

These notes are meant as a supplement to the material found at the end of chapters 2.4 and 3.1. Most of the basic results can be found there.

Theorem 1. *Two sets have the same size iff there is a bijection between them.*

Definition 1. *A set A is called countable iff $|A| \leq |\mathbb{N}|$. Otherwise, A is called uncountable.*

Definition 2. *A set A is called countably infinite iff it is countable and infinite*

Theorem 2. *The integers are countable.*

Proof. We will show $|\mathbb{Z}| = |\mathbb{N}|$ by exhibiting a bijection $f : \mathbb{Z} \rightarrow \mathbb{N}$.

$$\text{Let } f(n) = \begin{cases} 2n & n \geq 0 \\ 2(-n) - 1 & n < 0 \end{cases}$$

(f is injective) Note that f maps negative integers to odd numbers and positive integers to even numbers. Thus, if $f(n) = f(m)$, n and m are either both positive or both negative. In the first case, we have $2n = 2m$, so $n = m$. In the second, $2(-n) - 1 = 2(-m) - 1$, so $-2n = -2m$ and thus $n = m$. Thus, f is injective

(f is surjective) Let $n \in \mathbb{N}$. If n is even, then $\frac{n}{2}$ is a non-negative integer and $f(n/2) = n$. If n is odd, then $n + 1$ is even and positive so $-\frac{n+1}{2}$ is a negative integer and $f(-\frac{n+1}{2}) = 2(\frac{n+1}{2}) - 1 = n$. Thus, f is surjective.

$\therefore f$ is a bijection between \mathbb{Z} and \mathbb{N} and $|\mathbb{Z}| = |\mathbb{N}|$ □

Theorem 3. *The reals are uncountable*

Proof. In fact, we'll show that even the interval $[0, 1]$ is uncountable by showing that any function from $\mathbb{N} \rightarrow [0, 1]$ must not be a surjection. Suppose $f : \mathbb{N} \rightarrow [0, 1]$ were a surjection. Consider the real number r whose decimal expansion is $0.d_1d_2d_3d_4d_5\dots$. Where

$$d_n = \begin{cases} 5 & \text{if } n^{\text{th}} \text{ digit of } f(n) = 4 \\ 4 & \text{otherwise} \end{cases}$$

Note that $r \in [0, 1]$. However, we claim that there is no n such that $f(n) = r$. In particular, given any $n \in \mathbb{N}$, our construction ensured that the n^{th} digit of $f(n)$ is different than the n^{th} digit of r . Thus, $f(n) \neq r$.

Thus, there are no surjections from \mathbb{N} to $[0, 1]$, so $|[0, 1]| > |\mathbb{N}|$

Since $[0, 1] \subseteq \mathbb{R}$, we conclude that \mathbb{R} is uncountable. □

The following is one of the more surprising results you'll come across, and a good example of infinities are just not intuitive:

Theorem 4. *The rationals are countable*

Proof. We know that each rational number has a unique representation in the form $\frac{p}{q}$ where $p \in \mathbb{Z}, q \in \mathbb{N}, q \neq 0$ and p, q have no common factors. We can thus list all the rational numbers as follows: begin by listing all rationals whose representation has $|p|, |q| < 1$ in increasing order. Now list all rational numbers with $|p|, |q| < 2$ in increasing order, but do not repeat any from step 1. Now do the same with $|p|, |q| < 3$, etc. Note that each rational will eventually be listed, and will in fact be listed exactly once. Thus, the function $f : \mathbb{N} \rightarrow \mathbb{Q}$ given by $f(n) =$ the n^{th} number listed using the above algorithm is in fact a bijection.

$\therefore |\mathbb{Q}| = |\mathbb{N}|$ and \mathbb{Q} is countable. □

Using a similar technique, it's not hard to show that $|\mathbb{Z} \times \mathbb{Z}| = |\mathbb{N}|$. That is, there are the same number of pairs of integers as there are natural numbers.

