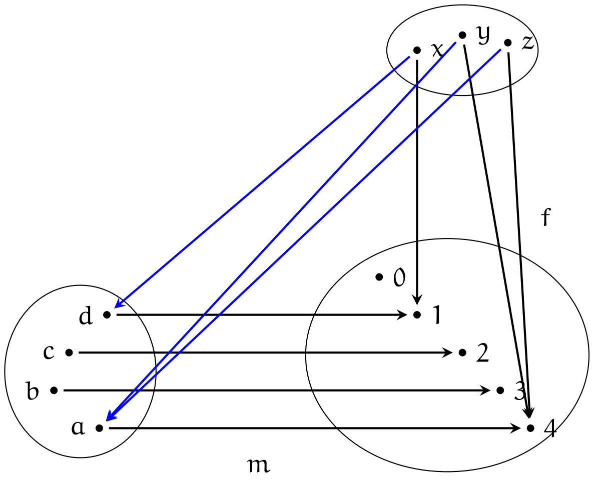


Figure 10.2: Function f *factors uniquely through* m

To detect that f is a solution of the equation $\kappa_m(y) = \top$ is a simple matter. An internal characterization is this: for every $w \in W$, there is some $x \in X$ so that $f(w) = m(x)$. An external characterization is also useful.

Definition 10.3

For functions $X \xrightarrow{m} Y \xleftarrow{f} W$, say that f is m-coinvariant if it is that case that for all $Y \overset{h}{\underset{k}{\rightrightarrows}} Z$, $h \circ m = k \circ m$ implies $k \circ f = h \circ f$.

Lemma 10.3

For function $X \xrightarrow{m} Y \xleftarrow{f} W$ where m is a monomorphism, f is m-coinvariant if and only if $\kappa_m \circ f = c_\top$.

Proof: First, note that $\kappa_m \circ f = c_\top$ if and only if $\kappa_m(f(w)) = \top$ for every $w \in W$. But $\kappa_m(y) = \top$

if and only if there is some $x \in X$ so that $y = m(x)$. Therefore, $\kappa_m \circ f = \tau$ if and only if for every $w \in W$ there exists some $x \in X$ for which $f(w) = m(x)$.

Assume $\kappa_m \circ f = c_\top$, and $h \circ m = k \circ m$. For any $w \in W$, there is some $x \in X$ so that $f(w) = m(x)$. Hence $h(f(w)) = h(m(x)) = k(m(x)) = k(f(w))$ for a suitable choice of x. Since this works for any w, $h \circ f = k \circ f$.

Assume there is some $w \in W$ for which $f(w) \neq m(x)$ for all $x \in X$. Define functions $Y \overset{h}{\underset{k}{\rightrightarrows}} \{0, 1, 2\}$ by

$$h(y) = \begin{cases} 0 & \text{if } y = f(w) \\ 2 & \text{otherwise} \end{cases}$$

$$k(y) = \begin{cases} 1 & \text{if } y = f(w) \\ 2 & \text{otherwise.} \end{cases}$$

It is easy to verify that $h \circ m = k \circ m$, but $h \circ f \neq k \circ k$. So f is not m-coinvariant. □

Lemma 10.4

For functions $W \xrightarrow{m} Y \xleftarrow{f} X$ where m is an monomorphism and f is m-coinvariant, there is a unique function $X \xrightarrow{m\backslash f} W$ so that $f = m \circ (m\backslash f)$.

Proof: There is nothing new to prove. Lemma **??** ensures if f is m-coinvariant, then $\kappa_m \circ f = c_\top$. Principle **??** ensures that $m\backslash f$ exists. □

Epimorphisms are dual to monomorphisms (they cancel on the left instead of on the right). The concept of m-coinvariance also has a useful dual.

Definition 10.4

For functions $W \xleftarrow{f} Y \xrightarrow{e} X$, say that f is e-coinvariant if it is that case that for all $Z \overset{h}{\underset{k}{\rightrightarrows}} Y$, $e \circ h = e \circ k$ implies $f \circ h = f \circ k$.

Lemma 10.5

For functions $W \xleftarrow{f} Y \xrightarrow{e} X$ where e is an epimorphism and f is e-invariant, there is a unique function $X \xrightarrow{f/e} W$ so that $f = (f/e) \circ e$.

Proof: A full proof is technical and not especially informative. The idea is that f/e can be defined by a "rule": $e(y) \mapsto f(y)$. Obviously, this does not look like a rule sending elements of X to elements of W, but in fact it is. This is because e is onto, each $x \in X$ takes the form $e(y)$ for some $y \in Y$. And because f is e-invariant, the choice of y does not matter. That is, $e(y_0) = x$ and $e(y_1) = x$ implies $f(y_0) = f(y_1)$. So the rule does indeed pick a single value $(f/e)(x)$ for each $x \in X$. □

Comparing Lemmas 10.4 and 10.5, we note that monomorphisms and epimorphisms play dual roles. Also a monomorphism behaves in most respects like the inclusion of a subset. So a question arises as to what should be the dual concept corresponding to subset. That is, what completes the formal analogy "Subsets are to monomorphisms as ___s are to epimorphisms".

Definition 10.5

For a set X, a *partition of* X is a subset $P \subseteq \mathcal{P}(X)$ so that

- for each $B \in P$, there is at least one $x \in X$ so that $x \in B$;
- for each $x \in X$, there is exactly one $B \in P$ so that $x \in B$.

The subsets $B \in P$ are called *blocks*.

Because each x is in exactly one block, we write $[x]_P$ for that block. This means that $x \mapsto [x]_P$ defines a function $X \xrightarrow{[-]} P$. Furthermore, since each $B \in P$ is inhabited ($x \in B$ for some x), the function $[-]_P$ is clearly onto. Suppose $e \colon X \to Y$ is an epimorphism. Then the collection $\{e^{-}(y) \mid y \in Y\}$ is a partition of X. That is, each $e^{-}(y)$ is non-empty because e is onto. Clearly, $x \in e^{-}(e(x))$. And if $x \in e^{-}(y)$, then $e(x) = y$, so x belongs only to one block. Let X/e denote the partition $\{e^{-}(y) \mid y \in Y\}$. Then $x \mapsto [x]_{X/e}$ simply picks out the set of all x' so that $e(x) = e(x')$. Putting these observations together, we see that every epimorphism e with domain X determines X/e, a partition of X. And every partition P of X determines an epimorphism $x \mapsto [x]_P$ of X onto P.

