CS 70 Discrete Mathematics and Probability Theory
Summer 2014 James Cook **Note 19**

To Infinity And Beyond: Countability and Computability

This note ties together two topics that might seem like they have nothing to do with each other. The nature of infinity (and more particularly, the distinction between different levels of infinity) and the fundamental nature of computation and proof. This note can only scratch the surface — if you want to understand this material more deeply, there are wonderful courses in the Math department as well as EECS172 and graduate courses like EECS229A that will connect this material to the nature of information and compression as well.

Cardinality

How can we determine whether two sets have the same *cardinality* (or "size")? The answer to this question, reassuringly, lies in early grade school memories: by demonstrating a *pairing* between elements of the two sets. More formally, we need to demonstrate a *bijection* f between the two sets. The bijection sets up a one-to-one correspondence, or pairing, between elements of the two sets. We know how this works for finite sets. In this lecture, we will see what it tells us about *infinite* sets.

Are there more natural numbers $\mathbf{N}$ than there are positive integers $\mathbf{Z}^+$? It is tempting to answer yes, since every positive integer is also a natural number, but the natural numbers have one extra element $0 \notin \mathbf{Z}^+$. Upon more careful observation, though, we see that we can generate a mapping between the natural numbers and the positive integers as follows:

$$\begin{array}{lccccccccc}
\mathbf{N} & 0 & 1 & 2 & 3 & 4 & 5 & \dots & & \\
\downarrow & & \searrow & \searrow & \searrow & \searrow & \searrow & \searrow & & \\
\mathbf{Z}^+ & & 1 & 2 & 3 & 4 & 5 & 6 & \dots &
\end{array}$$

Why is this mapping a bijection? Clearly, the function $f : \mathbf{N} \to \mathbf{Z}^+$ is onto because every positive integer is hit. And it is also one-to-one because no two natural numbers have the same image. (The image of n is $f(n) = n+1$, so if $f(n) = f(m)$ then we must have $n = m$.) Since we have shown a bijection between $\mathbf{N}$ and $\mathbf{Z}^+$, this tells us that there are as many natural numbers as there are positive integers! Informally, we have proved that "$\infty + 1 = \infty$."

What about the set of *even* natural numbers $2\mathbf{N} = \{0, 2, 4, 6, \dots\}$? In the previous example the difference was just one element. But in this example, there seem to be twice as many natural numbers as there are even natural numbers. Surely, the cardinality of $\mathbf{N}$ must be larger than that of $2\mathbf{N}$ since $\mathbf{N}$ contains all of the odd natural numbers as well. Though it might seem to be a more difficult task, let us attempt to find a bijection between the two sets using the following mapping:

$$\begin{array}{lccccccc}
\mathbf{N} & 0 & 1 & 2 & 3 & 4 & 5 & \dots \\
\downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \\
2\mathbf{N} & 0 & 2 & 4 & 6 & 8 & 10 & \dots
\end{array}$$

The mapping in this example is also a bijection. f is clearly one-to-one, since distinct natural numbers get mapped to distinct even natural numbers (because $f(n) = 2n$). f is also onto, since every n in the range is hit: its pre-image is $\frac{n}{2}$. Since we have found a bijection between these two sets, this tells us that in fact $\mathbf{N}$ and $2\mathbf{N}$ have the same cardinality!

What about the set of all integers, $\mathbf{Z}$? At first glance, it may seem obvious that the set of integers is larger than the set of natural numbers, since it includes negative numbers. However, as it turns out, it is possible to find a bijection between the two sets, meaning that the two sets have the same size! Consider the following mapping:

$0 \to 0,\ 1 \to -1,\ 2 \to 1,\ 3 \to -2,\ 4 \to 2, \ldots, 124 \to 62, \ldots$

In other words, our function is defined as follows:

$$f(x) = \begin{cases} \frac{x}{2}, & \text{if } x \text{ is even} \\ \frac{-(x+1)}{2}, & \text{if } x \text{ is odd} \end{cases}$$

We will prove that this function $f : \mathbf{N} \to \mathbf{Z}$ is a bijection, by first showing that it is one-to-one and then showing that it is onto.

