Infinite Sets: Countability[1]

- **Examples of some infinite sets.** A set S is an infinite set if $|S| = \infty$. What does that mean? Well, it means that for *any* natural number N, one can find $> N$ distinct elements of S. Here are some examples of infinite sets we will see the next two lectures.
  - *The Naturals.* $\mathbb{N} = \{1, 2, 3, \ldots\}$
  - *The Integers.* $\mathbb{Z} = \{\ldots, -3, -2, -1, 0, 1, 2, 3, \ldots\} \;\; = \{x : x \in \mathbb{N}\} \cup \{-x : x \in \mathbb{N}\} \cup \{0\}$
  - *The Rationals.* $\mathbb{Q} = \{\frac{p}{q} : \;\; p \in \mathbb{Z}, q \in \mathbb{Z} \setminus \{0\}\}$
  - *The Reals.* $\mathbb{R}$. What are the reals? That is a deep question and forms the first few lectures of an Analysis course. For us, we will go with

    $$\mathbb{R} = \{ \;\sum_{i=0}^{\infty} \frac{a_i}{10^i}, \quad a_0 \in \mathbb{Z}, \quad 0 \le a_i \le 9, \; \forall i \ge 1\}$$

    The numbers $a_1, a_2, \ldots$ form the *decimal notation* of the number denoted as the summation.
  - *Python Programs.* $\mathcal{P}$. The set of all possible Python programs.
  - *The Strings.* Σ^*. The set of all strings formed by using letters from a *finite set* Σ.
  - *Boolean Functions.* $\mathcal{F}$. The set of all functions which assign each natural number a value either 0 or 1.

    $$\mathcal{F} := \{f : \mathbb{N} \to \{0, 1\}\}$$

    For instance the $\mathsf{isPrime}(n)$ function is an element of $\mathcal{F}$.

  There are two main points to this and the next lecture.

  - There are many kinds of infinities (we will see two).
  - The cardinalities of the set of Boolean functions, and the set of Python programs are *different*! Thus, there ***must*** exists functions which have no programs.

  And then, lastly, we will see an *explicit* problem which cannot have any algorithm.

- **Recall.** A function $f : A \to B$ is an *injection* if for any two distinct $a_1 \neq a_2$ in A, we have $f(a_1) \neq f(a_2)$.

- **Countable Sets.** A set S is called *countable* if there exists an injection $f : S \to \mathbb{N}$.

  It is called so because the elements of S can be ordered and counted one at a time (although the counting may never finish).

  More precisely, using f one can devise an algorithm which for any natural number k gives the kth number in the ordering, such that for every $s \in S$, there is some k such that

  The following code prints this sequence.

[1] *Lecture notes by Deeparnab Chakrabarty. Last modified : 28th Aug, 2021*
These have not gone through scrutiny and may contain errors. If you find any, or have any other comments, please email me at deeparnab@dartmouth.edu. Highly appreciated!

1: **procedure** ORDERSET(k)
2: ▷ *Returns the kth element of S given by an injection $f : S \to \mathbb{N}$.*
3: ▷ *Assumes $k \geq 1$ is natural, and $k \leq |S|$.*
4: count $\leftarrow 0$; $n \leftarrow 1$
5: **while** count $< k$ **do**:
6: **if** there exists some $s \in S$ such that $f(s) = n$ **then**: ▷ *i.e.* $f^{-1}(n) \in S$
7: count $\leftarrow$ count $+ 1$ ▷ *Hit someone in S. Increment* count.
8: $s^* \leftarrow s$. ▷ s^* *is the* (count)*th element in the sequence.*
9: $n \leftarrow n + 1$. ▷ *Move to the next element in $\mathbb{N}$.*
10: **return** s^* ▷ *Since we exit the loop when* count $= k$, s^* *is the kth element.*

The next two theorems show that the above algorithm indeed returns a valid ordering. We need to show two things: (a) termination, and (b) every element in S is indeed in the order. We did not cover this in class. Make sure you agree.