A monomorphism also determines characteristic function: $A \xrightarrow{m} X$ determines $X \xrightarrow{\kappa_m} 2$. What is the analogue of a characteristic function for epimorphisms? The answer is an *equivalence* relation.

For a function $f \colon X \to Y$ (in practice, usually we expect f to be an epimorphism), define $\equiv_f \subseteq X \times X$ by $x_0 \equiv_f x_1$ if and only if $f(x_0) = f(x_1)$. This relation has three characteristic properties:

- $\equiv_f$ is *reflexive*: $x \equiv_f x$ for all $x \in X$;
- $\equiv_f$ is *transitive*: if $x_0 \equiv_f x_1$ and $x_1 \equiv_f x_2$, then $x_0 \equiv_f x_1$; and
- $\equiv_F$ is *symmetric*: if $x_0 \equiv_f x_1$, then $x_1 \equiv_f x_0$.

Definition 10.6

A relation E on X is an *equivalence relation on* X if it is reflexive, transitive and symmetric.

Lemma 10.6

Suppose E is an equivalence relation on X. Then there is a set X/E and epimorphism $[-] \colon X \to X/E$ so that $E = \equiv_{[-]}$.

Proof: For each $x \in X$, let $[x]_E = \{x' \in X \mid x \mathrel{E} x'\}$. Since E is reflexive, $x \in [x]_E$ for each $x \in X$. Also, if $[x]_E \cap [x']_E \neq \emptyset$, then $x \mathrel{E} x'$ because E is transitive and symmetric. So the collection $\{[x]_E \mid x \in X\}$ is a partition of X. The function $x \mapsto [x]_E$ is an epimorphism by construction. And evidently, $[x]_E = [x']_E$ holds if and only if $x \mathrel{E} x'$. So $E = \equiv_{[-]}$. □

A function $f \colon X \to Y$ is invariant with respect to $[-]_E$ if it is the case that $x_0 \mathrel{E} x_1$ implies $f(x_0) = f(x_1)$. Any function with this property determines a function from X/E to Y defined by $[x]_E \mapsto f(x)$. This is defined on all blocks of X/E because each block is $[x]_E$ for some x. Moreover, if $[x_0]_E = [x_1]_E$, then $x_0 \mathrel{E} x_1$. So $f(x_0) = f(x_1)$. That is, our rule is unambiguous.

Example 10.2

We might wish to define a 'clock'. Recall that $\mathbb{Z}$ denotes the set of all integers. On a clock 1 and 13 are the same because one hour after midnight and thirteen hours after midnight read the same on the clock. In fact, 1, 13, 25, and so on all read the same. So let is define the relation $\equiv_{12}$ on $\mathbb{Z}$ by saying $a \equiv_{12} b$ if and only if for some integer m, it is the case that $a + 12 \cdot m = b$. In other words, a and b differ by some even multiple of twelve.

Clearly, $a \equiv_{12} a$ is true for all integers a (take $m = 0$). And if $a \equiv_{12} b$ and $b \equiv_{12} c$, then $a + 12 \cdot m = b$ and $b + 12 \cdot n = c$ for some m and n. Hence $a + 12 \cdot (m + n) = c$. So $a \equiv_{12} c$. And finally, if $a \equiv_{12} b$, then $a + 12 \cdot m = b$, so $b + 12 \cdot -m = a$. So $b \equiv_{12} a$. This shows that $\equiv_{12}$ is an equivalence relation. Now, $\mathbb{Z}/\equiv_{12}$ consists of exactly twelve equivalence blocks: $[0]_{\equiv_{12}}, \ldots, [11]_{\equiv_{12}}$. Each block corresponds to an hour on the clock dial.

The quotient set $\mathbb{Z}/\equiv_{12}$ is usually denoted by $\mathbb{Z}_{12}$. Clearly, the same idea works for any positive integer *modulus* k in place of 12. We investigate this idea in depth in Lecture **??**, where applications in cryptography arise.

10.3 Co-products

Definition 10.7

For sets X and Y, a *co-table* is a set with a pair of functions $X \xrightarrow{f} C \xleftarrow{g} Y$.

A *co-product* is a co-table $X \xrightarrow{i} S \xleftarrow{j} Y$ so that for any co-table $X \xrightarrow{f} C \xleftarrow{g} Y$ there is exactly one function $c\colon S \to C$ so that $f = c \circ i$ and $g = c \circ j$.

Lemma 10.7

Any two sets X and Y have a co-product.

Proof: Define $X \uplus Y \subseteq \mathcal{P}(X) \times \mathcal{P}(Y)$ to consist only of pairs of the form $(\{x\}, \emptyset)$ for $x \in X$ and $(\emptyset, \{y\})$ for $y \in Y$.

Define the two functions $\mathsf{inj}_0\colon X \to X \uplus Y$ and $\mathsf{inj}_1\colon Y \to X \uplus Y$ by $\mathsf{inj}_0(x) = (\{x\}, \emptyset)$ and $\mathsf{inj}_1(y) = (\emptyset, \{y\})$. Now, for any pair of functions $X \xrightarrow{f} C \xleftarrow{g} Y$ define $[f, g]'\colon X \uplus Y \to \mathcal{P}(C)$ by the rule $(A, B) \mapsto f^+(A) \cup g^+(B)$. Since either $A = \emptyset$ and B is a singleton or A is a singleton and $B = \emptyset$, it is always the case that $[f, g]'(A, B)$ is a singleton. So it factors uniquely through the function sending $c \in C$ to $\{c\}$. That is, there is a unique function $[f, g]\colon X \uplus Y \to C$ so that $\{[f, g](A, B)\} = [f, g]'(A, B)$. Now it is easily checked that $[f, g] \circ \mathsf{inj}_0 = f$ and $[f, g] \circ \mathsf{inj}_1 = g$. And no other function has this property. □

The set $X \uplus Y$ is sometimes called the *disjoint union* of X and Y. It is a union, preceded by "marking" each $x \in X$ by putting it into a pair $(\{x\}, \emptyset)$ and marking each $y \in Y$ differently by putting it into a pair $(\emptyset, \{y\})$. Hence it is the union of disjoint copies of X and Y.