Proof (one-to-one): Suppose $f(x) = f(y)$. Then they both must have the same sign. Therefore either $f(x) = \frac{x}{2}$ and $f(y) = \frac{y}{2}$, or $f(x) = \frac{-(x+1)}{2}$ and $f(y) = \frac{-(y+1)}{2}$. In the first case, $f(x) = f(y) \Rightarrow \frac{x}{2} = \frac{y}{2} \Rightarrow x = y$. Hence $x = y$. In the second case, $f(x) = f(y) \Rightarrow \frac{-(x+1)}{2} = \frac{-(y+1)}{2} \Rightarrow x = y$. So in both cases $f(x) = f(y) \Rightarrow x = y$, so f is injective.

Proof (onto): If $y \in \mathbf{Z}$ is non-negative, then $f(2y) = y$. Therefore, y has a pre-image. If y is negative, then $f(-(2y+1)) = y$. Therefore, y has a pre-image. Thus every $y \in \mathbf{Z}$ has a preimage, so f is onto.

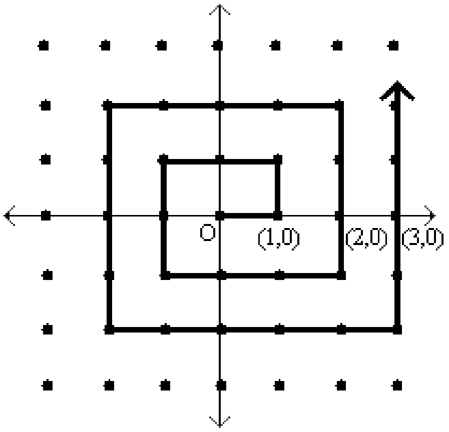
Since f is a bijection, this tells us that $\mathbf{N}$ and $\mathbf{Z}$ have the same size.

Now for an important definition. We say that a set S is *countably infinite* if there is a bijection between S and $\mathbf{N}$. We say a set is *countable* if it is finite or countably infinite. Thus any finite set S is countable (since finiteness means by definition that there is a bijection between S and the set $\{0, 1, 2, \ldots, m-1\}$, where $m = |S|$ is the size of S). And we have already seen three examples of countably infinite sets: $\mathbf{Z}^+$ and $2\mathbf{N}$ are obviously countable since they are themselves subsets of $\mathbf{N}$; and $\mathbf{Z}$ is countable because we have just seen a bijection between it and $\mathbf{N}$.

What about the set of all rational numbers? Recall that $\mathbf{Q} = \{\frac{x}{y} \mid x, y \in \mathbf{Z},\ y \neq 0\}$. Surely there are more rational numbers than natural numbers? After all, there are infinitely many rational numbers between any two natural numbers. Surprisingly, the two sets have the same cardinality! To see this, let us introduce a slightly different way of comparing the cardinality of two sets.

If there is a one-to-one function $f : A \to B$, then the cardinality of A is less than or equal to that of B. Now to show that the cardinality of A and B are the same we can show that $|A| \leq |B|$ and $|B| \leq |A|$. This corresponds to showing that there is a one-to-one function $f : A \to B$ and a one-to-one function $g : B \to A$. The existence of these two one-to-one functions implies that there is a bijection $h : A \to B$, thus showing that A and B have the same cardinality. The proof of this fact, which is called the Cantor-Bernstein theorem, is actually quite hard, and we will skip it here.

Back to comparing the natural numbers and the integers. First it is obvious that $|\mathbf{N}| \leq |\mathbf{Q}|$ because $\mathbf{N} \subseteq \mathbf{Q}$. So our goal now is to prove that also $|\mathbf{Q}| \leq |\mathbf{N}|$. To do this, we must exhibit an injection $f : \mathbf{Q} \to \mathbf{N}$. The following picture of a spiral conveys the idea of this injection:



Each rational number $\frac{a}{b}$ (written in its lowest terms, so that $\gcd(a,b) = 1$) is represented by the point (a,b) in the infinite two-dimensional grid shown (which corresponds to $\mathbf{Z} \times \mathbf{Z}$, the set of all pairs of integers). Note that not all points on the grid are valid representations of rationals: e.g, all points on the x-axis have $b = 0$ so none are valid (except for $(0,0)$, which we take to represent the rational number 0); and points such as $(2,8)$ and $(-1,-4)$ are not valid either as the rational number $\frac{1}{4}$ is represented by $(1,4)$. But $\mathbf{Z} \times \mathbf{Z}$ certainly contains all rationals under this representation, so if we come up with an injection from $\mathbf{Z} \times \mathbf{Z}$ to $\mathbf{N}$ then this will also be an injection from $\mathbf{Q}$ to $\mathbf{N}$ (why?).

The idea is to map each pair (a,b) to its position along the spiral, starting at the origin. (Thus, e.g, $(0,0) \to 0$, $(1,0) \to 1$, $(1,1) \to 2$, $(0,1) \to 3$, and so on.) This mapping certainly maps every rational number to a natural number, because every rational appears somewhere (exactly once) in the grid, and the spiral hits every point in the grid. Why is this mapping an injection? Well, we just have to check that no two rational numbers map to the same natural number. But that is true because no two pairs lie at the same position on the spiral. (Note that the mapping is *not* onto because some positions along the spiral do not correspond to valid representations of rationals; but that is fine.)

This tells us that $|\mathbf{Q}| \le |\mathbf{N}|$. Since also $|\mathbf{N}| \le |\mathbf{Q}|$, as we observed earlier, by the Cantor-Bernstein Theorem $\mathbf{N}$ and $\mathbf{Q}$ have the same cardinality.

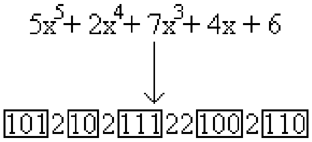
Our next example concerns the set of all binary strings (of any finite length), denoted $\{0,1\}^*$. Despite the fact that this set contains strings of unbounded length, it turns out to have the same cardinality as $\mathbf{N}$. To see this, we set up a direct bijection $f : \{0,1\}^* \to \mathbf{N}$ as follows. Note that it suffices to *enumerate* the elements of $\{0,1\}^*$ in such a way that each string appears exactly once in the list. We then get our bijection by setting $f(n)$ to be the nth string in the list. How do we enumerate the strings in $\{0,1\}^*$? Well, it's natural to list them in increasing order of length, and then (say) in *lexicographic* order (or, equivalently, numerically increasing order when viewed as binary numbers) within the strings of each length. This means that the list would look like

$$\varepsilon, 0, 1, 00, 01, 10, 11, 000, 001, 010, 011, 100, 101, 110, 111, 1000, \ldots,$$

where ε denotes the empty string (the only string of length 0). It should be clear that this list contains each binary string once and only once, so we get a bijection with $\mathbf{N}$ as desired.

Our final countable example is the set of all polynomials with natural number coefficients, which we denote $\mathbf{N}(x)$. To see that this set is countable, we will make use of (a variant of) the previous example. Note first that, by essentially the same argument as for $\{0,1\}^*$, we can see that the set of all *ternary* strings $\{0,1,2\}^*$ (that is, strings over the alphabet $\{0,1,2\}$) is countable. To see that $\mathbf{N}(x)$ is countable, it therefore suffices to exhibit an injection $f : \mathbf{N}(x) \to \{0,1,2\}^*$, which in turn will give an injection from $\mathbf{N}(x)$ to $\mathbf{N}$. (It is obvious that there exists an injection from $\mathbf{N}$ to $\mathbf{N}(x)$, since each natural number n is itself trivially a polynomial, namely the constant polynomial n itself.)

