8 Infinite Sets

This chapter is about infinite sets and some challenges in proving things about them.

Wait a minute! Why bring up infinity in a Mathematics for *Computer Science* text? After all, any data set in a computer is limited by the size of the computer's memory, and there is a bound on the possible size of computer memory, for the simple reason that the universe is (or at least appears to be) bounded. So why not stick with *finite* sets of some large, but bounded, size? This is a good question, but let's see if we can persuade you that dealing with infinite sets is inevitable.

You may not have noticed, but up to now you've already accepted the routine use of the integers, the rationals and irrationals, and sequences of them. These are all infinite sets. Further, do you really want Physics or the other sciences to give up the real numbers on the grounds that only a bounded number of bounded measurements can be made in a bounded universe? It's pretty convincing—and a lot simpler—to ignore such big and uncertain bounds (the universe seems to be getting bigger all the time) and accept theories using real numbers.

Likewise in computer science, it's implausible to think that writing a program to add nonnegative integers with up to as many digits as, say, the stars in the sky—trillions of galaxies each with billions of stars—would be different from writing a program that would add *any* two integers, no matter how many digits they had. The same is true in designing a compiler: it's neither useful nor sensible to make use of the fact that in a bounded universe, only a bounded number of programs will ever be compiled.

Infinite sets also provide a nice setting to practice proof methods, because it's harder to sneak in unjustified steps under the guise of intuition. And there has been a truly astonishing outcome of studying infinite sets. Their study led to the discovery of fundamental, logical limits on what computers can possibly do. For example, in Section 8.2, we'll use reasoning developed for infinite sets to prove that it's impossible to have a perfect type-checker for a programming language.

So in this chapter, we ask you to bite the bullet and start learning to cope with infinity.

8.1 Infinite Cardinality

In the late nineteenth century, the mathematician Georg Cantor was studying the convergence of Fourier series and found some series that he wanted to say converged "most of the time," even though there were an infinite number of points where they didn't converge. As a result, Cantor needed a way to compare the size of infinite sets. To get a grip on this, he got the idea of extending the Mapping Rule Theorem 4.5.4 to infinite sets: he regarded two infinite sets as having the "same size" when there was a bijection between them. Likewise, an infinite set A should be considered "as big as" a set B when A surj B. So we could consider A to be "strictly smaller" than B, which we abbreviate as A strict B, when A is *not* "as big as" B:

Definition 8.1.1. A strict B iff NOT(A surj B).

On finite sets, this strict relation really does mean "strictly smaller." This follows immediately from the Mapping Rule Theorem 4.5.4.

Corollary 8.1.2. *For finite sets A, B,*

$$A \text{ strict } B \quad \textit{iff} \quad |A| < |B|.$$

Proof.

$$\begin{array}{lll} A \text{ strict } B & \text{iff} & \text{NOT}(A \text{ surj } B) \qquad\qquad (\text{Def } 8.1.1) \\ & \text{iff} & \text{NOT}(|A| \geq |B|) \qquad (\text{Theorem } 4.5.4.(4.5)) \\ & \text{iff} & |A| < |B|. \end{array}$$

■

Cantor got diverted from his study of Fourier series by his effort to develop a theory of infinite sizes based on these ideas. His theory ultimately had profound consequences for the foundations of mathematics and computer science. But Cantor made a lot of enemies in his own time because of his work: the general mathematical community doubted the relevance of what they called "Cantor's paradise" of unheard-of infinite sizes.

A nice technical feature of Cantor's idea is that it avoids the need for a definition of what the "size" of an infinite set might be—all it does is compare "sizes."

Warning: We haven't, and won't, define what the "size" of an infinite set is. The definition of infinite "sizes" requires the definition of some infinite sets called

ordinals with special well-ordering properties. The theory of ordinals requires getting deeper into technical set theory than we need to go, and we can get by just fine without defining infinite sizes. All we need are the relations surj and bij which reflect "as big as" and "same size" relations between sets.

But there's something else to watch out for: surj is *metaphorically* an "as big as" relation, and likewise bij is *metaphorically* the "same size" relation on sets. As you would expect, most of the "as big as" and "same size" properties of surj and bij on finite sets do carry over to infinite sets, but *some important ones don't*—as we're about to show. So you have to be careful: don't assume that surj, for example, has any particular "as big as" property on *infinite* sets until it's been proven.

Let's begin with some familiar properties of the "as big as" and "same size" relations on finite sets that do carry over exactly to infinite sets:

Lemma 8.1.3. *For any sets A, B, C,*

1. A surj B *iff* B inj A.
2. *If* A surj B *and* B surj C, *then* A surj C.
3. *If* A bij B *and* B bij C, *then* A bij C.
4. A bij B *iff* B bij A.

Part 1. follows from the fact that R has the $[\leq 1 \text{ out}, \geq 1 \text{ in}]$ surjective function property iff R^{-1} has the $[\geq 1 \text{ out}, \leq 1 \text{ in}]$ total, injective property. Part 2. follows from the fact that compositions of surjections are surjections. Parts 3. and 4. follow from the first two parts because R is a bijection iff R and R^{-1} are surjective functions. We'll leave verification of these facts to Problem 4.22.

Another familiar property of finite sets carries over to infinite sets, but this time some real ingenuity is needed to prove it:

Theorem 8.1.4. *[Schröder-Bernstein] For any sets A, B, if A* surj *B and B* surj *A, then A* bij *B.*

That is, the Schröder-Bernstein Theorem says that if A is "at least as big as" B and conversely, B is "at least as big as" A, then A "is the same size as" B. Phrased this way, you might be tempted to take this theorem for granted, but that would be a mistake. For infinite sets A and B, the Schröder-Bernstein Theorem is actually pretty technical. Just because there is a surjective function $f : A \to B$—which need not be a bijection—and a surjective function $g : B \to A$—which also need not be a bijection—it's not at all clear that there must be a bijection $e : A \to B$. The idea is to construct e from parts of both f and g. We'll leave the actual construction to Problem 8.14.

Another familiar set property is that for any two sets, either the first is at least as big as the second, or vice-versa. For finite sets this follows trivially from the Mapping Rule. It's actually still true for infinite sets, but assuming it is obvious would be mistaken again.

Theorem 8.1.5. *For* all *sets* A, B,

$$A \text{ surj } B \quad \text{OR} \quad B \text{ surj } A.$$

Theorem 8.1.5 lets us prove that another basic property of finite sets carries over to infinite ones:

Lemma 8.1.6.

$$A \text{ strict } B \quad \text{AND} \quad B \text{ strict } C \tag{8.1}$$

implies

$$A \text{ strict } C$$

for all sets A, B, C.

Proof. (of Lemma 8.1.6)

Suppose 8.1 holds, and assume for the sake of contradiction that NOT(A strict C), which means that A surj C. Now since B strict C, Theorem 8.1.5 lets us conclude that C surj B. So we have

$$A \text{ surj } C \quad \text{AND} \quad C \text{ surj } B,$$

and Lemma 8.1.3.2 lets us conclude that A surj B, contradicting the fact that A strict B. ■

We're omitting a proof of Theorem 8.1.5 because proving it involves technical set theory—typically the theory of ordinals again—that we're not going to get into. But since proving Lemma 8.1.6 is the only use we'll make of Theorem 8.1.5, we hope you won't feel cheated not seeing a proof.

8.1.1 Infinity is different

A basic property of finite sets that does *not* carry over to infinite sets is that adding something new makes a set bigger. That is, if A is a finite set and $b \notin A$, then $|A \cup \{b\}| = |A| + 1$, and so A and $A \cup \{b\}$ are not the same size. But if A is infinite, then these two sets *are* "the same size!"

Lemma 8.1.7. *Let A be a set and $b \notin A$. Then A is infinite iff A* bij *$A \cup \{b\}$.*

Proof. Since A is *not* the same size as $A \cup \{b\}$ when A is finite, we only have to show that $A \cup \{b\}$ *is* the same size as A when A is infinite. That is, we have to find a bijection between $A \cup \{b\}$ and A when A is infinite.

Here's how: since A is infinite, it certainly has at least one element; call it a_0. But since A is infinite, it has at least two elements, and one of them must not equal to a_0; call this new element a_1. But since A is infinite, it has at least three elements, one of which must be different from both a_0 and a_1; call this new element a_2. Continuing in this way, we conclude that there is an infinite sequence $a_0, a_1, a_2, \ldots, a_n, \ldots$ of different elements of A. Now it's easy to define a bijection $e : A \cup \{b\} \to A$:

$$\begin{aligned} e(b) &::= a_0, \\ e(a_n) &::= a_{n+1} && \text{for } n \in \mathbb{N}, \\ e(a) &::= a && \text{for } a \in A - \{b, a_0, a_1, \ldots\}. \end{aligned}$$

■

8.1.2 Countable Sets

A set C is *countable* iff its elements can be listed in order, that is, the elements in C are precisely the elements in the sequence

$$c_0, c_1, \ldots, c_n, \ldots.$$

Assuming no repeats in the list, saying that C can be listed in this way is formally the same as saying that the function, $f : \mathbb{N} \to C$ defined by the rule that $f(i) ::= c_i$, is a bijection.

Definition 8.1.8. A set C is *countably infinite* iff $\mathbb{N}$ bij C. A set is *countable* iff it is finite or countably infinite. A set is *uncountable* iff it is not countable.

We can also make an infinite list using just a finite set of elements if we allow repeats. For example, we can list the elements in the three-element set $\{2, 4, 6\}$ as

$$2, 4, 6, 6, 6, \ldots.$$

This simple observation leads to an alternative characterization of countable sets that does not make separate cases of finite and infinite sets. Namely, a set C is countable iff there is a list

$$c_0, c_1, \ldots, c_n, \ldots$$

of the elements of C, possibly with repeats.

Lemma 8.1.9. *A set C is countable iff* $\mathbb{N}$ surj C. *In fact, a nonempty set C is countable iff there is a* total *surjective function $g : \mathbb{N} \to C$.*

The proof is left to Problem 8.15.