Exercises for Lecture 10

Let $A = \{a, b, c, d, e\}$, $B = \{w, x, y, z, a, b, c, \}$ and $C = \{2, 3\}$

1. Calculate $A \uplus B$

2. Calculate $A \uplus \emptyset$

3. Calculate $C \times (A \uplus B)$ and $(C \times A) \uplus (C \times B)$.

10.4 Quotients

Sometimes, elements of a set X will be classified into "like" kinds. For example, we might classify the natural numbers into even and odd. We might classify poker cards according to suit, ignoring the rank – or by rank, ignoring suit. Or, if C is set modelling a Discrete Mathematics class, we might classify the elements (students) according to their grade: A, B, etc. Situations like this are modelled by what is known as a *partion* of a set. If we also wish to think of all A students as being "equivalent", all B students as being "equivalent", we model this by what is known as an *equivalence relation*.

Definition 10.8

For a set X, a *partition of* X is a set $P \subseteq \mathcal{P}(X) \setminus \{\emptyset\}$ so that $\bigcup_{A \in P} A = X$ and for every $A, B \in P$, if $A \neq B$ then $A \cap B = \emptyset$. The sets in P are called *blocks*.

For a set X, an *equivalence relation on* X is a binary relation satisfying

- $x \mathrel{E} x$ for all $x \in X$,
- $x \mathrel{E} y$ and $y \mathrel{E} z$ implies $z \mathrel{E} z$ for all $x, y, z \in X$, and
- $x \mathrel{E} y$ imples $y \mathrel{E} x$

For a partition P define the relation $\equiv_P \subseteq X \times X$ by $x \equiv_P y$ if and only if for some $A \in P$, $x \in A$ and $y \in A$.

For an equivalence relation E and an element $x \in X$, let $[x]_E = \{y \in X \mid x \mathrel{E} y\}$. So $x \mapsto [x]_E$ defines a function from X to $\mathcal{P}(X)$. Let $X/E = \{[x]_E \mid x \in X\}$. That is, X/E is the range of the function $x \mapsto [x]_E$.

The two notions, partition and equivalence relation, are essentially the same in the sense that there is a natural way to pass from a partitions to an equivalence relation and vice versa.

Lemma 10.8

Let X be any set. Then

- For any partition P of X, the relation $\equiv_P$ is an equivalence relation on X;
- For any equivalence relation E on X, the collection $\mathcal{P}_E$ forms a partition of X;
- For any partition P of X, $P = X/{\equiv_P}$
- For any equivalence relation E on X, $E = \equiv_{X/E}$;
- the rule $x \mapsto [x]_E$ determines an onto function from X to X/E.

Proof: Exercise. □

Suppose $E \subseteq X \times X$ is an equivalance relation and $f\colon X \longrightarrow Y$ is a function so that $x \mathrel{E} x'$ implies $f(x) = f(x')$. Then we can define a function from X/E to Y by the "rule" $[x]_E \mapsto f(x)$. We can not call this a rule in the usual way because the left side is not a variable or a tuple of variables. But we can define a relation $F \subseteq (X/E) \times Y$ by stipulating that for $B \in X/E$ and $y \in Y$, $B \mathrel{F} y$ if and only if $f(x) = y$ for some $x \in B$. This relation is total because every block $B \in X/E$ is non-empty. So there is some $x \in B$, and thus $B\ F\ f(x)$. The relation F is also deterministic because if $B\ F\ y$ and $B\ F\ y'$, then there is some $x \in B$ for which $f(x) = y$ and there is some $x' \in B$ for which $f(x') = y'$. But $x, x' \in B$ implies $x\ E\ x'$. Hence $f(x) = f(x')$.

Definition 10.9

Suppose $E \subseteq X \times X$ is an equivalence relation. A function $f\colon X \longrightarrow Y$ is E-*invariant* if $x\ E\ x'$ implies $f(x) = f(x')$. For an E-invariant function $f\colon X \longrightarrow Y$, let f/E denote the function from X/E to Y defined by $(f/E)([x]_E) = f(x)$.

Example 10.3

Let $E \subseteq \mathbb{R} \times \mathbb{R}$ be the relation xEy if and only if $x - y = k2\pi$ for some integer k. This is an equivalence relation: xEx is true because $x - x = 0\dot{2}\pi$. If $x - y = k2\pi$ then $y - x = -k2\pi$, so E is symmetric. And if $x - y = k2\pi$ and $y - z = j2\pi$, then $x - z = (x - y) + (y - z) = (k + j)2\pi$. So E is transitive. The functions sin and cos are E invariant because $\sin(x + k2pi) = \sin(x)$ and $\cos(x + k2\pi) = \cos(x)$ for any x and any k.

Exercises for Lecture 10

On the integers $\mathbb{Z}$, define a relation $\equiv_7$ by $i \equiv_7 j$ if and only if there is some integer k so that $i + 7k = j$.

1. Show that $\equiv_7$ is an equivalence relation.
2. Describe the set $[5]_{\equiv_7}$.
3. Show that the function $f(n) = n + 3$ is $\equiv_7$-invariant.
4. Show that the function $g(n) = 2n$ is $\equiv_7$-invariant.
5. Determine whether the function $h(n) = 2^n$ is $\equiv_7$-invariant.

10.5 Function Graphs

Suppose S is a set modelling the students at Chapman and M is a set modelling the academic majors the university offers: Mathematics, Philosophy, HeadScratching, and so on. Then elements of S (students) can be related to elements of M (majors) by 'is majoring in' as in "Jethro is majoring in Phrenology." Because a student might have a double major, we can not model the situation as a function, at least not in the most obvious way. Instead, we introduce the notion of a *(binary) relation*. The same idea, generalized to higher dimensional relations, is at the heart of what we call *relational databases*.

Definition 10.10

A *binary relation from* X *to* Y is a subset $R \subseteq X \times Y$. A *relation on* X is a binary relation from X to X. Since we will only be concerned with binary relations, from now on we refer them simply as *relations*

For a relation R from X to Y, we will say "x is R-related to y" and write $R(x, y)$ when $(x, y) \in R$. In many situations, we use "infix" notation, writing $x\ R\ y$ instead of $R(x, y)$.

Note that there are other equivalent ways to think about relations.

- $R \subseteq X \times Y$ determines a characteristic function $\kappa_R \colon X \times Y \to 2$ so that $R = \{(x, y) \in X \times Y \mid \kappa_R(x, y) = \mathrm{T}\}$
- R determines a function $R[-] \colon X \to \mathcal{P}(Y)$ so that $R = \{(x, y) \in X \times Y \mid y \in R[x]\}$.