How do we define f? Let's first consider an example, namely the polynomial $p(x) = 5x^5 + 2x^4 + 7x^3 + 4x + 6$. We can list the coefficients of $p(x)$ as follows: $(5,2,7,0,4,6)$. We can then write these coefficients as binary strings: $(101_2, 10_2, 111_2, 0_2, 100_2, 110_2)$. Now, we can construct a ternary string where a "2" is inserted as a separator between each binary coefficient (ignoring coefficients that are 0). Thus we map $p(x)$ to a ternary string as illustrated below:

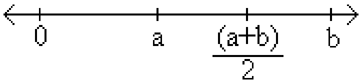


It is easy to check that this is an injection, since the original polynomial can be uniquely recovered from this ternary string by simply reading off the coefficients between each successive pair of 2's. (Notice that this mapping $f : \mathbf{N}(x) \to \{0,1,2\}^*$ is not onto (and hence not a bijection) since many ternary strings will not be the image of any polynomials; this will be the case, for example, for any ternary strings that contain binary subsequences with leading zeros.)

Hence we have an injection from $\mathbf{N}(x)$ to $\mathbf{N}$, so $\mathbf{N}(x)$ is countable.

Cantor's Diagonalization

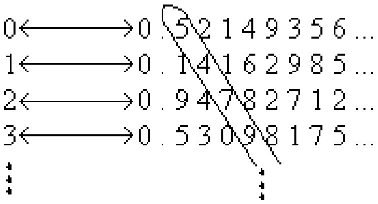
So we have established that **N**, **Z**, **Q** all have the same cardinality. What about the real numbers, the set of all points on the real line? Surely they are countable too? After all, the rational numbers, like the real numbers, are dense (i.e, between any two rational numbers there is a rational number):



In fact, between any two *real* numbers there is always a rational number. It is really surprising, then, that there are more real numbers than rationals. That is, there is no bijection between the rationals (or the natural numbers) and the reals. In fact, we will show something even stronger: even the real numbers in the interval $[0,1]$ are uncountable!

Recall that a real number can be written out in an infinite decimal expansion. A real number in the interval $[0,1]$ can be written as $0.d_1d_2d_3...$ Note that this representation is not unique; for example, $1 = 0.999...$; for definiteness we shall assume that every real number is represented as a recurring decimal where possible (for example, we choose the representation $0.999\ldots$ rather than 1 and $0.14999\ldots$ rather than 0.15).

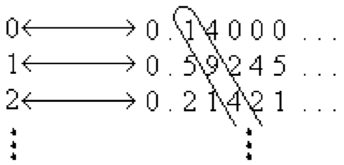
Cantor's Diagonalization Proof: Suppose towards a contradiction that there is a bijection $f : \mathbf{N} \to [0,1]$. Then, we can enumerate the infinite list as follows:



The number circled in the diagonal is some real number $r = 0.5479\ldots$, since it is an infinite decimal expansion. Now consider the real number s obtained by modifying every digit of r, say by replacing each digit d with $d + 2 \bmod 10$; thus in our example above, $s = 0.7691\ldots$. We claim that s does not occur in our infinite list of real numbers. Suppose for contradiction that it did, and that it was the n^{th} number in the list. Then r and s differ in the n^{th} digit: the n^{th} digit of s is the n^{th} digit of r plus 2 mod 10. So we have a real number s that is not in the range of f. But this contradicts the assertion that f is a bijection. Thus the real numbers are not countable.

Let us remark that the reason that we modified each digit by adding 2 mod 10 as opposed to adding 1 is that the same real number can have two decimal expansions; for example 0.999... = 1.000.... But if two real numbers differ by more than 1 in any digit they cannot be equal. Thus we are completely safe in our assertion.

With Cantor's diagonalization method, we proved that $\mathbf{R}$ is uncountable. What happens if we apply the same method to $\mathbf{Q}$, in a (futile) attempt to show the rationals are uncountable? Well, suppose for a contradiction that our bijective function $f : \mathbf{N} \to \mathbf{Q}[0,1]$ produces the following mapping:



This time, let us consider the number q obtained by modifying every digit of the diagonal, say by replacing each digit d with $d+2$ mod 10. Then in the above example q = 0.316..., and we want to try to show that it does not occur in our infinite list of rational numbers. However, we do not know if q is rational. This is why the method fails when applied to the rationals. When dealing with the reals, the modified diagonal number was guaranteed to be a real number.