The most fundamental countably infinite set is the set $\mathbb{N}$ itself. But the set $\mathbb{Z}$ of *all* integers is also countably infinite, because the integers can be listed in the order:

$$0, -1, 1, -2, 2, -3, 3, \ldots. \tag{8.2}$$

In this case, there is a simple formula for the nth element of the list (8.2). That is, the bijection $f : \mathbb{N} \to \mathbb{Z}$ such that $f(n)$ is the nth element of the list can be defined as:

$$f(n) ::= \begin{cases} n/2 & \text{if } n \text{ is even,} \\ -(n+1)/2 & \text{if } n \text{ is odd.} \end{cases}$$

There is also a simple way to list all *pairs* of nonnegative integers, which shows that $(\mathbb{N} \times \mathbb{N})$ is also countably infinite (Problem 8.25). From this, it's a small step to reach the conclusion that the set $\mathbb{Q}^{\geq 0}$ of nonnegative rational numbers is countable. This may be a surprise—after all, the rationals densely fill up the space between integers, and for any two, there's another in between. So it might seem that you couldn't write out all the rationals in a list, but Problem 8.13 illustrates how to do it. More generally, it is easy to show that countable sets are closed under unions and products (Problems 8.24 and 8.25) which implies the countability of a bunch of familiar sets:

Corollary 8.1.10. *The following sets are countably infinite:*

$$\mathbb{Z}^+, \mathbb{Z}, \mathbb{N} \times \mathbb{N}, \mathbb{Q}^+, \mathbb{Z} \times \mathbb{Z}, \mathbb{Q}.$$

A small modification of the proof of Lemma 8.1.7 shows that countably infinite sets are the "smallest" infinite sets. Namely,

Lemma 8.1.11. *If A is an infinite set, and B is countable, then A* surj *B.*

We leave the proof to Problem 8.12.

Also, since adding one new element to an infinite set doesn't change its size, you can add any *finite* number of elements without changing the size by simply adding one element after another. Something even stronger is true: you can add a *countably* infinite number of new elements to an infinite set and still wind up with just a set of the same size (Problem 8.18).

By the way, it's a common mistake to think that, because you can add any finite number of elements to an infinite set and have a bijection with the original set, that you can also throw in infinitely many new elements. In general it isn't true that just because it's OK to do something any finite number of times, it's also OK to do it an infinite number of times. For example, starting from 3, you can increment by 1 any finite number of times, and the result will be some integer greater than or equal to 3. But if you increment an infinite number of times, you don't get an integer at all.

8.1.3 Power sets are strictly bigger

Cantor's astonishing discovery was that *not all infinite sets are the same size*. In particular, he proved that for any set A the power set $\text{pow}(A)$ is "strictly bigger" than A. That is,

Theorem 8.1.12. *[Cantor] For any set A,*

$$A \text{ strict } \text{pow}(A).$$

Proof. To show that A is strictly smaller than $\text{pow}(A)$, we have to show that if g is a function from A to $\text{pow}(A)$, then g is *not* a surjection. Since any partial function with nonempty codomain can be extended to a total function with the same range (ask yourself how), we can safely assume that g is total.

To show that g is not a surjection, we'll simply find a subset $A_g \subseteq A$ that is not in the range of g. The idea is, for each element $a \in A$, to look at the set $g(a) \subseteq A$ and ask whether or not a happens to be in $g(a)$. First, define

$$A_g ::= \{a \in A \mid a \notin g(a)\}.$$

A_g is a well-defined subset of A, which means it is a member of $\text{pow}(A)$. But A_g can't be in the range of g, because it differs at a from each set $g(a)$ in the range of g.

To spell this out more, suppose to the contrary that A_g was in the range of g, that is,

$$A_g = g(a_0)$$

for some $a_0 \in A$. Now by definition of A_g,

$$a \in g(a_0) \quad \text{iff} \quad a \in A_g \quad \text{iff} \quad a \notin g(a)$$

for all $a \in A$. Now letting $a = a_0$ yields the contradiction

$$a_0 \in g(a_0) \quad \text{iff} \quad a_0 \notin g(a_0).$$

So g is not a surjection, because there is an element in the power set of A, specifically the set A_g, that is not in the range of g. ■

Cantor's Theorem immediately implies:

Corollary 8.1.13. $\text{pow}(\mathbb{N})$ *is uncountable.*

Proof. By Lemma 8.1.9, U is uncountable iff $\mathbb{N}$ strict U. ■

The bijection between subsets of an n-element set and the length n bit-strings $\{0,1\}^n$ used to prove Theorem 4.5.5, carries over to a bijection between subsets of a countably infinite set and the infinite bit-strings, $\{0,1\}^\omega$. That is,

Lemma 8.1.14. $\text{pow}(\mathbb{N}) \text{ bij} \{0,1\}^\omega$.

This immediately implies

Corollary 8.1.15. *$\{0,1\}^\omega$ is uncountable.*

More Countable and Uncountable Sets

Once we have a few sets we know are countable or uncountable, we can get lots more examples using Lemma 8.1.3. In particular, we can appeal to the following immediate corollary of the Lemma:

Corollary 8.1.16.

(a) If U is an uncountable set and A surj U, then A is uncountable.

(b) If C is a countable set and C surj A, then A is countable.

For example, now that we know that the set $\{0,1\}^\omega$ of infinite bit strings is uncountable, it's a small step to conclude that

Corollary 8.1.17. *The set $\mathbb{R}$ of real numbers is uncountable.*

To prove this, think about the infinite decimal expansion of a real number:

$$\begin{aligned}
\sqrt{2} &= 1.4142\ldots,\\
5 &= 5.000\ldots,\\
1/10 &= 0.1000\ldots,\\
1/3 &= 0.333\ldots,\\
1/9 &= 0.111\ldots,\\
4\frac{1}{99} &= 4.010101\ldots.
\end{aligned}$$

Let's map any real number r to the infinite bit string $b(r)$ equal to the sequence of bits in the decimal expansion of r, starting at the decimal point. If the decimal expansion of r happens to contain a digit other than 0 or 1, leave $b(r)$ undefined.

For example,

$$\begin{aligned} b(5) &= 000\ldots, \\ b(1/10) &= 1000\ldots, \\ b(1/9) &= 111\ldots, \\ b(4\tfrac{1}{99}) &= 010101\ldots \\ b(\sqrt{2}), b(1/3) &\text{ are undefined.} \end{aligned}$$

Now b is a function from real numbers to infinite bit strings.[1] It is not a total function, but it clearly is a surjection. This shows that

$$\mathbb{R} \text{ surj } \{0,1\}^\omega,$$

and the uncountability of the reals now follows by Corollary 8.1.16.(a).

For another example, let's prove

Corollary 8.1.18. *The set $(\mathbb{Z}^+)^*$ of all finite sequences of positive integers is countable.*

To prove this, think about the prime factorization of a nonnegative integer:

$$\begin{aligned} 20 &= 2^2 \cdot 3^0 \cdot 5^1 \cdot 7^0 \cdot 11^0 \cdot 13^0 \cdots, \\ 6615 &= 2^0 \cdot 3^3 \cdot 5^1 \cdot 7^2 \cdot 11^0 \cdot 13^0 \cdots. \end{aligned}$$

Let's map any nonnegative integer n to the finite sequence $e(n)$ of nonzero exponents in its prime factorization. For example,

$$\begin{aligned} e(20) &= (2,1), \\ e(6615) &= (3,1,2), \\ e(5^{13} \cdot 11^9 \cdot 47^{817} \cdot 103^{44}) &= (13,9,817,44), \\ e(1) &= \lambda, \qquad \text{(the empty string)} \\ e(0) &\text{ is undefined.} \end{aligned}$$

[1] Some rational numbers can be expanded in two ways—as an infinite sequence ending in all 0's or as an infinite sequence ending in all 9's. For example,

$$\begin{aligned} 5 &= 5.000\cdots = 4.999\ldots, \\ \frac{1}{10} &= 0.1000\cdots = 0.0999\ldots. \end{aligned}$$

In such cases, define $b(r)$ to be the sequence that ends with all 0's.

Now e is a function from $\mathbb{N}$ to $(\mathbb{Z}^+)^*$. It is defined on all positive integers, and it clearly is a surjection. This shows that

$$\mathbb{N} \text{ surj } (\mathbb{Z}^+)^*,$$

and the countability of the finite strings of positive integers now follows by Corollary 8.1.16.(b).

Larger Infinities

There are lots of different sizes of infinite sets. For example, starting with the infinite set $\mathbb{N}$ of nonnegative integers, we can build the infinite sequence of sets

$$\mathbb{N} \text{ strict } \text{pow}(\mathbb{N}) \text{ strict } \text{pow}(\text{pow}(\mathbb{N})) \text{ strict } \text{pow}(\text{pow}(\text{pow}(\mathbb{N}))) \text{ strict } \ldots.$$

By Cantor's Theorem 8.1.12, each of these sets is strictly bigger than all the preceding ones. But that's not all: the union of all the sets in the sequence is strictly bigger than each set in the sequence (see Problem 8.16). In this way you can keep going indefinitely, building "bigger" infinities all the way.