The ability to move between these can be helpful. So it is worth practicing.

Exercises for Lecture 10

Let $r \colon \mathbb{N} \times \mathbb{N} \to 2$ be the function defined by

$$r(m, n) = \begin{cases} \mathrm{T} & \text{if } m \geq n^2 \\ \mathrm{F} & \text{otherwise.} \end{cases}$$

Consider the relation $R = r^{-}(\mathrm{T})$.

1. Is it the case that $3 \; R \; 2$?
2. Is it the case that $9 \; R \; 3$?
3. What is $R[5]$?
4. What is $\mathsf{curry}[r]$?

Like functions, relations allow a kind of composition and every set has an identity relation.

Definition 10.11

Suppose $R \subseteq X \times Y$ and $S \subseteq Y \times Z$ are relations. Define $R; S \subseteq X \times Z$ by $x \; R; S \; z$ if and only if there is some $y \in Y$ so that $x \; R \; y$ and $y \; S \; z$. For set X, define $\Delta_X = \{(x, y) \in X \times X \mid x = y\}$.

It will be useful to know that this form of composition behaves similar to function composition.

Lemma 10.9

For a relation $R \subseteq X \times Y$,

$$\Delta_X; R = R = R; \Delta_Y.$$

For relations $R \subseteq W \times X$, $S \subseteq X \times Y$ and $T \subseteq Y \times Z$,

$$R; (S; T) = (R; S); T.$$

Proof: Exercise. □

Definition 10.12

A function $f \colon X \to Y$ determines a relation called the *graph of* f defined by $\Gamma_f = \{(x, y) \in X \times Y \mid f(x) = y\}$.

Note: Γ_f is an equalizer $f \circ \pi_0$ and π_1.

The composition of functions and identity functions are essentially the same as composition of graphs and identity relations.

Lemma 10.10

For functions $X \xrightarrow{f} Y$ and $Y \xrightarrow{g} Z$, $\Gamma_{g \circ f} = \Gamma_f ; \Gamma_g$. For a set X,

$$\mathsf{id}_X = \Gamma_{\Delta_X}$$

.

Proof: Exercise. □

This lemma tells us that Γ is a *functor*. This is an operation that, in some suitable sense, preserves composition and identities. You will encounter functors in further studies. We don't need to investigate the general idea now.

From these simple observations, we might ask which relations arise as graphs of functions. The obvious conditions to consider are these:

Definition 10.13

A relation $R \subseteq X \times Y$ is

- *total* if for every $x \in X$, there is at least one $y \in Y$ for which $x \mathrel{R} y$;
- *deterministic* if for every $x \in X$, there is at most one $y \in Y$ for which $x \mathrel{R} y$;
- *functional* if it is total and deterministic.

Lemma 10.11

For any function $X \xrightarrow{f} Y$, the relation $\Gamma_f \subseteq X \times Y$ is functional.

Moreover, if $R \subseteq X \times Y$ is functional, then there is a unique function $X \xrightarrow{f} Y$ so that $R = \Gamma_f$.

Proof: The first claim is easy: Γ_f is total because for each $x \in X$, $x \mathrel{\Gamma_f} f(x)$. It is deterministic because if $x \mathrel{\Gamma_f} y_0$ and $x \mathrel{\Gamma_f} y_1$, then $y_0 = f(x) = y_1$.

Suppose R is a functional relation. Recall that $Y \xrightarrow{\{-\}} \mathcal{P}(Y)$ is the "singleton" function. Let

$$S(Y) = \{A \in \mathcal{P}(Y) \mid \exists y \in Y. A = \{y\}\}.$$

In other words, $S(Y)$ consists of the collection of singleton subsets of Y. Let $\kappa_{S(Y)}$ be the characteristic function of $S(Y)$. So $S(Y)$ is an equalizer $S(Y) = \{A \in \mathcal{P}(Y) \mid \kappa_{S(Y)}(A) = \top\}$. But in fact, Now consider the function $R[-]$ from X to $\mathcal{P}(Y)$ given by $R[x] = \{y \in Y \mid x \mathrel{R} y\}$. Because R is functional, $\kappa_{S(Y)}(R[x]) = \top$ for each x in

□

Example 10.4

The relation of "less than or equal to" $\leq$ on real numbers can be regarded as a subset $\leq \subseteq \mathbb{R} \times \mathbb{R}$ defined by $(x, y) \in \leq$ when x is actually less than or equal to y and $(x, y) \notin \leq$ otherwise. This is a good example of why we prefer to write $x \leq y$ instead of $\leq(x, y)$ or $(x, y) \in \leq(x, y)$. Note that $\leq$ is a total relation, because for any $x \in \mathbb{R}$ there is a $y \in \mathbb{R}$ for which $x \leq y$.

The relation $=$ on any set is functional because, for any $x \in X$ there is exactly one $y \in X$ so that $x = y$.

Define the relation S on $\mathbb{R}$ by $x \mathrel{S} y$ if and only if $x = y^2$. Then S is not deterministic because $1 \mathrel{S} 1$ and $1 \mathrel{S} -1$. It is not total because there is no y for which $-1 \mathrel{S} y$.

Lemma 10.12

For any function $f\colon Y \longrightarrow X$, the graph Γ_f is functional.

Proof: This is pretty obvious from the basic properties of functions. That is, for each $x \in X$, $f(x) \in Y$ and obviously $f(x) = f(x)$. So Γ_f is total. On the other hand if $f(x) = y$ and $f(x) = y'$, then $y = y'$. So Γ_f is deterministic. □

Suppose we have a functional relation $R \subseteq X \times Y$. Then it is reasonable to suppose it actually determines a function.

Principle 10.3

Suppose $R \subseteq X \times Y$ is a functional relation. Then there is a function $F_R\colon X \longrightarrow Y$ so that $\Gamma_{F_R} = R$.

It is easy enough to check that $F_{\Gamma_f} = f$ for any function f. That is, $F_{\Gamma_f}(x) = y$ if and only if $x \mathrel{\Gamma_f} y$ if and only if $f(x) = y$. So functions from $X \xrightarrow{f} Y$ correspond exactly to functional relations from $R \subseteq X \times Y$.