Theorem 1. The algorithm ORDERSET always terminates for any $k \in \mathbb{N}$ if $|S| \geq k$.

Proof. Since $|S| > k$, there exists a subset $\{s_1, \ldots, s_k\}$ of S of some k different elements of S. Let $N := \max_{i=1}^{k} f(s_i)$. Note that this is well defined since we are taking a maximum over a finite number of elements. We assert that by N rounds of the while loop, the above algorithm will terminate. Indeed, by N rounds, the algorithms will encounters $f(s_1), f(s_2), \ldots, f(s_k)$ and the count would reach k. It may terminate even earlier since there may be some $s' \notin \{s_1, \ldots, s_k\}$ with $f(s') < N$ which may increase count. But definitely by the Nth while loop the algo would terminate. □

Theorem 2. For every element $s \in S$, there is some $k \in \mathbb{N}$ s.t. ORDERSET(k) returns s.

Proof. Let $N = f(s)$. We claim that for one $k \leq N$, ORDERSET(k) must return s. Why? If not, let $s_1, s_2, \ldots, s_N$ be the N elements of S returned by ORDERSET(k) for $1 \leq k \leq N$. Firstly, we claim these s_i's are different, and indeed $f(s_1) < f(s_2) < \cdots < f(s_N)$. This is because if $k < \ell$, then the n in Line 9 is incremented more in the run for ℓ than the run for k. But all these numbers must be in $\{1, 2, \ldots, N-1\}$ if none of the s_i's are s; we don't skip any natural number n. This is a contradiction. □

Remark: *Different injective functions can lead to different orderings. But the important fact is that **any** countable set can be ordered into a sequence.*

- **Examples of Countable Sets.**
  - *Finite sets* are trivially countable. If a set S is finite and $|S| = k$, then the elements of S can be renamed as $\{e_1, e_2, \ldots, e_k\}$. The injective function $f(e_i) = i$ implies S is countable.
  - $\mathbb{N}$ is countable by definition. But there are many more interesting examples.

– *Set of Integers.* The set $\mathbb{Z}$ is countable. To see this, consider the following function $f : \mathbb{Z} \to \mathbb{N}$. If $x > 0$, then $f(x) = 2x$. If $x \leq 0$, then $f(x) = 2(-x) + 1$. Note that the co-domain of this function is indeed the natural numbers.

For instance, $f(2) = 4$, $f(-2) = 5$, and $f(0) = 1$.

Claim 1. The function $f : \mathbb{Z} \to \mathbb{N}$ defined above is injective.

Proof. To see this is injective, we need to show $f(x) \neq f(y)$ for two integers $x \neq y$. We may assume, without loss of generality, $x < y$. If both x and y are positive, then $f(x) = 2x < 2y = f(y)$. Similarly, if both are non-negative, then we get $f(x) = -2x + 1 > -2y + 1 = f(y)$. The only other case is x is non-negative and y is positive. In this case, $f(x)$ is odd while $f(y)$ is even. □

If we use the above algorithm to figure out the ordering of $\mathbb{Z}$, we get:

$$(0, 1, -1, 2, -2, 3, -3, 4, -4, \cdots)$$

- **Some operations that preserve countability.**

Theorem 3. If S is countable, and $T \subseteq S$, then T is countable.

Proof. If $f : S \to \mathbb{N}$ is an injection, then the restriction of f to T, that is, $g : T \to \mathbb{N}$ defined as $g(t) = f(t)$ is also an injection. □

Theorem 4. If S is countable and T is countable and $S \cap T = \emptyset$, then $S \cup T$ is countable.

Proof. We can use the trick for showing integers are countable.