8.1.4 Diagonal Argument

Theorem 8.1.12 and similar proofs are collectively known as "diagonal arguments" because of a more intuitive version of the proof described in terms of on an infinite square array. Namely, suppose there was a bijection between $\mathbb{N}$ and $\{0,1\}^\omega$. If such a relation existed, we would be able to display it as a list of the infinite bit strings in some countable order or another. Once we'd found a viable way to organize this list, any given string in $\{0,1\}^\omega$ would appear in a finite number of steps, just as any integer you can name will show up a finite number of steps from 0. This hypothetical list would look something like the one below, extending to infinity both vertically and horizontally:

$$\begin{array}{lcccccccc}
A_0 & = & 1 & 0 & 0 & 0 & 1 & 1 & \cdots \\
A_1 & = & 0 & 1 & 1 & 1 & 0 & 1 & \cdots \\
A_2 & = & 1 & 1 & 1 & 1 & 1 & 1 & \cdots \\
A_3 & = & 0 & 1 & 0 & 0 & 1 & 0 & \cdots \\
A_4 & = & 0 & 0 & 1 & 0 & 0 & 0 & \cdots \\
A_5 & = & 1 & 0 & 0 & 1 & 1 & 1 & \cdots \\
\vdots & & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots
\end{array}$$

But now we can exhibit a sequence that's missing from our allegedly complete list of all the sequences. Look at the diagonal in our sample list:

$$\begin{array}{lcccccccc}
A_0 & = & \mathbf{1} & 0 & 0 & 0 & 1 & 1 & \cdots \\
A_1 & = & 0 & \mathbf{1} & 1 & 1 & 0 & 1 & \cdots \\
A_2 & = & 1 & 1 & \mathbf{1} & 1 & 1 & 1 & \cdots \\
A_3 & = & 0 & 1 & 0 & \mathbf{0} & 1 & 0 & \cdots \\
A_4 & = & 0 & 0 & 1 & 0 & \mathbf{0} & 0 & \cdots \\
A_5 & = & 1 & 0 & 0 & 1 & 1 & \mathbf{1} & \cdots \\
\vdots & & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots
\end{array}$$

Here is why the diagonal argument has its name: we can form a sequence D consisting of the bits on the diagonal.

$$D = \begin{array}{cccccccc} 1 & 1 & 1 & 0 & 0 & 1 & \cdots, \end{array}$$

Then, we can form another sequence by switching the **1**'s and **0**'s along the diagonal. Call this sequence C:

$$C = \begin{array}{cccccccc} 0 & 0 & 0 & 1 & 1 & 0 & \cdots. \end{array}$$

Now if the nth term of A_n is **1** then the nth term of C is **0**, and *vice versa*, which guarantees that C differs from A_n. In other words, C has at least one bit different from *every* sequence on our list. So C is an element of $\{0,1\}^\omega$ that does not appear in our list—our list can't be complete!

This diagonal sequence C corresponds to the set $\{a \in A \mid a \notin g(a)\}$ in the proof of Theorem 8.1.12. Both are defined in terms of a countable subset of the uncountable infinity in a way that excludes them from that subset, thereby proving that no countable subset can be as big as the uncountable set.

8.2 The Halting Problem

Although towers of larger and larger infinite sets are at best a romantic concern for a computer scientist, the *reasoning* that leads to these conclusions plays a critical role in the theory of computation. Diagonal arguments are used to show that lots of problems can't be solved by computation, and there is no getting around it.

This story begins with a reminder that having procedures operate on programs is a basic part of computer science technology. For example, *compilation* refers to taking any given program text written in some "high level" programming language

like Java, C++, Python, ..., and then generating a program of low-level instructions that does the same thing but is targeted to run well on available hardware. Similarly, *interpreters* or *virtual machines* are procedures that take a program text designed to be run on one kind of computer and simulate it on another kind of computer. Routine features of compilers involve "type-checking" programs to ensure that certain kinds of run-time errors won't happen, and "optimizing" the generated programs so they run faster or use less memory.

The fundamental thing that just can't be done by computation is a *perfect* job of type-checking, optimizing, or any kind of analysis of the overall run time behavior of programs. In this section, we'll illustrate this with a basic example known as the *Halting Problem*. Pick your favorite programming language—Python, Java, C++,...—and assume that "program" refers to a program written in your language.

Once a program is started, if its initial computation stops for some reason—such as producing a final value, waiting for an input, suffering an error interrupt, or simply freezing—the program is said to *halt*. So a program that does *not* halt would run forever using up cycles and energy (unless it was interrupted by an external operating system command). The Halting Problem is the general problem of determining, given an arbitrary program, whether or not the program halts.

There is a simple way to determine when an arbitrary program does halt, at least in theory: just run it until it stops. Well not quite. Just running the program in the usual way would not detect when it freezes without warning. What really needs to be done is to simulate the program using an interpreter or a virtual machine that will recognize any kind of halting, including a freeze. But interpreters and virtual machines capable of simulating any program are familiar technology. So there is a general way to detect when a program halts.

The hard part is determining when a program does *not* halt. At any point in simulating it, you generally won't know whether to continue the simulation because the program will halt later, or to abort the simulation because it wouldn't stop otherwise.

So is there some way besides simulation to detect when a program does *not* halt? Could there be some program analysis tool that could inspect any program and correctly report when the program does not halt? The answer is "No." Using a standard diagonal argument, we will prove that it is impossible to have such a non-halting analysis tool. Any method for detecting non-halting is bound to go wrong. It will falsely report that some halting program does not halt, or it will fail to report anything about some program. That is, there will be a program on which the analyzer runs forever without halting.

To set up the diagonal argument, we will focus on *string procedures*. A string procedure is a procedure that takes a single argument that is supposed to be a string

over the 256 character ASCII alphabet. As a simple example, you might think about how to write a string procedure that halts precisely when it is applied to a *double letter* string in ASCII*, namely, a string in which every character occurs twice in a row. For example, `aaCC33`, and `zz++ccBB` are double letter strings, but `aa;bb`, `b33`, and `AAAAA` are not.

When the computation of a procedure applied to a string halts, the procedure will be said to *recognize* the string. In this context, a set of strings a commonly called a (formal) *language*. We let lang(P) to be the language recognized by procedure P:

$$\text{lang}(P) ::= \{s \in \text{ASCII}^* \mid P \text{ applied to } s \text{ halts}\}.$$

A language is called *recognizable* when it equals lang(P) for some string procedure P. For example, we've just agreed that the set of double letter strings is recognizable.

There is no harm in assuming that every program can be written as a string in ASCII*—that's typically how we would enter them into a computer in the first place. When a string $s \in \text{ASCII}^*$ is actually the ASCII description of some string procedure, we'll refer to that string procedure as P_s. You can think of P_s as the result of compiling s into something executable.[2] It's technically helpful to treat *every* string in ASCII* as a program for a string procedure. So when a string $s \in \text{ASCII}^*$ doesn't parse as a proper string procedure, we'll define P_s to be some default string procedure—say one that never halts on anything it is applied to.

Focusing just on string procedures, the general Halting Problem is to decide, given strings s and t, whether or not the procedure P_s recognizes t. Following the usual diagonal approach, we define the language No-halt:

Definition 8.2.1.

$$\text{No-halt} ::= \{s \mid P_s \text{ applied to } s \text{ does not halt}\} = \{s \notin \text{lang}(P_s)\}. \tag{8.3}$$

So if P_s is the recognizer for some language, then No-halt differs from that language on the string s. This shows that No-halt cannot have a recognizer:

Theorem 8.2.2. *No-halt is not recognizable.*

Let's spell out the reasoning behind this theorem more fully. By definition, we have

$$s \in \text{No-halt IFF } s \notin \text{lang}(P_s), \tag{8.4}$$

[2] The string $s \in \text{ASCII}^*$ and the procedure P_s have to be distinguished to avoid a type error: you can't apply a string to string. For example, let s be the string that you wrote as your program to recognize the double letter strings. Applying s to a string argument, say `aabbccdd`, should throw a type exception; what you need to do is compile s to the procedure P_s and then apply P_s to `aabbccdd`.

for all strings $s \in \text{ASCII}^*$.

Now suppose to the contrary that No-halt was recognizable. This means there is some procedure P_{s_0} that recognizes No-halt, that is,

$$\text{No-halt} = \text{lang}(P_{s_0})\,.$$

Combined with (8.4), we have

$$s \in \text{lang}(P_{s_0}) \quad \text{iff} \quad s \notin \text{lang}(P_s) \tag{8.5}$$

for all $s \in \text{ASCII}^*$. Now letting $s = s_0$ in (8.5) yields the immediate contradiction

$$s_0 \in \text{lang}(P_{s_0}) \quad \text{iff} \quad s_0 \notin \text{lang}(P_{s_0})\,.$$

So that does it: the reasoning above applied to whatever favorite programming language you picked. It is *logically impossible* for a Java program to be a recognizer for non-halting Java programs, or for a Python program to be a recognizer for non-halting Python programs, and so forth for other favorite programming languages.

Now you might wonder if there was a loophole around this logical limitation by having a recognizer the non-halting programs in one language that was written in another language. In other words, could there be a C++ procedure that recognizes all the non-halting Java programs? After all, C++ does allow more intimate manipulation of computer memory than Java does. But there is no loophole here. If you've learned about programming language implementation, you will realize that it's possible to write a simulator in Java for C++ programs. This means that if there were a C++ procedure that recognized non-halting Java programs, then a Java procedure could also do it by simulating the C++ program, and that's impossible. This reasoning finally leads to a transcendent insight. No procedure can be written in *any* programming language that recognizes No-halt for your favorite language. Recognizing No-halt is simply beyond the capacity of computation.

But it's not just No-halt. If there was a perfect recognizer for *any* property that depends on the complete run time behavior programs,[3] it could be altered slightly to become a recognizer for No-halt. We'll take this claim for granted now, giving its full justification in an assigned problem.

For example, most compilers do "static" type-checking at compile time to ensure that programs won't make run-time type errors. A program that type-checks is guaranteed not to cause a run-time type-error. But given that it's impossible to recognize when the complete run time behavior of a program does not lead to a

[3] The weasel words "complete run time behavior" creep in here to rule out some run time properties that are easy to recognize because they depend only on part of the run time behavior. For example, the set of programs that halt after executing at most 100 instructions is recognizable.

type-error, it follows that the type-checker must be rejecting programs that really wouldn't cause a type-error. The conclusion is that no type-checker is perfect—you can always do better!

It's a different story if we think about the *practical* possibility of writing program analyzers. The fact that it's logically impossible to analyze utterly arbitrary programs does not mean that you can't do a very good job analyzing interesting programs that come up in practice. In fact, these "interesting" programs are commonly *intended* to be analyzable in order to confirm that they do what they're supposed to do.

In the end, it's not clear how much of a hurdle this theoretical limitation implies in practice. But the theory does provide some perspective on claims about general analysis methods for programs. The theory tells us that people who make such claims either

- are exaggerating the power (if any) of their methods, perhaps to make a sale or get a grant, or
- are trying to keep things simple by not going into technical limitations they're aware of, or
- perhaps most commonly, are so excited about some useful practical successes of their methods that they haven't bothered to think about the limitations which must be there.

So from now on, if you hear people making claims about having general program analysis/verification/optimization methods, you'll know they can't be telling the whole story.