Exercises for Lecture 10

1. Define $T \subseteq \mathbb{R} \times \mathbb{R}$ by $x \mathrel{T} y$ if and only if $\tan x = y$. Is T deterministic? Is it total?
2. Show that for any relations $R \subseteq X \times Y$, $\Delta_X; R = R = R; \Delta_Y$.
3. Show that for any relations $R \subseteq W \times X$, $S \subseteq X \times Y$ and $T \subseteq Y \times Z$, $R; (S; T) = (R; S); T$.
4. Show that for any functions $X \xrightarrow{f} Y$ and $Y \xrightarrow{g} Z$, $\Gamma_f; \Gamma_g = \Gamma_{g \circ f}$. Show that for any set X, $\Gamma_{\mathrm{id}_X} = \Delta_X$.

10.6 Splitting and the Axiom of Choice

The us start the section with exercises.

Exercises for Lecture 10

Suppose functions $s\colon Y \longrightarrow X$ and $r\colon X \longrightarrow Y$ satisfy $r \circ s = \mathrm{id}_Y$.

1. Show that r is an epimorphism.
2. Show that s is a monomorphism.

3. Show that $s \circ r$ is *idempotent*: $(s \circ r) \circ (s \circ r) = s \circ r$.

Definition 10.14

Two functions $s\colon Y \longrightarrow X$ and $r\colon X \longrightarrow Y$ satisfying $r \circ s = \mathsf{id}_Y$ are called a *section-retract pair*. An idempotent function fXX ($f \circ f = f$) is called a *retraction*.

The latest exercise shows that every section-retract pair gives rise to a retraction. Conversely, if $f\colon X \longrightarrow X$ is a retraction, we may define $Y = f^+(X)$ – the forward image of X. Then f restricted to Y is a function $s\colon Y \longrightarrow X$ and corestricted to Y is a function $r\colon X \longrightarrow Y$. Evidently, s and r form a section-retract pair for which $f = r \circ s$. So retractions and section-retract pairs are essentially the same things.

In a section-retract pair, the section is a monomorphism. In fact, nearly all monomorphisms are sections.

Lemma 10.13

Suppose $Y \neq \emptyset$. For any monomorphism $m\colon Y \longrightarrow X$, there is a function $r\colon X \longrightarrow Y$ so that $r \circ m = \mathsf{id}_Y$.

Proof: Since $Y \neq \emptyset$, we may pick some $b_0 \in Y$. Now define a relation $R \subseteq X \times Y$ by $a \mathrel{R} b$ if and only if either $a = m(b)$ or it is the case that $\forall \in Y.a \neq m(y)$ and $b = b_0$. This is a total relation clearly. And it is deterministic because m is one-to-one. Hence R determines a function r from X to Y. Apparently, $r(m(b)) = b$ because $m(b) \mathrel{R} b$. □

This tells us that every monomorphism from a non-empty domain is *split*: it is the section of a section-retract pair.

The dual of this would be that every epimorphism is also split. But it turns out that the dual is not provable without additional assumptions about Y. We won't go what additional assumptions would help in general, but a hint is given by the natural numbers. Suppose $e\colon \mathbb{N} \longrightarrow X$ is an epimorphism. Then each subset $e^-(x) \subseteq \mathbb{N}$ is not empty. Every non-empty subset of $\mathbb{N}$ has a least element. So we may define a section corresponding to e by setting $s(x) =$ the least $n \in \mathbb{N}$ so that $e(n) = x$. The point is that $\mathbb{N}$ has enough structure (every non-empty subset has a least element) to allow us to *define* a section.

Without additional structure, it seems reasonable that a section might *exist* for an epimorphism, even though we may not be able to *construct* one explicitly. The Axiom of Choice addresses this situation. Roughly is says that some how, one can make arbitrarily many choices all at once, without having an explicit construction of those choices. Though we will not need it very often in this course, the Axiom of Choice (AC) is quite useful in other areas of mathematics, sometimes to prove something that genuinely needs the axiom, other times to simplify a proof that could have been proved without it.

Principle 10.4

Axiom of Choice

For every epimorphism $X \xrightarrow{e} Y$ there is a function $Y \xrightarrow{c} X$ for which $e \circ c = \mathsf{id}_Y$

We will rarely use the Axiom of Choice in this course and will draw attention to it when we do. It is a mainstay, however, of most approaches to analysis and topology. There are many equivalent formulations of the axiom that are useful to those areas of mathematics.

Other strong axioms of sets are possible. There are even axioms that inconsistent with AC in the sense that they implies that there actually are epimorphisms that can not be split. For these lectures, though, our goal is to have machinery to do “ordinary” mathematics. The Axiom of Choice falls into that category. The interested student should follow up with a course in Set Theory.

Part III

Applications

Lecture 11

Infinities

Evidently $\mathbb{N}$ is an infinite set. So are $\mathbb{Z}$ and $\mathbb{R}$. Are these sets the same size? Or are there different magnitudes of infinite sets?

For finite sets, it is easy to understand how to compare sizes. We can count the elements in each set and compare the result. But suppose we do not know how to count. We can still compare the two sets by trying to match the elements of the first set to elements of the second. Think of checking whether a pile of pennies is the same size as a pile of nickels. We could simply line up one penny next to one nickel until we run out of one or the other. If every penny lines up with a nickel and vice versa, we know the two collections have the same size without having to count anything.

The same idea works as well for any sets, even infinite ones. That is, if two sets "match" they are the same size. we simply need to understand what "match" means.

Definition 11.1

Sets X and Y are *equipotent* if there is a pair of functions $X \xrightarrow{f} Y$ and $Y \xrightarrow{g} X$ s that $g \circ f = \mathsf{id}_X$ and $f \circ g = \mathsf{id}_Y$. When X and Y are equipotent, we write $X \sim Y$.

Example 11.1

The sets $\mathbb{N}$ and $\mathbb{Z}$ are equipotent. Define $f\colon \mathbb{N} \longrightarrow \mathbb{Z}$ by $f(2n) = n$ and $f(2n+1) = -n-1$. This is easily verified to be one-to-one and onto. So there is an inverse function $g\colon \mathbb{Z} \longrightarrow \mathbb{N}$. Specifically,

$$g(a) = \begin{cases} 2a & \text{if } 0 \leq a \\ -2a-1 & \text{otherwise} \end{cases}$$

The fact that $\mathbb{N}$ and $\mathbb{Z}$ are the same size should be surprising. After all, intuitively there are more integers that non-negative integers (natural numbers). But the fact is we can set up a perfect match between the two sets. That matching is not the inclusion of $\mathbb{N} \subseteq \mathbb{Z}$, but it is a matching nonetheless. Sets that are either finite or equipotent to $\mathbb{N}$ are called *countable*. We will look carefully at countable sets in LEcture **??**.