Self-Reference and The Liar's Paradox

Propositions are statements that are either true or false. We saw before that some statements are not well defined or too imprecise to be called propositions. But here is a statement that is problematic for more subtle reasons: "All Cretans are liars." So said a Cretan in antiquity, thus giving rise to the so-called liar's paradox which has amused and confounded people over the centuries. Actually the above statement isn't really a paradox; it simply yields a contradiction if we assume it is true, but if it is false then there is no problem. A true formulation of this paradox is the following statement: "this statement is false." Is the statement true? If the statement is true, then what it asserts must be true; namely that it is false. But if it is false, then it must be true. So it really is a paradox. Around a century ago, this paradox found itself at the center of foundational questions about mathematics and computation.

We will now study how this paradox relates to computation. Before doing so, let us consider another manifestation of the paradox, created by the great logician Bertrand Russell. In a village with just one barber, every man keeps himself clean-shaven. Some of the men shave themselves, while others go to the barber. The barber proclaims: "I shave all and only those men who do not shave themselves." It seems reasonable then to ask the question: Does the barber shave himself? Thinking more carefully about the question though, we see that we are presented with a logically impossible scenario. If the barber does not shave himself, then according to what he announced, he shaves himself. If the barber does shave himself, then according to his statement he does not shave himself!

(Of course, the real solution to the barber paradox is simple: the barber is a woman.)

The Halting Problem

Are there tasks that a computer cannot perform? For example, we would like to ask the following basic question when compiling a program: does it go into an infinite loop? In 1936, Alan Turing showed that there is no program that can perform this test. The proof of this remarkable fact is very elegant and combines two ingredients: self-reference (as in the liar's paradox), and the fact that we cannot separate programs from data. In computers, a program is represented by a string of bits just as integers, characters, and other data are. The only difference is in how the string of bits is interpreted.

We will now examine the Halting Problem. Given the description of a program and its input, we would like to know if the program ever halts when it is executed on the given input. In other words, we would like to write a program TestHalt that behaves as follows:

$$\text{TestHalt(P,I)} = \begin{cases} \text{“yes”}, & \text{if program P halts on input I} \\ \text{“no”}, & \text{if program P loops on input I} \end{cases}$$

Why can't such a program exist? First, let us use the fact that a program is just a bit string, so it can be input as data. This means that it is perfectly valid to consider the behavior of TestHalt(P,P), which will output "yes" if P halts on P, and "no" if P loops forever on P. We now prove that such a program cannot exist.

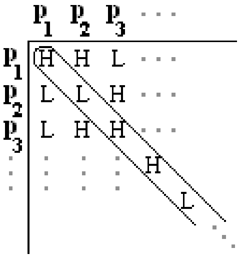
Proof: Define the program

```
Turing(P)
  if TestHalt(P,P) = "yes" then loop forever
  else halt
```

So if the program P when given P as input halts, then Turing loops forever; otherwise, Turing halts. Assuming we have the program TestHalt, we can easily use it as a subroutine in the above program Turing.

Now let us look at the behavior of Turing(Turing). There are two cases: either it halts, or it does not. If Turing(Turing) halts, then it must be the case that TestHalt(Turing, Turing) returned "no." But that would mean that Turing(Turing) should not have halted. In the second case, if Turing(Turing) does not halt, then it must be the case that TestHalt(Turing, Turing) returned "yes," which would mean that Turing(Turing) should have halted. In both cases, we arrive at a contradiction which must mean that our initial assumption, namely that the program TestHalt exists, was wrong. Thus, TestHalt cannot exist, so it is impossible for a program to check if any general program halts.

This is an example of diagonalization. Why? Since the set of all computer programs is countable (they are, after all, just finite-length strings over some alphabet, and the set of all finite-length strings is countable), we can enumerate all programs as follows (where P_i represents the i^{th} program):



	P_1	P_2	P_3	⋯
P_1	H	H	L	⋯
P_2	L	L	H	⋯
P_3	L	H	H	⋯
⋮	⋮	⋮	⋮	

The $(i,j)^{\text{th}}$ entry is H if program P_i halts on input P_j, and L if it does not halt. Now if the program Turing exists it must occur somewhere on our list of programs, say as P_n. But this cannot be, since if the n^{th} entry in the diagonal is H, meaning that P_n halts on P_n, then by its definition Turing loops on P_n; and if the entry is L, then by definition Turing halts on P_n. Thus the behavior of Turing is different from that of P_n, and hence Turing does not appear on our list. Since the list contains all possible programs, we must conclude that the program Turing does not exist. And since Turing is constructed by a simple modification of TestHalt, we can conclude that TestHalt does not exist either. Hence the Halting Problem cannot be solved.