8.3 The Logic of Sets

8.3.1 Russell's Paradox

Reasoning naively about sets turns out to be risky. In fact, one of the earliest attempts to come up with precise axioms for sets in the late nineteenth century by the logician Gotlob Frege, was shot down by a three line argument known as *Russell's Paradox*[4] which reasons in nearly the same way as the proof of Cantor's

[4]Bertrand Russell was a mathematician/logician at Cambridge University at the turn of the Twentieth Century. He reported that when he felt too old to do mathematics, he began to study and write about philosophy, and when he was no longer smart enough to do philosophy, he began writing about politics. He was jailed as a conscientious objector during World War I. For his extensive philosophical and political writing, he won a Nobel Prize for Literature.

Theorem 8.1.12. This was an astonishing blow to efforts to provide an axiomatic foundation for mathematics:

> Russell's Paradox
>
> Let S be a variable ranging over all sets, and define
>
> $$W ::= \{S \mid S \notin S\}.$$
>
> So by definition,
>
> $$S \in W \text{ iff } S \notin S,$$
>
> for every set S. In particular, we can let S be W, and obtain the contradictory result that
>
> $$W \in W \text{ iff } W \notin W.$$

The simplest reasoning about sets crashes mathematics! Russell and his colleague Whitehead spent years trying to develop a set theory that was not contradictory, but would still do the job of serving as a solid logical foundation for all of mathematics.

Actually, a way out of the paradox was clear to Russell and others at the time: *it's unjustified to assume that W is a set*. The step in the proof where we let S be W has no justification, because S ranges over sets, and W might not be a set. In fact, the paradox implies that W had better not be a set!

But denying that W is a set means we must *reject* the very natural axiom that every mathematically well-defined collection of sets is actually a set. The problem faced by Frege, Russell and their fellow logicians was how to specify *which* well-defined collections are sets. Russell and his Cambridge University colleague Whitehead immediately went to work on this problem. They spent a dozen years developing a huge new axiom system in an even huger monograph called *Principia Mathematica*, but for all intents and purposes, their approach failed. It was so cumbersome no one ever used it, and it was subsumed by a much simpler, and now widely accepted, axiomatization of set theory by the logicians Zermelo and Fraenkel.

8.3.2 The ZFC Axioms for Sets

It's generally agreed that essentially *all of mathematics* can be derived from a few formulas of set theory, called the Axioms of *Zermelo-Fraenkel Set Theory* with

Choice (ZFC), using a few simple logical deduction rules.

These axioms are about "pure" sets that are literally built up from nothing. For example, we can start with the set $\emptyset$ with nothing in it, and then build new sets from stuff we already have. So from $\emptyset$ we can build the set $\{\emptyset\}$ whose sole member is the set we started with. But now we have built two things, so we can build two new sets

$$\{\{\emptyset\}\}, \quad \{\emptyset, \{\emptyset\}\}.$$

Continuing like this, we could build up a simple sequence of sets

$$\emptyset, \quad \{\emptyset\}, \quad \{\{\emptyset\}\}, \quad \{\{\{\emptyset\}\}\}, \ldots$$

where each element is simply the set whose sole element is the previous element. This means we can now build an infinite set N consisting of all the elements in the sequence.

Of course there are lots of types of mathematical objects—sets of numbers like $\mathbb{N}$ or $\mathbb{R}$, sets of functions on numbers, sets of sequences like $\{0,1\}^*$ or $\{0,1\}^\omega$, or sets of graphs, for example—that we don't normally regard as pure sets. But in fact pure sets are enough to model all these other types of objects.

For example, the infinite set N is like a copy of the nonnegative integers $\mathbb{N}$, where we define the result of "adding one" to an element $s \in N$ to be $\{x\}$. If we go on and take all the pairs[5] of elements $(a, b) \in N \times (N - \{\emptyset\})$, we get a set that is like the numerator/denominator pairs of all the fractions with nonnegative numerators and positive denominators. With a little fiddling of these "fractions," we can get a set that acts like a copy of the rational numbers $\mathbb{Q}$. Next from the rationals we can get a set that acts like the real numbers by modelling a real number $r \in \mathbb{R}$ by the set of rationals greater than r. And so on. The point is that if we can reason correctly about pure sets, we get a solid basis for reasoning about all mathematical objects.

So the domain of discourse of ZFC is the pure sets, and ZFC involves using a few logical inference rules to prove properties of pure sets starting from a few axioms. The axioms themselves are written as *pure first-order formulas* of set theory. These are predicate formulas that only talk about membership in sets. That is, a first-order formula of set theory is built using logical connectives and quantifiers starting *solely* from expressions of the form "$x \in y$" which is interpreted to mean that x and y are pure sets, and x is one of the elements in y.

Formulas of set theory are not even allowed to have the equality symbol "=." However, sets are equal iff they have the same elements, so there is an easy way to

[5] Doing this requires a way to represent an ordered pair (a, b) of things in terms of sets built up from just a and b. See Problem 8.38 for a slightly ingenious, but still simple, way to represent the pair as a set.

express equality of sets purely in terms of membership:

$$(x = y) ::= \forall z.\ (z \in x \text{ IFF } z \in y). \tag{8.6}$$

Similarly, the subset symbol "$\subseteq$" is not allowed in formulas of set theory, but we can also express subset purely in terms of membership:

$$(x \subseteq y) ::= \forall z.\ (z \in x \text{ IMPLIES } z \in y). \tag{8.7}$$

So formulas using symbols "$=$," "$\subseteq$," in addition to "$\in$" can be understood as *abbreviations* for pure formulas only using "$\in$." We won't worry about this distinction between formulas and abbreviations for formulas—we'll now just call them all "formulas of set theory." For example,

$$x = y \text{ IFF } [x \subseteq y \text{ AND } y \subseteq x]$$

is a formula of set theory that explains a basic connection between set equality and set containment.

We're *not* going to develop much of set theory in this text, but we thought you might like to see the actual axioms of ZFC—and while you're at it, get some more practice reading and writing quantified formulas:

The Axioms

Extensionality. Two sets are equal iff they are members of the same sets:

$$x = y \text{ IFF } (\forall z.\ x \in z \text{ IFF } y \in z).$$

Pairing. For any two sets x and y, there is a set $\{x, y\}$ with x and y as its only elements:

$$\forall x, y \exists u \forall z.\ [z \in u \text{ IFF } (z = x \text{ OR } z = y)]$$

Union. The union u of a collection z of sets is also a set:

$$\forall z \exists u \forall x.\ (x \in u) \text{ IFF } (\exists y.\ x \in y \text{ AND } y \in z)$$

Infinity. There is an infinite set. Specifically, there is a nonempty set x such that for any set $y \in x$, the set $\{y\}$ is also a member of x.

Subset. Given any set x and any definable property of sets, there is a set y containing precisely those elements in x that have the property.

$$\forall x \exists y \forall z.\ z \in y \text{ IFF } [z \in x \text{ AND } \phi(z)]$$

where $\phi(z)$ is a formula of set theory.[6]

[6] This axiom is more commonly called the *Comprehension Axiom*.

Power Set. All the subsets of a set form another set:

$$\forall x \exists p \forall u.\ u \subseteq x \text{ IFF } u \in p.$$

Replacement. Suppose a formula ϕ of set theory defines the graph of a function on a set s, that is,

$$\forall x \in s\ \forall y, z.\ [\phi(x, y) \text{ AND } \phi(x, z)] \text{ IMPLIES } y = z.$$

Then the image of s under that function is also a set t. Namely,

$$\exists t \forall y.\ y \in t \text{ IFF } [\exists x \in s.\ \phi(x, y)].$$

Foundation. The aim is to forbid any infinite sequence of sets of the form

$$\cdots \in x_n \in \cdots \in x_1 \in x_0$$

in which each set is a member of the next one. This can be captured by saying every nonempty set has a "member-minimal" element. Namely, define

$$\text{member-minimal}(m, x) ::= [m \in x \text{ AND } \forall y \in x.\ y \notin m].$$

Then the Foundation Axiom[7] is

$$\forall x.\ x \neq \emptyset \text{ IMPLIES } \exists m.\ \text{member-minimal}(m, x).$$

Choice. Let s be a set of nonempty, disjoint sets. Then there is a set c consisting of exactly one element from each set in s. The formula is given in Problem 8.39.

8.3.3 Avoiding Russell's Paradox

These modern ZFC axioms for set theory are much simpler than the system Russell and Whitehead first came up with to avoid paradox. In fact, the ZFC axioms are as simple and intuitive as Frege's original axioms, with one technical addition: the Foundation axiom. Foundation captures the intuitive idea that sets must be built up from "simpler" sets in certain standard ways. And in particular, Foundation avoids the "paradox" of Russell's set collection $W ::= \{S \mid S \notin S\}$ because it implies that W is not a set. Namely Foundation implies that $S \notin S$ for *every* set S, so W is the collection of *all* sets. Now if W was a set, we would have $W \in W$, violating Foundation.

The Foundation axiom also justifies proofs about sets using structural induction. We'll begin with a recursive definition of a class of sets called Recset.

[7] This axiom is also called the *Regularity Axiom*.

Definition 8.3.1. The class of *recursive sets* Recset is defined as follows:

Base case: The empty set $\emptyset$ is a Recset.

Constructor step: If S is a nonempty set of Recset's, then S is a Recset.

Since Recset is defined recursively, we know that we can use structural induction to prove things about it. This means that we can actually use structural induction to prove things about all sets, because the Foundation axiom implies that *all* sets are recursive sets:

Theorem 8.3.2. *The class Recset of recursive sets is the same as the class of all sets.*

Proof. Everything in Recset is a set by definition, so we need only show that every set is in Recset.

Suppose to the contrary that a set S is not in Recset. So S must have an element N_0 that is *not* in Recset, since otherwise S would by definition be in Recset. Likewise, N_0 must have an element N_1 that is *not* in Recset, and N_1 must have an element N_2 that is *not* in Recset, and so on. So we have an infinite sequence

$$S \ni N_0 \ni N_1 \ni N_2 \ni \ldots.$$

Therefore the set[8]

$$T ::= \{S, N_0, N_1, N_2, \ldots\}$$

has no member-minimal element, in violation of Foundation. ■

8.4 Does All This Really Work?

So this is where mainstream mathematics stands today: there is a handful of ZFC axioms from which virtually everything else in mathematics can be derived logically. This sounds like a rosy situation, but there are several dark clouds, suggesting that the essence of truth in mathematics is not completely resolved.