For two sets to be equipotent, there must be a one-to-one, onto function between them. So if we can show that between two sets there can not possibly be such a function, then we know definitively that they are not equipotent. The following is an important example.

Theorem 11.1

For any set X, there is no onto function from X to $\mathcal{P}(X)$.

Proof: Consider a function $f\colon X \to \mathcal{P}(X)$. We show that f is not an onto function. That is, we must construct an element $D \in \mathcal{P}(X)$ so that $D \neq f(x)$ for all $x \in X$. This will show that f "missed" the D in the codomain. Any $D \in \mathcal{P}(X)$ with this property will do. Note that each $f(x)$ is a subset of X. So it makes sense to ask whether or not x is an element of $f(x)$.

Let $D = \{x \in X \mid x \notin f(x)\}$. We claim that $D \neq f(x)$ for every $x \in X$. Consider an arbitrary $x \in X$. Suppose $x \in f(x)$. Then by definition $x \notin D$. So $D \neq f(x)$. On the other hand, suppose $x \notin f(x)$. Then by definition $x \in D$. So again $D = f(x)$. Thus for every $x \in X$, it is the case that $D \neq f(x)$. □

This theorem tells us that there is no "largest" set. Any set is strictly smaller than its own powerset.

Definition 11.2

Write $X \lesssim Y$ if there is a monomorphism from X to Y.

We already know that $X \lesssim \mathcal{P}(X)$ for any X because the function $x \mapsto \{x\}$ is a monomorphism. Evidently, $X \subseteq Y$ implies $X \lesssim Y$ because inclusion functions are monomorphisms. So $\mathbb{N} \lesssim \mathbb{R}$. But suppose $X \lesssim Y$ and $Y \lesssim X$. That means there is a monomorphism from X to Y and, completely separately, a monomorphism from Y to X.

Example 11.2

The intervals $(0,1)$ and $[0,1]$ satisfy $(0,1) \lesssim [0,1]$ because $(0,1) \subseteq [0,1]$. But $[0,1] \lesssim (0,1)$ is also true because the function $f\colon [0,1] \longrightarrow (0,1)$ defined by $f(x) = (x+1)/4$ is also a monomorphism. So $(0,1) \lesssim [0,1]$ and $[0,1] \lesssim (0,1)$. The question is, are these two sets equipotent? That is, can we find an actual bijection between the two sets.

Lemma 11.1

Suppose $\{A_i\}_{i \in I}$ is a partition of X and $\{B_i\}_{i \in I}$ is a partition of Y. If for each $i \in I$, $A_i \sim B_i$, then $X \sim Y$.

Proof: For each $i \in I$, let $f_i\colon A_i \longrightarrow B_i$ be a bijection. Then define $h\colon X \longrightarrow Y$ by $h(x) = f_i(x)$ when $x \in A_i$. Verifying that h is a bijection is easy. □

Theorem 11.2

For any two sets X and Y, if $X \lesssim Y$ and $Y \lesssim X$, then $X \sim Y$.

Proof: We define sequences $X_0, X_1, \ldots$ and $Y_0, Y_1, \ldots$ of subsets X and Y by recursion:

$$\begin{aligned} X_0 &= X \setminus g^+(Y) \\ Y_0 &= Y \setminus f^+(X) \\ X_{k\curvearrowright} &= g^+(Y_k) \\ Y_{k\curvearrowright} &= f^+(X_k). \end{aligned}$$

Then we also define

$$X_e = \bigcup_{k \in \mathbb{N}} X_{2k}$$
$$Y_e = \bigcup_{k \in \mathbb{N}} Y_{2k}$$
$$X_o = \bigcup_{k \in \mathbb{N}} X_{2k+1}$$
$$Y_o = \bigcup_{k \in \mathbb{N}} Y_{2k+1}$$
$$X_* = X \setminus \bigcup_{n \in \mathbb{N}} X_n$$
$$Y_* = Y \setminus \bigcup_{n \in \mathbb{N}} Y_n.$$

Thus X_e consists of the elements in some even indexed X_{2k}, X_o of the elements in some odd indexed X_{2k+1}, and X_* of all the elements not falling in either of those cases. The sets Y_e, Y_o and Y_* are defined similarly.

By induction on m, we show that for all $n > m$, it is the case that $X_m \cap X_n = \emptyset$ and $Y_m \cap Y_n = \emptyset$.

- [Basis] By definition X_0 is disjoint from the range of g. For $n > 0$, the set X_n is contained in the range of g. So X_0 and X_n are disjoint. The proof for Y_0 and Y_n is identical.
- [Inductive Hypothesis] Suppose that for some m, it is the case that $X_m \cap X_n = \emptyset$ and $Y_m \cap Y_n = \emptyset$ for all $n > m$.
- [Inductive Step] We must show that $X_{m^\frown} \cap X_n = \emptyset$ and likewise $Y_{m^\frown} \cap Y_n = \emptyset$ for every $n > m^\frown$. Since $n > m^\frown$, it has a predecessor. That is, let $j^\frown = n$. Then $m > j$. But $X_{m^\frown} = g^+(Y_m)$ and $X_n = g^+(Y_j)$. And g^+ preserves intersections because g is one-to-one. By the inductive hypothesis, $Y_m \cap Y_j = \emptyset$. Hence $X_{m^\frown} \cap X_n = \emptyset$. The proof for the Y sets is essentially the same.

Now it easily follows that $X_e \cap X_o = \emptyset$, and by definition $X_e \cap X_*$ and $X_o \cap X_* = \emptyset$. And of course, $X_e \cup X_o \cup X_* = X$. Now evidently $f^+(X_e) = Y_o$ and $g^+(Y_e) = X_o$. So $X_e \sim Y_o$ and $X_o \sim Y_e$. We claim that $f^+(X_*) = Y_*$. For $x \in X_*$, $f(x) \notin Y_0$. And if $f(x) \in Y_n$ for some $n > 0$, then $x \in X_{n-1}$. But this contradicts $x \in X_*$. So indeed $f(x) \in Y_*$. For $y \in Y_*$, $y \notin Y_0$. So there is some $x \in X$ for which $f(x) = y$. But if $x \in X_n$ for some n, then $y \in Y_{n+1}$. Again, this is a contradiction. So $x \in X_*$.