In fact, there are many more cases of questions we would like to answer about a program, but cannot. For example, we cannot know if a program ever outputs anything or if it ever executes a specific line. We cannot even definitively check to see if the program is a virus.

Uncomputable numbers

The fact that the real numbers are uncountably infinite and that there are only a countable number of computer programs tells us that the vast majority of real numbers are fundamentally unknowable to computers. The halting problem above tells us that many of them are also interesting. For example, consider the real number between 0 and 1 whose ith binary digit is 0 if the ith computer program doesn't halt and is 1 if the ith computer program does halt. The halting problem argument tells us that this number is uncomputable.

In a very related proof, the logician Gödel showed that the following real number is also uncomputable. Consider all finite strings of mathematical symbols involving $\forall, \exists$, variables, as well as the arithmetic operations $+, *, -, /$ and exponentiation, comparisons $=, <, >$ and the logical operators $\neg, \wedge, \vee, \implies$. A string like that is either a syntax error or it is a valid proposition about the natural numbers. All such finite strings are certainly countable. So we can talk about the ith such string. Consider the real number between 0 and 1 in which the ith digit is 0 if the string is a syntax error or the proposition is false. The ith digit is 1 if the string is well-formed and the proposition it represents is true (i.e. there is no counterexample to it). The resulting real number is uncomputable.

This turns out to mean that there are true statements about the integers for which there is no proof[1]. In a very real sense, they just happen to be true for no good reason. The even more surprising consequence is that the same uncomputability result holds if we replace "true" and "false" with "provable" and "unprovable." Here, we can consider a proposition provable if it is either a syntax error (syntax can be checked by a program), has a counterexample, or has a valid proof. It is unprovable otherwise. So, not only are there lots of unprovable assertions out there, but they are impossible for a computer program to reliably recognize[2] as unprovable.

The proof of these facts is beyond the scope of the course, but is related to the deep connection between proofs and computation. Computation is a kind of living proof.

Advanced Material On Counting

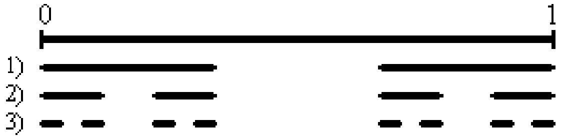
Please read on only if interested.

[1] This is assuming that (as many mathematicians believe) every mathematical statement must be true or false, even the ones we can't prove or disprove. A web search on the philosophy of mathematics will lead to some interesting reading.

[2] Being a valid proof can be checked by a computer program. Each statement has to follow logically from those that came before. The number of potential proofs is countably infinite. The problem is that the program that simply compares statements to all valid proofs is not guaranteed to halt! Because there are unprovable statements, it might just run forever and never a proof or counterexample.

The Cantor Set

The Cantor set is a remarkable set construction involving the real numbers in the interval $[0,1]$. The set is defined by repeatedly removing the middle thirds of line segments infinitely many times, starting with the original interval. For example, the first iteration would involve the removal of the interval $(\frac{1}{3},\frac{2}{3})$, leaving $[0,\frac{1}{3}]\cup[\frac{2}{3},1]$. The first three iterations are illustrated below:



The Cantor set contains all points that have *not* been removed: $C=\{x : x \text{ not thrown out}\}$. How much of the original unit interval is left after this process is repeated infinitely? Well, we start with 1, and after the first iteration we remove $\frac{1}{3}$ of the interval, leaving us with $\frac{2}{3}$. For the second iteration, we keep $\frac{2}{3}\times\frac{2}{3}$ of the original interval. As we repeat the iterations infinitely, we are left with:

$1 \longrightarrow \frac{2}{3} \longrightarrow \frac{2}{3}\times\frac{2}{3} \longrightarrow \frac{2}{3}\times\frac{2}{3}\times\frac{2}{3} \longrightarrow \cdots \longrightarrow \lim_{n\to\infty}(\frac{2}{3})^n = 0$

According to the calculations, we have removed everything from the original interval! Does this mean that the Cantor set is empty? No, it doesn't. What it means is that the *measure* of the Cantor set is zero; the Cantor set consists of isolated points and does not contain any non-trivial intervals. In fact, not only is the Cantor set non-empty, it is uncountable![3]

To see why, let us first make a few observations about ternary strings. In ternary notation, all strings consist of digits (called "trits") from the set $\{0,1,2\}$. All real numbers in the interval $[0,1]$ can be written in ternary notation. (E.g, $\frac{1}{3}$ can be written as 0.1, or equivalently as $0.0222\ldots$, and $\frac{2}{3}$ can be written as 0.2 or as $0.1222\ldots$.) Thus, in the first iteration, the middle third removed contains all ternary numbers of the form 0.1xxxxx. The ternary numbers left after the first removal can all be expressed either in the form 0.0xxxxx... or 0.2xxxxx... (We have to be a little careful here with the endpoints of the intervals; but we can handle them by writing $\frac{1}{3}$ as $0.02222\ldots$ and $\frac{2}{3}$ as 0.2.) The second iteration removes ternary numbers of the form 0.01xxxxx and 0.21xxxxx (i.e, any number with 1 in the second position). The third iteration removes 1's in the third position, and so on. Therefore, what remains is all ternary numbers with only 0's and 2's. Thus we have shown that

$$C=\{x\in[0,1] : x \text{ has a ternary representation consisting only of 0's and 2's}\}.$$

Finally, using this characterization, we can set up an *onto* map f from C to $[0,1]$. Since we already know that $[0,1]$ is uncountable, this implies that C is uncountable also. The map f is defined as follows: for $x\in C$, $f(x)$ is defined as the binary decimal obtained by dividing each digit of the ternary representation of x by 2. Thus, for example, if $x=0.0220$ (in ternary), then $f(x)$ is the binary decimal 0.0110). But the set of all binary decimals $0.\text{xxxxx}\ldots$ is in 1-1 correspondence with the real interval $[0,1]$, and the map f is onto because every binary decimal is the image of some ternary string under f (obtained by doubling every binary digit).[4] This completes the proof that C is uncountable.

[3] It's actually easy to see that C contains at least countably many points, namely the endpoints of the intervals in the construction—i.e, numbers such as $\frac{1}{3}$, $\frac{2}{3}$, $\frac{1}{9}$, $\frac{1}{27}$ etc. It's less obvious that C also contains various other points, such as $\frac{1}{4}$ and $\frac{3}{10}$. (Why?)

[4] Note that f is *not* injective; for example, the ternary strings $0.20222\ldots$ and 0.22 map to binary strings $0.10111\ldots$ and 0.11 respectively, which denote the same real number. Thus f is not a bijection. However, the current proof shows that the cardinality of C is at least that of $[0,1]$, while it is obvious that the cardinality of C is at most that of $[0,1]$ since $C\subset[0,1]$. Hence C has the same cardinality as $[0,1]$ (and as $\mathbf{R}$).

The Cantor set turns out to be an example of a set for which it is natural to define dimensionality differently. It isn't quite a 1d set (like a line segment) but it is certainly bigger than a 0d set (a countable collection of isolated points). This is an example of a set with "fractional dimension" — also referred to as fractals.

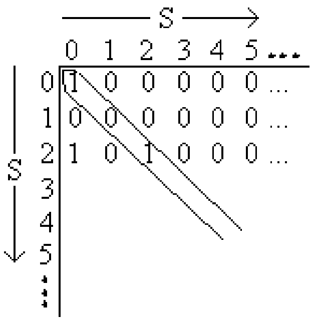
Power Sets and Higher Orders of Infinity

Let S be any set. Then the *power set* of S, denoted by $\mathscr{P}(S)$, is the set of all subsets of S. More formally, it is defined as: $\mathscr{P}(S) = \{T : T \subseteq S\}$. For example, if $S = \{1,2,3\}$, then $\mathscr{P}(S) = \{\{\},\{1\},\{2\},\{3\},\{1,2\},\{1,3\},\{2,3\},\{1,2,3\}\}$.