- The ZFC axioms weren't etched in stone by God. Instead, they were mostly made up by Zermelo, who may have been a brilliant logician, but was also a fallible human being—probably some days he forgot his house keys. So maybe Zermelo, just like Frege, didn't get his axioms right and will be

[8] A rigorous proof, which we are skipping, requires careful use of the Choice axiom to prove that T is a set.

shot down by some successor to Russell who will use his axioms to prove a proposition P and its negation $\overline{P}$. Then math as we understand it would be broken—this may sound crazy, but it has happened before.

In fact, while there is broad agreement that the ZFC axioms are capable of proving all of standard mathematics, the axioms have some further consequences that sound paradoxical. For example, the Banach-Tarski Theorem says that, as a consequence of the *axiom of choice*, a solid ball can be divided into six pieces and then the pieces can be rigidly rearranged to give *two* solid balls of the same size as the original!

- Some basic questions about the nature of sets remain unresolved. For example, Cantor raised the question whether there is a set whose size is strictly between the smallest infinite set $\mathbb{N}$ (see Problem 8.12) and the strictly larger set $\text{pow}(\mathbb{N})$? Cantor guessed not:

  Cantor's *Contiuum Hypothesis*: There is no set A such that

  $$\mathbb{N} \text{ strict } A \text{ strict } \text{pow}(\mathbb{N}).$$

  The Continuum Hypothesis remains an open problem a century later. Its difficulty arises from one of the deepest results in modern Set Theory—discovered in part by Gödel in the 1930's and Paul Cohen in the 1960's—namely, the ZFC axioms are not sufficient to settle the Continuum Hypothesis: there are two collections of sets, each obeying the laws of ZFC, and in one collection the Continuum Hypothesis is true, and in the other it is false. Until a mathematician with a deep understanding of sets can extend ZFC with persuasive new axioms, the Continuum Hypothesis will remain undecided.

- But even if we use more or different axioms about sets, there are some unavoidable problems. In the 1930's, Gödel proved that, assuming that an axiom system like ZFC is consistent—meaning you can't prove both P and $\overline{P}$ for any proposition, P—then the very proposition that the system is consistent (which is not too hard to express as a logical formula) cannot be proved in the system. In other words, no consistent system is strong enough to verify itself In particular, every axiom system is *incomplete*: it cannot prove all the truths of mathematics.

8.4.1 Large Infinities in Computer Science

If the romance of different-size infinities and continuum hypotheses doesn't appeal to you, not knowing about them is not going to limit you as a computer scientist.

These abstract issues about infinite sets rarely come up in mainstream mathematics, and they don't come up at all in computer science, where the focus is generally on "countable," and often just finite, sets. In practice, only logicians and set theorists have to worry about collections that are "too big" to be sets. That's part of the reason that the 19th century mathematical community made jokes about "Cantor's paradise" of obscure infinities. But the challenge of reasoning correctly about this far-out stuff led directly to the profound discoveries about the logical limits of computation described in Section 8.2, and that really is something every computer scientist should understand.

Problems for Section 8.1

Practice Problems

Problem 8.1.
Show that the set $\{0,1\}^*$ of finite binary strings is countable.

Problem 8.2.
Describe an example of two **un**countable sets A and B such that there is no bijection between A and B.

Problem 8.3.
Indicate which of the following assertions (there may be more than one) are equivalent to

$$A \text{ strict } \mathbb{N}.$$

- $|A|$ is undefined.
- A is countably infinite.
- A is uncountable.
- A is finite.
- $\mathbb{N}$ surj A.
- $\forall n \in \mathbb{N}, |A| \leq n$.
- $\forall n \in \mathbb{N}, |A| \geq n$.

- $\exists n \in \mathbb{N}. |A| \leq n$.
- $\exists n \in \mathbb{N}. |A| < n$.

Problem 8.4.
Prove that if there is a total injective ([$\geq$ 1 out, $\leq$ 1 in]) relation from S to $\mathbb{N}$, then S is countable.

Problem 8.5.
Prove that if S is an infinite set, then $\text{pow}(S)$ is uncountable.

Problem 8.6.
Let

$$f_0, f_1, f_2, \ldots, f_k, \ldots$$

be an infinite sequence of positive real-valued total functions $f_k : \mathbb{N} \to \mathbb{R}^+$.

(a) A function from $\mathbb{N}$ to $\mathbb{R}$ that eventually gets bigger that every one of the f_k is said to *majorize* the set of f_k's. So finding a majorizing function is a different way than diagonalization to find a function that is not equal to any of the f_k's.

Given an explicit formula for a majorizing function $g : \mathbb{N} \to \mathbb{R}$ for the f_k's, and indicate how big n should be to ensure that $g(n) > f_k(n)$.

(b) Modify your answer to part (a), if necessary, to define a majorizing function h that *grows* more rapidly than f_k for every $k \in \mathbb{N}$, namely,

$$\lim_{n\to\infty} \frac{f_k(n)}{h(n)} = 0.$$

Explain why.

Problem 8.7.
Let

$$f_0, f_1, f_2, \ldots, f_k, \ldots$$

be an infinite sequence of (not necessarily different) nonnegative real-valued total functions $f_k : \mathbb{N} \to \mathbb{R}^{\geq 0}$. Define the function $s : \mathbb{N} \to \mathbb{R}^{\geq 0}$ by the rule:

$$s(n) ::= \sum_{k \leq \sqrt{n}} f_k(n).$$

(a) There is a function $b(k)$ such that from $b(k)$ on, s is at least as big as f_k:

$$\forall k \forall n \geq b(k).\ s(n) \geq f_k(n).$$

Which functions of k below are examples of a possible $b(k)$?

i k
ii $\sqrt{k}$
iii $\sqrt{n}$
iv n^2
v k^2
vi k^3
vii 2^k

(b) Which properties of the sequence of f_k's ensures that s does not appear in the sequence?

i f_7 is positive ($\forall n.\ f_7(n) > 0$).
ii f_7 is positive infinitely often ($\forall k \exists n.\ f_7(n) > 0$).
iii f_7 has no upper bound.
iv $f_7(n) = n^2$.
v f_7 and f_8 are positive.
vi f_7 and f_8 are positive infinitely often (not necessarily at the same places),
vii neither f_7 nor f_8 has an upper bound.

Problem 8.8.
Let A to be some infinite set. We know from Lemma 8.1.7 that

$$A \text{ bij } (A \cup \{b_0\}) \tag{Aub}$$

for any element b_0. An easy induction implies that

$$A \text{ bij } (A \cup \{b_0, b_1, \ldots, b_n\}) \tag{Aub$_i$}$$

for any finite set $\{b_0, b_1, \ldots, b_n\}$.

Students sometimes assume that this implies that

$$A \text{ bij } (A \cup B), \tag{8.8}$$

but it doesn't. For example, (AuB) is not true if A is $\mathbb{N}$ and B is the real numbers $\mathbb{R}$.[9]

A collection $\mathcal{C}$ of sets is called a *chain* when, given any two sets in $\mathcal{C}$, one is a subset of the other. A predicate $P(C)$ is said to be *finitely continuous* if, whenever $\mathcal{F}$ is a chain of *finite* sets, and $P(F)$ is true for every $F \in \mathcal{F}$, then $P(\bigcup \mathcal{F})$ is true. Claiming that Aub_i implies (AuB) amounts to claiming that the predicate $P_A(C)$ is finitely continuous, where

$$P_A(C) ::= A \text{ bij } (A \cup C).$$

But it isn't, as the example with $A = \mathbb{N}$ and $C = \mathbb{R}$ demonstrates.

Briefly explain which of the following predicates $P(C)$ is finitely **c**ontinuous and which **n**ot.

1. C is finite.
2. C is uncountable.
3. $C = \emptyset$.
4. There is a minimum element $b \in C \cap \mathbb{N}$.
5. There is a minimum element $b \in C \cap \mathbb{Z}$.
6. $\pi/2 \in C$.
7. $\exists \epsilon > 0\, \forall a, b \in C \cap \mathbb{R}.\ |a - b| > \epsilon$.
8. $C \cup \mathbb{N}$ is finite.
9. $C \subseteq \mathbb{N}$.
10. $C \subset \mathbb{N}$.
11. C is countable.

Problem 8.9.
Prove that the set of all *finite* subsets of positive integers is countable. *Hint:* Classify the subsets by the sum of their elements.

[9] It happens that(AuB) is true *if B is countable*, (Problem 8.18), but this is not completely obvious and takes some proving.

Problem 8.10.
A collection $\mathcal{C}$ of sets is called a *chain* when, given any two sets in $\mathcal{C}$, one is a subset of the other. Prove that if $\mathcal{F}$ is chain of *finite* sets, then $\bigcup \mathcal{F}$ is countable. (Notice that without the chain condition, every set is the union of its finite subsets.)

Class Problems

Problem 8.11.
Show that the set $\mathbb{N}^*$ of finite sequences of nonnegative integers is countable.

Problem 8.12. **(a)** Several students felt the proof of Lemma 8.1.7 was worrisome, if not circular. What do you think?

(b) Use the proof of Lemma 8.1.7 to show that if A is an infinite set, then A surj $\mathbb{N}$, that is, every infinite set is "at least as big as" the set of nonnegative integers.

Problem 8.13.
The rational numbers fill the space between integers, so a first thought is that there must be more of them than the integers, but it's not true. In this problem you'll show that there are the same number of positive rationals as positive integers. That is, the positive rationals are countable.

(a) Define a bijection between the set $\mathbb{Z}^+$ of positive integers, and the set $(\mathbb{Z}^+ \times \mathbb{Z}^+)$ of all pairs of positive integers:

$$\begin{array}{l}(1,1),(1,2),(1,3),(1,4),(1,5),\ldots\\(2,1),(2,2),(2,3),(2,4),(2,5),\ldots\\(3,1),(3,2),(3,3),(3,4),(3,5),\ldots\\(4,1),(4,2),(4,3),(4,4),(4,5),\ldots\\(5,1),(5,2),(5,3),(5,4),(5,5),\ldots\\\quad\vdots\end{array}$$

(b) Conclude that the set $\mathbb{Q}^+$ of all positive rational numbers is countable.