Summarizing, we have $X_e \sim Y_o$, $X_o \sim Y_e$ and $X_* \sim Y_*$. Hence $X \sim Y$. □

This theorem answers our original question. $[0, 1]$ and $(0, 1)$ are equipotent. A much more surprising fact is that $\mathbb{N} \times \mathbb{N} \sim \mathbb{N}$. A proof is quite simple. The function $n \mapsto (n, n)$ is a monomorphism. So $\mathbb{N} \lesssim \mathbb{N} \times \mathbb{N}$. But also the function $(m, n) \mapsto 2^m 3^n$ is a monomorphism from $\mathbb{N} \times \mathbb{N}$ to $\mathbb{N}$. So $\mathbb{N} \times \mathbb{N} \lesssim \mathbb{N}$. According to Theorem 11.2, the two sets are equipotent.

Exercises for Lecture 11

1. Show that the set of even natural numbers is equipotent to $\mathbb{N}$ by an explicit bijection. That is, do not use Theorem 11.2.
2. Is the set of primes numbers equipotent to $\mathbb{N}$?
3. Show that $\mathcal{P}(\mathbb{N})$ and $\mathbb{R}$ are equipotent by finding monomorphisms $\mathcal{P}(\mathbb{N}) \to \mathbb{R}$ and $\mathbb{R} \to \mathcal{P}(\mathbb{N})$.

4. Show that $\mathbb{N}$ and $\mathbb{R}$ are not equipotent. Hint: Prove that every function $\mathtt{f}\colon \mathbb{N} \to \mathbb{R}$ fails to be onto.

APPENDIX **A**

Python as Mathematics

In the main text, we sometimes want to think about how mathematical ideas connect to computation. In fact, mathematics and computation are so intertwined that I don't see much point in thinking of them as distinct disciplines. We will occasionally need to describe an *algorithm* as part of our mathematics. We use an informal presentation based on Python for this, but many other languages would do as well. The reasons for choosing to base tings on Python are

- Many students in Discrete Math already know a bit of Python.
- Algorithms expressed in Python read very close to the "natural" way we would write them.

The reasons not to just use Python directly are

- We want to be able to use honest mathematical notation where that helps. We should not be able to write 2^n wheras Python would require the translation to $2{*}{*}n$.
- Python has many features we simply don't need, or want. So sticking to Python faithfully does not gain anything.
- There are computationally useful constructs that are not part of Python. To handle these, we need to extend Python anyway.

A.1 Basics

Python is a general purpose, interpreted programming language. In this course, we will use an informal "mathified" version of Python to illustrate certain concepts where mathematics and computation overlap. We do not need a thorough understanding of the entire language. In fact, our main interest will be how to calculate with natural numbers, lists and a few other structures.

Examples in these notes will not be executable directly as Python programs, but I have tried to strike a balance between readability (for us humans) and correctness as programs. You should be able to see the obvious changes needed to turn these examples into working code.

A wide variety of introductions to actual Python are available online, and of course, at Chapman CPSC230 is the course *Introduction to Computer Science* course that employs Python. If you have taken CPSC230, you can safely skim this lecture. A good, and concise introduction to Python for a complete novice is available at `http://learnpythonthehardway.org`.

An *identifier* in Python is simply a name for something. For some purposes, mathematicians typically call these things *variables*. For example, we might refer to x as a variable, but what we really mean is that x is our (perhaps temporary) name for something. Identifiers are more general than variables. An identifier may name a variable quantity (this is how x is typically used), or it may name a specific gadget that will never change. For example, sin is the identifier we all use to name the sine function. In Python, an identifier is typically a string of letters and digits, not beginning with a digit. Python relies on this fact to distinguish between numerals like 123 and identifiers like $a123$. An identifier can also include the underscore character _, but there are subtle conventions about identifiers that start with underscores. We are better off avoiding them except for certain built-in uses. Also, certain symbols, known as *keywords*, that look like identifiers are explicitly ruled out. In these note, keywords are typeset in bold fact. So it is easy to distinguish them from identifies. Here is a list of some common keywords: **def**, **if**, **else**, **elif**, **return**, **while**, **and**, **or**, **not**.

Typically, we will use the conventions that we use in the text for identifiers. A typical number variable may be named x, y, z and so on. Sometimes we will explicitly define a new function (like gcd) and typeset it in the standard "up shape" type face.

A.2 Arithmetic

For our purposes, we will almost exclusively use natural numbers or integers for numerical data. Python is a syntactically untyped language. So there is no purely syntactic way to tell that one variable n is intended to vary over natural numbers and other variable x, say, over lists of natural numbers. This is not a problem because the context almost always makes things clear. If there is a need, we will indicate the 'type' of a variable in comments. We describe typing for function definitions below.

For integers, we will more often need the *integer quotient* than the *rational quotient*. Remember that the for integer a and positive integer b, the integer quotient of a divided by b is the largest integer q so that $qb \leq a$. For negative b, it is the smallest integer q so that $a \leq qb$. In Python (3.x) integer quotients are indicated by the operator //. We follow that notation typeset in mathematics as //. So $6 \mathbin{//} 5 = 1$ and not 1.2.

For the remainder of a division, Python uses a percent sign a\%b. Mathematicians typically write $a \bmod b$. We will use the mathematical convention. Putting // and mod together amounts to what known as the Division Algorithm. It states that for any integer a and non-zero integer b, there is a unique pair integers numbers q and r so that

- $a = qb + r$
- $-b < r < b$
- $0 \leq rb$

In our notation, $q = a \mathbin{//} b$ and $r = a \bmod b$.

A.3 Assignment and Update

In Python, we can use an identifier as "storage" for a value. That is, $x = 3$ acts by storing the number 3 under the name x. This is called an *assignment*. In subsequent arithmetic, x is evaluated as 3. For example, $y = x + 2$ will store the value 5 under the name y. The equal sign in Python (and Pythonish) is *only* used for assignment. It never means anything else.

The same identifier can be assigned and reassigned. So for example,

Algorithm A.1

```
$x = 4$
$y = x + 2$
$x = 3$
```

executes the three statements in the order they appear. So in the second line, x has the value 4. So at the end of execution, x has the value 3 and y has the value 6.