What is the cardinality of $\mathscr{P}(S)$? If $|S| = k$ is finite, then $|\mathscr{P}(S)| = 2^k$. To see this, let us think of each subset of S corresponding to a k bit string. In the example above the subset $\{1,3\}$ corresponds to the string 101. A 1 in the i^{th} position indicates that the i^{th} element of S is in the subset and a 0 indicates that it is not. Now the number of binary strings of length k is 2^k, since there are two choices for each bit position. Thus $|\mathscr{P}(S)| = 2^k$. So for finite sets S, the cardinality of the power set of S is exponentially larger than the cardinality of S. What about infinite (countable) sets? We claim that there is no bijection from S to $\mathscr{P}(S)$, so $\mathscr{P}(S)$ is not countable. Thus for example the set of all subsets of natural numbers is not countable, even though the set of natural numbers itself is countable.

Theorem: Let S be countably infinite. Then $|\mathscr{P}(S)| > |S|$.

Proof: Suppose towards a contradiction that there is a bijection $f : S \to \mathscr{P}(S)$. Recall that we can represent a subset by a binary string, with one bit for each element of S. (So, since S is infinite, the string will be infinitely long. Contrast the case of $\{0,1\}^*$ discussed earlier, which consists of all binary strings of *finite* length.) Consider the following diagonalization picture in which the function f maps natural numbers x to binary strings which correspond to subsets of S (e.g, 2 → 10100... = {0,2}):



	0	1	2	3	4	5	...
0	1	0	0	0	0	0	...
1	0	0	0	0	0	0	...
2	1	0	1	0	0	0	...
3							
4							
5							
⋮							

In this case, we have assigned the following mapping: 0 → {0}, 1 → {}, 2 → {0,2}, ... (i.e, the nth row describes the n^{th} subset as follows: if there is a 1 in the k^{th} column, then k is in this subset, else it is not.) Using a similar diagonalization argument to the earlier one, flip each bit along the diagonal: 1 → 0, 0 → 1, and let b denote the resulting binary string. First, we must show that the new element is a subset of S. Clearly it is, since b is an infinite binary string which corresponds to a subset of S. Now suppose b were the n^{th} binary string. This cannot be the case though, since the n^{th} bit of b differs from the n^{th} bit of the diagonal (the bits are flipped). So it's not on our list, but it should be, since we assumed that the list enumerated all possible subsets of S. Thus we have a contradiction, implying that $\mathscr{P}(S)$ is uncountable.

Thus we have seen that the cardinality of $\mathscr{P}(\mathbf{N})$ (the power set of the natural numbers) is strictly larger than the cardinality of $\mathbf{N}$ itself. The cardinality of $\mathbf{N}$ is denoted $\aleph_0$ (pronounced "aleph null"), while that of $\mathscr{P}(\mathbf{N})$ is denoted $2^{\aleph_0}$. It turns out that in fact $\mathscr{P}(\mathbf{N})$ has the same cardinality as $\mathbf{R}$ (the real numbers), and indeed as the real numbers in $[0,1]$. This cardinality is known as $\mathbf{c}$, the "cardinality of the continuum." So we know that $2^{\aleph_0} = \mathbf{c} > \aleph_0$. Even larger infinite cardinalities (or "orders of infinity"), denoted $\aleph_1, \aleph_2, \ldots$, can be defined using the machinery of set theory; these obey (to the uninitiated somewhat bizarre) rules of arithmetic. Several fundamental questions in modern mathematics concern these objects. For example, the famous “continuum hypothesis” asserts that $\mathbf{c} = \aleph_1$ (which is equivalent to saying that there are no sets with cardinality between that of the natural numbers and that of the real numbers).