Problem 8.14.
This problem provides a proof of the [Schröder-Bernstein] Theorem:

$$\text{If } A \text{ inj } B \text{ and } B \text{ inj } A, \text{ then } A \text{ bij } B. \tag{8.9}$$

Since A inj B and B inj A, there are are total injective functions $f : A \to B$ and $g : B \to A$.

Assume for simplicity that A and B have no elements in common. Let's picture the elements of A arranged in a column, and likewise B arranged in a second column to the right, with left-to-right arrows connecting a to $f(a)$ for each $a \in A$ and likewise right-to-left arrows for g. Since f and g are total functions, there is *exactly one* arrow *out* of each element. Also, since f and g are injections, there is *at most one* arrow *into* any element.

So starting at any element, there is a unique and unending path of arrows going forwards (it might repeat). There is also a unique path of arrows going backwards, which might be unending, or might end at an element that has no arrow into it. These paths are completely separate: if two ran into each other, there would be two arrows into the element where they ran together.

This divides all the elements into separate paths of four kinds:

(i) paths that are infinite in both directions,

(ii) paths that are infinite going forwards starting from some element of A.

(iii) paths that are infinite going forwards starting from some element of B.

(iv) paths that are unending but finite.

(a) What do the paths of the last type (iv) look like?

(b) Show that for each type of path, either

(i) the f-arrows define a bijection between the A and B elements on the path, or

(ii) the g-arrows define a bijection between B and A elements on the path, or

(iii) both sets of arrows define bijections.

For which kinds of paths do both sets of arrows define bijections?

(c) Explain how to piece these bijections together to form a bijection between A and B.

(d) Justify the assumption that A and B are disjoint.

Problem 8.15. (a) Prove that if a nonempty set C is countable, then there is a *total* surjective function $f : \mathbb{N} \to C$.

(b) Conversely, suppose that $\mathbb{N}$ surj D, that is, there is a not necessarily total surjective function $f : \mathbb{N} D$. Prove that D is countable.

Problem 8.16.
There are lots of different sizes of infinite sets. For example, starting with the infinite set $\mathbb{N}$ of nonnegative integers, we can build the infinite sequence of sets

$$\mathbb{N} \text{ strict } \text{pow}(\mathbb{N}) \text{ strict } \text{pow}(\text{pow}(\mathbb{N})) \text{ strict } \text{pow}(\text{pow}(\text{pow}(\mathbb{N}))) \text{ strict } \ldots.$$

where each set is "strictly smaller" than the next one by Theorem 8.1.12. Let $\text{pow}^n(\mathbb{N})$ be the nth set in the sequence, and

$$U ::= \bigcup_{n=0}^{\infty} \text{pow}^n(\mathbb{N}).$$

(a) Prove that

$$U \text{ surj } \text{pow}^n(\mathbb{N}), \tag{8.10}$$

for all $n > 0$.

(b) Prove that

$$\text{pow}^n(\mathbb{N}) \text{ strict } U$$

for all $n \in \mathbb{N}$.

Now of course, we could take $U, \text{pow}(U), \text{pow}(\text{pow}(U)), \ldots$ and keep on in this way building still bigger infinities indefinitely.

Problem 8.17.
Let $\{0,1\}^\omega$ be the set of infinite binary sequences. Call a sequence in $\{0,1\}^\omega$ *lonely* if it never has two 1s in a row. For example, the repeating sequence

$$(0,1,0,1,0,1,0,1,0,\ldots)$$

is lonely, but the sequence

$$(0,0,1,1,0,1,0,1,0,0,0,1,\ldots)$$

is not lonely because it has two 1s next to each other. Let F be the set of lonely sequences. Show that F is uncountable.

Hint: Consider an "off-diagonal" argument as in Problem 8.20.

Homework Problems

Problem 8.18.
Prove that if A is an infinite set and B is a countable, then

$$A \text{ bij } (A \cup B).$$

Hint: Since A is infinite, we can find an infinite sequence $a_0, a_1, a_2, \ldots$ of distinct elements of A as in the proof of Lemma 8.1.7.

Problem 8.19.
In this problem you will prove a fact that may surprise you—or make you even more convinced that set theory is nonsense: the half-open unit interval is actually the "*same size*" as the nonnegative quadrant of the real plane![10] Namely, there is a bijection from $(0, 1]$ to $[0, \infty) \times [0, \infty)$.

(a) Describe a bijection from $(0, 1]$ to $[0, \infty)$.

Hint: $1/x$ almost works.

(b) An infinite sequence of the decimal digits $\{0, 1, \ldots, 9\}$ will be called *long* if it does not end with all 0's. An equivalent way to say this is that a long sequence is one that has infinitely many occurrences of nonzero digits. Let L be the set of all such long sequences. Describe a bijection from L to the half-open real interval $(0, 1]$.

Hint: Put a decimal point at the beginning of the sequence.

(c) Describe a surjective function from L to L^2 that involves alternating digits from two long sequences. *Hint:* The surjection need not be total.

(d) Prove the following lemma and use it to conclude that there is a bijection from L^2 to $(0, 1]^2$.
Lemma. *Let A and B be nonempty sets. If there is a bijection from A to B, then there is also a bijection from $A \times A$ to $B \times B$.*

(e) Conclude from the previous parts that there is a surjection from $(0, 1]$ to $(0, 1]^2$. Then appeal to the Schröder-Bernstein Theorem to show that there is actually a bijection from $(0, 1]$ to $(0, 1]^2$.

(f) Complete the proof that there is a bijection from $(0, 1]$ to $[0, \infty)^2$.

[10] The half-open unit interval $(0, 1]$ is $\{r \in \mathbb{R} \mid 0 < r \leq 1\}$. Similarly, $[0, \infty) ::= \{r \in \mathbb{R} \mid r \geq 0\}$.

Problem 8.20.

You don't really have to go down the diagonal in a "diagonal" argument.

Let's review the historic application of a diagonal argument to one-way infinite sequences

$$\langle e_0, e_1, e_2, \ldots, e_k, \ldots \rangle .$$

The angle brackets appear above as a reminder that the sequence is not a set: its elements appear in order, and the same element may appear multiple times.[11]

The general setup for a diagonal argument is that we have some sequence S whose elements are themselves one-way infinite sequences. We picture the sequence $S = \langle s_0, s_1, s_2, \ldots \rangle$ running vertically downward, and each sequence $s_k \in S$ running horizontally to the right:

$$s_k = \langle s_{k,0}, s_{k,1}, s_{k,2}, \ldots \rangle .$$

So we have a 2-D matrix that is infinite down and to the right:

	0	1	2	...	k ...
s_0	$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	...	$s_{0,k}$ ...
s_1	$s_{1,0}$	$s_{1,1}$	$s_{1,2}$	...	$s_{1,k}$ ...
s_2	$s_{2,0}$	$s_{2,1}$	$s_{2,2}$	...	$s_{2,k}$ ...
$\vdots$				$\vdots$	
s_k				...	$s_{k.k}$ ...

The diagonal argument explains how to find a "new" sequence, that is, a sequence that is not in S. Namely, create the new sequence by going down the diagonal of the matrix and, for each element encountered, add a differing element to the sequence being created. In other words, the diagonal sequence is

$$D_S ::= \langle \overline{s_{0,0}},\ \overline{s_{1,1}},\ \overline{s_{2,2}}, \ldots, \overline{s_{k,k}}, \ldots \rangle$$

where $\overline{x}$ indicates some element that is not equal to x. Now D_S is a sequence that is not in S because it differs from every sequence in S, namely, it differs from the kth sequence in S at position k.

For definiteness, let's say

$$\overline{x} ::= \begin{cases} 1 & \text{if } x \neq 1, \\ 2 & \text{otherwise.} \end{cases}$$

[11] The right angle-bracket is not really visible, since the sequence does not have a right end. If such one-way infinite sequences seem worrisome, you can replace them with total functions on $\mathbb{N}$. So the sequence above simply becomes the function e on $\mathbb{N}$ where $e(n) ::= e_n$.

With this contrivance, we have gotten the diagonal sequence D_S to be a sequence of 1's and 2's that is not in S.

But as we said at the beginning, you don't have to go down the diagonal. You could, for example, follow a line with a slope of $-1/4$ to get a new sequence

$$T_S ::= \langle \overline{s_{0,0}},\ t_1,\ t_2,\ t_3,\ \overline{s_{1,4}},\ t_5,\ t_6,\ t_7,\ \overline{s_{2,8}},\ \ldots,\ t_{4k-1},\ \overline{s_{k,4k}},\ t_{4k+1},\ \ldots \rangle$$

where the t_i's can be anything at all. Any such T_s will be a new sequence because it will differ from every row of the matrix, but this time it differs from the kth row at position $4k$.

(a) By letting all the t_i's be 2's, we get a new sequence T_S of 1's and 2's whose elements (in the limit) are at least three-quarters equal to 2. Explain how to find an *uncountable* number of such sequences of 1's and 2's.

Hint: Use slope $-1/8$ and fill six out of eight places with 2's. The abbreviation

$$2^{(n)} ::= \underbrace{2, 2, \ldots, 2}_{\text{length } n}$$

for a length-n sequence of 2's may be helpful, in particular $2^{(6)}$.

(b) Let's say a sequence has a *negligible fraction of non-2 elements* if, in the limit, it has a fraction of at most ϵ non-2 elements for *every* $\epsilon > 0$.[12] Describe how to define a sequence not in S that has a *negligible fraction of non-2 elements*.

(c) Describe how to find a sequence that differs *infinitely many times* from every sequence in S.

Hint: Divide $\mathbb{N}$ into an infinite number of non-overlapping infinite pieces.

Problem 8.21.

Cantor's Powerset Theorem 8.1.12 implies that for any set A, no total function $g : A \to \text{pow}(A)$ is a surjection. In particular, the theorem is proved by showing that the "diagonal" set

$$A_g ::= \{a \in A \mid a \notin g(a)\}$$

is an element of $\text{pow}(A)$ that is not in the range of g, because for every $a \in A$ it differs at the element a from the set $g(a)$ in the range of g.