The technique used to show \mathbb{R} is uncountable is called "diagonalization" because we make sure r and $f(n)$ differ on the "diagonal" of all the different decimal expansions. The following proof illustrates a similar technique:

Theorem 5. *For any set S , $|\mathcal{P}(S)| > |S|$*

Proof. Let S be any set. We'll show that there is no surjection from S to $\mathcal{P}(S)$. Let $f : S \rightarrow \mathcal{P}(S)$ be a function and consider the set

$$A = \{x \in S : x \notin f(x)\}$$

Note that $A \subseteq S$. We claim that there is no $s \in S$ such that $f(s) = A$, which we'll prove by contradiction. AFSOC that there is an s such that $f(s) = A$. Then if $s \in f(s)(= A)$, by the definition of A we would have $s \notin A$, which is a contradiction. Similarly, if $s \notin f(s)(= A)$, then by the definition of A we would have $s \in A$. Thus, in either case we get a contradiction and we conclude that no such s can exist.

\therefore there is no surjection from $\mathcal{P}(S)$ to S and we must have $|\mathcal{P}(S)| > |S|$

□

Note that Theorem 5 shows that there are infinitely many infinities, since we have

$$|\mathbb{N}| < |\mathcal{P}(\mathbb{N})| < |\mathcal{P}(\mathcal{P}(\mathbb{N}))| < |\mathcal{P}(\mathcal{P}(\mathcal{P}(\mathbb{N})))| < \dots$$

This raises the question of how exactly to name all these infinities, which we accomplish as follows:

Definition 3. The symbol \aleph_0 is used to denote $|\mathbb{N}|$. \aleph_1 denotes the cardinality (ie, size) of the smallest uncountable set. \aleph_2 is the size of the smallest set with size \aleph_1 , etc.

A problem that was open for many, many years was whether $|\mathbb{R}| = \aleph_1$ and is known as the Continuum Hypothesis. It is not hard to show that $|\mathbb{R}| = |\mathcal{P}(\mathbb{N})| = 2^{\aleph_0}$ but it in fact turns out to be **impossible** to decide whether or not this is the same \aleph_1 . That is, one can **prove** that there is no proof that $|\mathbb{R}| = \aleph_1$ and that there is no proof that $|\mathbb{R}| \neq \aleph_1$.

As another application of diagonalization, we'll prove the following amazing result:

Theorem 6. There is no computer program that takes in the source code to another program and an input to that program, and correctly determines whether the program ever finishes running on that input.

Proof. We'll do this by contradiction, so suppose there were such a program called $H(P, I)$ and suppose

$$H(P, I) = \begin{cases} 1 & P(I) \text{ eventually finishes} \\ 0 & \text{otherwise} \end{cases}$$

Since H is a program, we can run it from within other programs. Consider the program $K(x)$ which takes in some source code x and runs the following algorithm:

```
Run H(x, x).
If H(x, x) = 1, then loop forever.
Otherwise, stop running.
```

K is some program, so it has some source code and thus we can feed it to itself and consider what happens when we run $K(K)$. By the definition of K , we'll first run $H(K, K)$ which from the definition of H will determine whether $K(K)$ ever finishes. However, if H thinks $K(K)$ will finish, it will return 1, and the code for $K(K)$ will thus cause K to loop forever, making H incorrect. Similarly, if H thinks $K(K)$ will never finish, it will return 0 and $K(K)$ will immediately stop running, thus also making H incorrect. In either case, we get a contradiction, so H cannot exist. Thus, no program can correctly determine when other programs will finish running.

□

While the above may seem a little contrived, the following theorem (known as Gödel's First Incompleteness Theorem) is one of the cornerstones of Mathematical Logic. It is short, elegant, and is widely accepted as one of the most important results of the 20th century:

Theorem 7. There is no program that will correctly determine whether statements are true or not.

Proof. Suppose there were such a program and suppose its source code were P . Now consider the sentence "Program P thinks this sentence is false", which we'll call G for convenience. If in fact P does think G is false, then G is in fact true and thus P is wrong about it. If on the other hand P thinks G is true, then G is in fact false and P is wrong about it again. Thus, the program P cannot exist.

□

You might be a little sketched out by the self-reference in the sentence G , but it is in fact not hard (though slightly tedious) to show that this is not a problem at all. Take Math 125, a CS theory class, or ask me for some links to the result if you're curious.