We can also increment a value with a statement $n \mathrel{+}= 1$. If x contains a value 5 prior this, then it contains 6 after. The same idea works for incrementing or decrementing by anything other than 1. Also decrementing is accomplished by using n\minuseq1. Also other operations like multiplication and division work the same way, but incrementing or decrementing by 1 is most common.

A.4 Conditionals

The following is a very simple example of an algorithm illustrating some of the structure of Python(ish). Suppose we have numbers in variables a and b and wish to set a new variable z to be equal to the larger of the two. This can be accomplished by

Algorithm A.2

```
if $a \geq b$:
    $z = a$
else:
    $z = b$
```

The keywords **if** and **else**, along with the punctuation : and the indentation indicate that $a \geq b$ is checked. If it is true, then the statement indented after **if** ...: is exectuted. If it is false, the statement after **else**: is executed. Although this example does not illustrate it, the indented code (called a *block*) can consist of more than one statement.

Suppose we have three numbers in variables a, b and c and wish to assign the *smallest* value to the variable z. Here is an algorithm for doing this.

Algorithm A.3

```
if $a \leq b$: # then b is not smallest
    if $a \leq c$:
        $z = a$
    else:
        $z = c$
    else: # a is not smallest
if $b \leq c$:
    $z = b$
else:
    $z = c$
```

This illustrates that structures like **if**... **else** can be “nested”. Also, the octothorpe character # marks the beginning of a comment – not part of the algorithm itself, but only a bit of explanatory text.

Fairly commonly, we see a nesting where and **else**: statement is immediately followed by an indented **if** (as in lines 6 and 7 above). Since this is so common, Python provides a simplification. The following code is equivalent to the previous example.

Algorithm A.4

```
if $a \leq b$: # then b is not smallest
    if $a \leq c$:
        $z = a$
    else:
        $z = c$
    elif $b \leq c$: # then b is the smallest
        $z = b$
else:
    $z = c$
```

We can also combine test like $a \leq b$ by what are known as *Boolean* operators. [We discuss these in more detail in a later lecture.] The above code can be simplified using **and** as follows:

Algorithm A.5

```
if $a \leq b$ and $a \leq c$:
    $z = a$
elif $b \leq a$ and $b \leq c$:
    $z = b$
else:
    $z = c$
```

A.5 Function Definitions

To define a new function, we can write

Algorithm A.6

```
def max(a,b):
    if $a \geq b$:
        return a
    else:
        return b

z = max(3,6)
# Now z contains 6
```

Sometimes it will obvious that the arguments and results of a function are, say, integers and not lists. That is the case for max above. On the other hand, we might have intended to restrict the definition only to natural numbers, or the tyoes of arguments and results may not be clear. In those cases we “decorate” a function definition as in

Algorithm A.7

```
def max($a \in \mathbb{N}$,$b \in \mathbb{N}$)$\in \mathbb{N}$:
    if $a \geq b$:
        return a
    else:
        return b
```

Now it is clear that this defines a function $\mathrm{max}\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$.

A.6 Iteration

To indicate that a block of code is meant to be repeated as long as a certain condition holds, use **while**. For example, the following code defines a function that computes the factorial of a number:

Algorithm A.8

```
def fact($n \in \mathbb{N}$)$\in \mathbb{N}$:
    $r = 1$
    while $n > 0$:
        $r* = n$
        $n- = 1$
    return r

n = fact(5)
# Now n contains $5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 \cdot 1$
```

Iteration over each item in a list is accomplished by a **for** loop as in the following.

Algorithm A.9

```
def sum($L \in \mathsf{List}[\mathbb{N}]$)$\in \mathbb{N}$:
    $r = 0$
    for a in L:
        $r += a$
    return r
```

Iteration for a fixed number of times n is accomplished by iterating over the list $[0, \ldots, n-1]$. This is produced by the built =in range function. So for example, factorial can be defined by

Algorithm A.10

```
def fact($n \in \mathbb{N}$)$\in \mathbb{N}$:
    $r = 1$
    for i in range(n):
        $r* = i + 1$
    return r
```

A.7 Recursion

The last feature covered in this quick primer is that a defined function is permitted to refer to itself in its definition. Here is yet another definition of factorial.

Algorithm A.11

```
def fact($n \in \mathbb{N}$)$\in \mathbb{N}$:
    if n == 0:
        return 1
    else:
        return $n*$fact($n-1$)
```

A definition like this is said to be *recursive*.

To evaluate fact (3), the program must evaluate 3∗fact(2). In turn, the program must evaluate 3∗2∗fact(1). In turn, the program must evaluate 3∗2∗1∗fact(0). Finally, fact (0) returns 1. So the fact (3) evaluates 3∗2∗1∗1 and returns 6.

Exercises for Lecture A

Write a Python function hundred(n) that rounds an integer n to its nearest 100. So hundred(403) should return 400, whereas hundred(451) should return 500.

Write a Python function median5(a, b, c, d, e) that returns the median value from its five arguments. For example, if $a \leq b \leq c \leq d \leq e$, then the function should return the value of c.

A.8 Patterns for Natural Numbers and Lists

We will use a kind of pattern matching scheme for dealing with natural numbers, especially in recursive definitions. Since a natural number must either be 0 or $k^\frown$ for some k, we may write

```
# Suppose $n \in \mathbb{N}$
if $n$ == 0:
    ...
else $k^\frown$ = $n$:
    ... code using $k$ and $n$
```

For example, we may define a function computing factorial by

```
def fact($n \in \mathbb{N}$)$\in \mathbb{N}$:
    if $n$ == 0:
        return 1
    else $n$ == $k^\frown$:
        return $n \cdot \mathrm{fact}(k)$
```

We have introduced lists in a purely mathematical context, but in truth, their centrality to mathematics came to light because of computation. Our mathematical notation for lists is borrowed directly from languages like Python. In particular, notation like $[4,3,6]$ works equally as a mathematical list and as a list in Python. There are differences between our usage and Python that we must take into account.

Just as natural numbers meet the pattern 0 or $k^\frown$, lists follow the pattern $[\,]$ or $x : L$. The construction $x : L$ is not part of Python, but we will use it in Pythonish. We use it in analogy with $k^\frown$.

To illustrate, the following algorithm computes the concatenation of two lists:

```
def concat($l_1$,$l_2$):
    if $l_1$ == []:
        return $l_2$
    else $l_1$ == $x$:$l'$:
        return $x$:concat($l'$,$l_2$)
```