[12] In other words, this means

$$\lim_{n\to\infty} (\text{number of non-2s among the first } n \text{ elements})/n = 0.$$

But there's no need to stick to the diagonal. If $p : A \to A$ is some total function, we can differ from $g(a)$ at $p(a)$ instead of "on the diagonal" at a. That is, define

$$A_{g,p} ::= \{p(a) \mid a \in A \text{ AND } p(a) \notin g(a)\}.$$

So $A_{g,p}$ is an element of $\text{pow}(A)$, and

False Claim. *$A_{g,p}$ is not in the range of g,*

because for every $a \in A$, the set $A_{g,p}$ differs at the element $p(a)$ from the set $g(a)$ in the range of g.

(a) Show that the claim is false by letting A be $\{0, 1\}$, and $g(n) = \{n\}$ and $p(n) = 0$ for all $n \in \mathbb{N}$.

(b) Identify the mistake in the argument above and show that it is fixed by requiring that p be an injection.

Problem 8.22.

Suppose $\mathcal{S} = \{S_0, S_1, \ldots\}$ is a countable set each of whose elements S_n is an infinite set of nonnegative integers. Using a diagonal argument, we can find a "new" infinite set U of nonnegative integers that is not in $\mathcal{S}$.

In this problem we describe how to find an infinite set U of nonnegative integers that not only is not in $\mathcal{S}$, but does not even have a subset that is in $\mathcal{S}$.

Rather than describing U directly, it's a little easier if we describe its complement C. Then U will be $\overline{C} ::= \mathbb{N} - C$.

Define a function $f : \mathbb{N} \to \mathbb{N}$ as follows:

$$f(n) ::= \min\{k \in S_n \mid k \geq n^2\},$$

and define

$$C ::= \text{range}(f).$$

(a) Prove that no subset of U is in $\mathcal{S}$.

(b) Show that the limiting density of U is one. That is,

$$\lim_{k\to\infty} \frac{|U \cap [0..k]|}{k} = 1.$$

Exam Problems

Problem 8.23. (a) For each of the following sets, indicate whether it is finite, countably infinite, or uncountable.

(i) The set of even integers greater than 10^{100}.

(ii) The set of "pure" complex numbers of the form ri for nonzero real numbers r.

(iii) The powerset of the integer interval $[10..10^{10}]$.

(iv) The complex numbers c such that c is the root of a quadratic with integer coefficients, that is,

$$\exists m, n, p \in \mathbb{Z}, m \neq 0.\ mc^2 + nc + p = 0.$$

Let $\mathcal{U}$ be an uncountable set, $\mathcal{C}$ be a countably infinite subset of $\mathcal{U}$, and $\mathcal{D}$ be a countably infinite set.

(v) $\mathcal{U} \cup \mathcal{D}$.

(vi) $\mathcal{U} \cap \mathcal{C}$

(vii) $\mathcal{U} - \mathcal{D}$

(b) Give an example of sets A and B such that

$$\mathbb{R} \text{ strict } A \text{ strict } B.$$

Problem 8.24.
Prove that if $A_0, A_1, \ldots, A_n, \ldots$ is an infinite sequence of countable sets, then so is

$$\bigcup_{n=0}^{\infty} A_n$$

Problem 8.25.
Let A and B be countably infinite sets:

$$A = \{a_0, a_1, a_2, a_3, \ldots\}$$
$$B = \{b_0, b_1, b_2, b_3, \ldots\}$$

Show that their product $A \times B$ is also a countable set by showing how to list the elements of $A \times B$. You need only show enough of the initial terms in your sequence to make the pattern clear—a half dozen or so terms usually suffice.

Problem 8.26.
Let $\{1,2,3\}^\omega$ be the set of infinite sequences containing only the numbers 1, 2, and 3. For example, some sequences of this kind are:

$$\langle 1,1,1,1\ldots\rangle,$$
$$\langle 2,2,2,2\ldots\rangle,$$
$$\langle 3,2,1,3\ldots\rangle.$$

Give two proofs that $\{1,2,3\}^\omega$ is uncountable:

(a) by showing $\{1,2,3\}^\omega$ surj S for some known uncountable set S, and

(b) by a direct diagonalization argument.

Problem 8.27.
An infinite binary string $b ::= b_0 b_1 b_2 \ldots b_n \ldots$ is called *OK* when the `1`'s in b occur only at perfect square positions. That is, b is OK when

$$b_i = \begin{cases} \texttt{0} & \text{if } i \notin \{0,1,4,9,\ldots,n^2,(n+1)^2\ldots\}, \\ \texttt{0 or 1} & \text{otherwise.} \end{cases}$$

(a) Prove that the set of OK strings is uncountable.

(b) Prove that a set with an uncountable subset must itself be uncountable.

(c) Let *Sparse* be the set of infinite binary strings whose fraction of `1`'s approaches zero. Conclude that Sparse is uncountable.

Problem 8.28.
A subset of the nonnegative integers $\mathbb{N}$ is called *lonely* when it doesn't contain any pair of consecutive integers. For example, the set $\{1,5,25,125,\ldots\}$ of powers of 5 is lonely, but the set of primes $\{2,3,5,7,11,\ldots\}$ is not lonely because it contains both 2 and 3.

Let L be the set of lonely subsets of $\mathbb{N}$. Show that L is uncountable.

Problem 8.29.
A function $f : \mathbb{N} \to \mathbb{N}$ is defined to *rapidly increasing* if

$$f(n+1) \geq 2^{f(n)}$$

for all $n \in \mathbb{N}$. Let Rapid be the set of rapidly increasing functions.

Use a *diagonalization argument* to prove that Rapid is uncountable.

Problem 8.30.

A real number is called *quadratic* when it is a root of a degree two polynomial with integer coefficients. Explain why there are only countably many quadratic reals.

Problem 8.31.

Describe which of the following twelve sets have bijections between them, with brief explanations.

$\mathbb{Z}$ (integers),	$\mathbb{R}$ (real numbers),
$\mathbb{C}$ (complex numbers),	$\mathbb{Q}$ (rational numbers),
pow($\mathbb{Z}$) (all subsets of integers),	pow($\emptyset$),
pow(pow($\emptyset$)),	$\{0,1\}^*$ (finite binary sequences),
$\{0,1\}^\omega$ (infinite binary sequences)	$\{\mathbf{T},\mathbf{F}\}$ (truth values)
pow($\{\mathbf{T},\mathbf{F}\}$),	pow($\{0,1\}^\omega$)

Problem 8.32.

Prove that the set $\left(\mathbb{Z}^+\right)^*$ of all finite sequences of positive integers is countable.

Hint: If $s \in \left(\mathbb{Z}^+\right)^*$, let sum($s$) be the sum of the successive integers in s.

Problems for Section 8.2

Class Problems

Problem 8.33.

Let $\mathbb{N}^\omega$ be the set of infinite sequences of nonnegative integers. For example, some sequences of this kind are:

$$(0,1,2,3,4,\ldots),$$
$$(2,3,5,7,11,\ldots),$$
$$(3,1,4,5,9,\ldots).$$

Prove that this set of sequences is uncountable.

Homework Problems

Problem 8.34.

For any sets A and B, let $[A \to B]$ be the set of total functions from A to B. Prove that if A is not empty and B has more than one element, then A strict $[A \to B]$.

Hint: Suppose that σ is a function from A to $[A \to B]$ mapping each element $a \in A$ to a function $\sigma_a : A \to B$. Suppose $b, c \in B$ and $b \neq c$. Then define

$$\text{diag}(a) ::= \begin{cases} c \text{ if } \sigma_a(a) = b, \\ b \text{ otherwise.} \end{cases}$$

Problem 8.35.

String procedures are one-argument procedures that apply to strings over the ASCII alphabet. If application of procedure P to string s results in a computation that eventually halts, we say that P *recognizes* s. We define $\text{lang}(P)$ to be the set of strings or *language* recognized by P:

$$\text{lang}(P) ::= \{s \in \text{ASCII}^* \mid P \text{ recognizes } s\}.$$

A language is *unrecognizable* when it is not equal to $\text{lang}(P)$ for any procedure P.

A string procedure declaration is a text $s \in \text{ASCII}^*$ that conforms to the grammatical rules for programs. The declaration defines a procedure P_s, which we can think of as the result of compiling s into an executable object. If $s \in \text{ASCII}^*$ is not a grammatically well-formed procedure declaration, we arbitrarily define P_s to be the string procedure that fails to halt when applied to any string. Now every string defines a string procedure, and every string procedure is P_s for some $s \in \text{ASCII}^*$.

An easy diagonal argument in Section 8.2 showed that

$$\text{No-halt} ::= \{s \mid P_s \text{ applied to } s \text{ does not halt}\} = \{s \mid s \notin \text{lang}(P_s)\}$$

is not recognizable.

It may seem pretty weird to apply a procedure to its own declaration. Are there any less weird examples of unrecognizable set? The answer is "many more." In this problem, we'll show three more:

$$\begin{aligned} \text{No-halt-}\lambda &::= \{s \mid P_s \text{ applied to } \lambda \text{ does not halt}\} = \{s \mid \lambda \notin \text{lang}(P_s)\}, \\ \text{Finite-halt} &::= \{s \mid \text{lang}(P_s) \text{ is finite}\}, \\ \text{Always-halt} &::= \{s \mid \text{lang}(P_s) = \text{ASCII}^*\}. \end{aligned}$$

Let's begin by showing how we could use a recognizer for No-halt-λ to define a recognizer for No-halt. That is, we will "reduce" the weird problem of recognizing

No-halt to the more understandable problem of recognizing No-halt-λ. Since there is no recognizer for No-halt, it follows that there can't be one for No-halt-λ either.

Here's how this reduction would work: suppose we want to recognize when a given string s is in No-halt. Revise s to be the declaration of a slightly modified procedure $P_{s'}$ which behaves as follows:

> $P_{s'}$ applied to argument $t \in \text{ASCII}^*$, ignores t, and simulates P_s applied to s.

So, if P_s applied to s halts, then $P_{s'}$ halts on every string it is applied to, and if P_s applied to s does not halt, then $P_{s'}$ does not halt on any string it is applied to. That is,

$$\begin{aligned} s \in \text{No-halt} &\text{ IMPLIES } \text{lang}(P_{s'}) = \emptyset \\ &\text{IMPLIES } \lambda \notin \text{lang}(P_{s'}) \\ &\text{IMPLIES } s' \in \text{No-halt-}\lambda, \\ s \notin \text{No-halt} &\text{ IMPLIES } \text{lang}(P_{s'}) = \text{ASCII}^* \\ &\text{IMPLIES } \lambda \in \text{lang}(P_{s'}) \\ &\text{IMPLIES } s' \notin \text{No-halt-}\lambda. \end{aligned}$$

In short,

$$s \in \text{No-halt IFF } s' \in \text{No-halt-}\lambda.$$

So to recognize when $s \in$ No-halt all you need to do is recognize when $s' \in$ No-halt-λ. As already noted above (but we know that remark got by several students, so we're repeating the explanation), this means that if No-halt-λ was recognizable, then No-halt would be as well. Since we know that No-halt is unrecognizable, then No-halt-λ must also be unrecognizable, as claimed.

(a) Conclude that Finite-halt is unrecognizable.

Hint: Same s'.

Next, let's see how a reduction of No-halt to Always-halt would work. Suppose we want to recognize when a given string s is in No-halt. Revise s to be the declaration of a slightly modified procedure $P_{s''}$ which behaves as follows:

> When $P_{s''}$ is applied to argument $t \in \text{ASCII}^*$, it simulates P_s applied to s for up to $|t|$ "steps" (executions of individual machine instructions). If P_s applied to s has *not* halted in $|t|$ steps, then the application of $P_{s''}$ to t halts. If P_s applied to s *has* halted within $|t|$ steps, then the application of $P_{s''}$ to t runs forever.

(b) Conclude that Always-halt is unrecognizable.

Hint: Explain why

$$s \in \text{No-halt IFF } s'' \in \text{Always-halt}.$$

(c) Explain why $\overline{\text{Finite-halt}}$ is unrecognizable.

Hint: Same s''.

Note that it's easy to recognize when P_s does halt on s: just simulate the application of P_s to s until it halts. This shows that $\overline{\text{No-halt}}$ *is* recognizable. We've just concluded that Finite-halt is nastier: neither it nor its complement is recognizable.

Problems for Section 8.3

Practice Problems

Problem 8.36.
Let P be any set predicate, and define $\mathcal{S}_P$ as the collection of all sets satisfying P:

$$\mathcal{S}_P ::= \{S \in \text{Sets} \mid P(S)\}.$$

(a) Give an example P such that $\mathcal{S}_P$ is not a set.

(b) Give an example P such that $\mathcal{S}_P$ is a set.

(c) Prove that if $\mathcal{S}_P$ is a set, then $P(\mathcal{S}_P)$ must be false.

Problem 8.37.
We know that every pure set is a recursive set Recset Theorem 8.3.2, so why we can't dispense with the axioms of set theory (ZFC) and just use the recursive definition to define sets?

Class Problems

Problem 8.38.
Forming a pair (a, b) of items a and b is a mathematical operation that we can safely take for granted. But when we're trying to show how all of mathematics can be reduced to set theory, we need a way to represent the pair (a, b) as a set.

(a) Explain why representing (a, b) by $\{a, b\}$ won't work.

(b) Explain why representing (a, b) by $\{a, \{b\}\}$ won't work either. *Hint:* What pair does $\{\{1\}, \{2\}\}$ represent?

(c) Define

$$\text{pair}(a, b) ::= \{a, \{a, b\}\}.$$

Explain why representing (a, b) as $\text{pair}(a, b)$ uniquely determines a and b. *Hint:* Sets can't be indirect members of themselves: $a \in a$ never holds for any set a, and neither can $a \in b \in a$ hold for any b.

Problem 8.39.

The axiom of choice says that if s is a set whose members are nonempty sets that are *pairwise disjoint*—that is, no two sets in s have an element in common—then there is a set c consisting of exactly one element from each set in s.

In formal logic, we could describe s with the formula,

$$\begin{aligned}\text{pairwise-disjoint}(s) ::= &\forall x \in s.\, x \neq \emptyset \text{ AND} \\ &\forall x, y \in s.\, x \neq y \text{ IMPLIES } x \cap y = \emptyset.\end{aligned}$$

Similarly we could describe c with the formula

$$\text{choice-set}(c, s) ::= \quad \forall x \in s.\, \exists! z.\, z \in c \cap x.$$

Here "$\exists!\, z$." is fairly standard notation for "there exists a *unique* z."

Now we can give the formal definition:

Definition (Axiom of Choice)**.**

$$\forall s.\, \text{pairwise-disjoint}(s) \text{ IMPLIES } \exists c.\, \text{choice-set}(c, s).$$

The only issue here is that set theory is technically supposed to be expressed in terms of *pure* formulas in the language of sets, which means formula that uses only the membership relation $\in$ propositional connectives, the two quantifies $\forall$ and $\exists$, and variables ranging over all sets. Verify that the axiom of choice can be expressed as a pure formula, by explaining how to replace all impure subformulas above with equivalent pure formulas.

For example, the formula $x = y$ could be replaced with the pure formula $\forall z.\, z \in x \text{ IFF } z \in y$.

Problem 8.40.

Let $R : A \to A$ be a binary relation on a set A. If $a_1 \; R \; a_0$, we'll say that a_1 is "R-smaller" than a_0. R is called *well founded* when there is no infinite "R-decreasing" sequence:

$$\cdots R\, a_n \; R \cdots R\, a_1 \; R\, a_0, \tag{8.11}$$

of elements $a_i \in A$.

For example, if $A = \mathbb{N}$ and R is the $<$-relation, then R is well founded because if you keep counting down with nonnegative integers, you eventually get stuck at zero:

$$0 < \cdots < n-1 < n.$$

But you can keep counting up forever, so the $>$-relation is not well founded:

$$\cdots > n > \cdots > 1 > 0.$$

Also, the $\leq$-relation on $\mathbb{N}$ is not well founded because a *constant* sequence of, say, 2's, gets $\leq$-smaller forever:

$$\cdots \leq 2 \leq \cdots \leq 2 \leq 2.$$

(a) If B is a subset of A, an element $b \in B$ is defined to be *R-minimal in B* iff there is no R-smaller element in B. Prove that $R : A \to A$ is well founded iff every nonempty subset of A has an R-minimal element.

A logic *formula of set theory* has only predicates of the form "$x \in y$" for variables x, y ranging over sets, along with quantifiers and propositional operations. For example,

$$\text{isempty}(x) ::= \forall w.\ \text{NOT}(w \in x)$$

is a formula of set theory that means that "x is empty."

(b) Write a formula member-minimal(u, v) of set theory that means that u is $\in$-minimal in v.

(c) The Foundation axiom of set theory says that $\in$ is a well founded relation on sets. Without looking it up in the text, express the Foundation axiom as a formula of set theory.

You may use "member-minimal" and "isempty" in your formula as abbreviations for the formulas defined above.

(d) Explain why the Foundation axiom implies that no set is a member of itself.

Homework Problems

Problem 8.41.
In writing formulas, it is OK to use abbreviations introduced earlier (so it is now legal to use "=" because we just defined it).

(a) Explain how to write a formula, $\text{Subset}_n(x, y_1, y_2, \ldots, y_n)$, of set theory [13] that means $x \subseteq \{y_1, y_2, \ldots, y_n\}$.

(b) Now use the formula Subset_n to write a formula, $\text{Atmost}_n(x)$, of set theory that means that x has at most n elements.

(c) Explain how to write a formula Exactly_n of set theory that means that x has exactly n elements. Your formula should only be about twice the length of the formula Atmost_n.

(d) The direct way to write a formula $D_n(y_1, \ldots, y_n)$ of set theory that means that $y_1, \ldots, y_n$ are distinct elements is to write an AND of subformulas "$y_i \neq y_j$" for $1 \leq i < j \leq n$. Since there are $n(n-1)/2$ such subformulas, this approach leads to a formula D_n whose length grows proportional to n^2. Describe how to write such a formula $D_n(y_1, \ldots, y_n)$ whose length only grows proportional to n.

Hint: Use Subset_n and Exactly_n.

Problem 8.42.
In this problem, structural induction provides a simple proof about some utterly infinite objects.

Every pure set defines a two-person game in which a player's move consists of choosing any element of the game. The two players alternate moves, and a player loses when it is their turn to move and there is no move to make. That is, whoever moves to the empty set is a winner, because the next player has no move.

So we think of a set R as the initial "board position" of a *set game*. The player who goes first in R is called the *Next* player, and the player who moves second in R is called the *Previous* player. When the Next player moves to an $S \in R$, the game continues with the new set game S in which the Previous player moves first.

Prove by structural induction on the Definition 8.3.1 of recursive sets Recset that for every set game, either the Previous player or the Next player has a winning strategy.[14]

[13] See Section 8.3.2.

[14] Set games are called "uniform" because the two players have the *same* objective: to leave the other player stuck with no move to make. In more general games, the two players have different

Exam Problems

Problem 8.43. (a) Explain how to write a formula Members(p, a, b) of set theory[15] that means $p = \{a, b\}$.

Hint: Say that everything in p is either a or b. It's OK to use subformulas of the form "$x = y$," since we can regard "$x = y$" as an abbreviation for a genuine set theory formula.

A *pair* (a, b) is simply a sequence of length two whose first item is a and whose second is b. Sequences are a basic mathematical data type we take for granted, but when we're trying to show how all of mathematics can be reduced to set theory, we need a way to represent the ordered pair (a, b) as a set. One way that will work[16] is to represent (a, b) as

$$\text{pair}(a, b) ::= \{a, \{a, b\}\}.$$

(b) Explain how to write a formula Pair(p, a, b), of set theory [17] that means $p = \text{pair}(a, b)$.

Hint: Now it's OK to use subformulas of the form "Members(p, a, b)."

(c) Explain how to write a formula Second(p, b), of set theory that means p is a pair whose second item is b.

objectives, for example, one wants to maximize the final payoff and the other wants to minimize it (Problem 7.35).

[15] See Section 8.3.2.

[16] Some similar ways that don't work are described in problem 8.38.

[17] See Section 8.3.2.