Let $f : S \to \mathbb{N}$ be the injective function and $g : T \to \mathbb{N}$ be the injective function which show they are countable. We now define a function $h : S \cup T \to \mathbb{N}$ which is injective. Indeed,

$$h(x) = \begin{cases} 2f(x) & \text{if } x \in S. \\ 2g(x) + 1 & \text{if } x \in T. \end{cases}$$

To prove this is an injective function, take any two $a \neq b$ in $S \cup T$. Either both are in S, in which case $h(a) = 2f(a) \neq 2f(b) = h(b)$ where $f(a) \neq f(b)$ for f is an injection. Similarly, if both are in T, then $h(a) \neq h(b)$. If one is in S and the other is in T, then $h(a)$ (if $a \in S$) is even while $h(b)$ is odd. Thus, $h(a) \neq h(b)$ here as well. □

Theorem 5. If there is a function $g : A \to B$ which is an injection, and the set B is countable, then the set A is countable.

Proof. Since B is countable, there is an injective function $f : B \to \mathbb{N}$. We claim that the function $(f \circ g)$ is an injective function from A to $\mathbb{N}$. Indeed, if $a \neq a'$, then $g(a) \neq g(a')$. Let $b = g(a)$ and $b' = g(a')$. We get $(f \circ g)(a) = f(b)$ and $(f \circ g)(a') = f(b')$. Since $b \neq b'$, we get $(f \circ g)(a) \neq (f \circ g)(a')$.

□

- **The Set of Rationals is Countable.** This may be a surprise since the set of rationals are dense, that is, between any two rational numbers, there is a rational number. Nevertheless, they are countable.

  To show this, we need to construct an injection $g : \mathbb{Q} \to \mathbb{N}$. For now, we only show an injection of $g : \mathbb{Q}_+ \to \mathbb{N}$ where $\mathbb{Q}_+$ are all the positive rationals; we leave the extension to the full set of rationals as an exercise.

  This can be defined as follows: given any positive rational number $z = p/q$ in the *reduced form* (that is, $\gcd(p, q) = 1$), define

  $$z = p/q \qquad\qquad g : z \mapsto 2^p 3^q$$

  Clearly, the functions maps a positive rational number to a positive integer.

  We claim that the above function $g : \mathbb{Q}_+ \to \mathbb{N}$ is injective. To see this, pick two different positive rationals $x = p/q$ and $y = r/s$ such that $x \neq y$. We need to prove $g(x) \neq g(y)$, that is, $2^p 3^q \neq 2^r 3^s$.

  Since $x \neq y$, we have $p \neq r$, or $q \neq s$, or both. If $p \neq r$, then the largest power of 2 dividing $g(x)$ and $g(y)$ are different, implying $g(x) \neq g(y)$. If $q \neq s$, then the largest power of 3 dividing $g(x)$ and $g(y)$ are different, implying $g(x) \neq g(y)$. In either case, $g(x) \neq g(y)$.

  Exercise: *Extend the above proof to give an injection $g : \mathbb{Q} \to \mathbb{N}$. Hint: use the fact that the union of two countable sets is countable.*

  Exercise: *What ordering of the (positive) rationals does the above give using the algorithm for getting ordering from the injective function? Order the first 7 positive rationals.*

- **The Set of Python Programs is Countable.**

  Indeed, we show the set of strings Σ^* over any finite alphabet Σ is countable. Since $\mathcal{P} \subseteq \Sigma^*$ for Σ given by all the < 200 symbols on your keyboard, Theorem 3 would show $\mathcal{P}$ is countable.

  To do this, for any $n \in \mathbb{N} \cup \{0\}$, let us define $\Sigma_n \subseteq \Sigma^*$ be the collection of all strings over the alphabet Σ which have ***exactly*** length n. Clearly,

  $$\Sigma^* = \Sigma_0 \cup \Sigma_1 \cup \Sigma_2 \cup \cdots = \bigcup_{n=0}^{\infty} \Sigma_n$$

  Observation. For any fixed n, the set Σ_n is indeed a *finite* set. Indeed, it has size exactly $|\Sigma|^n$ which is a large but finite number. And therefore, since finite sets are countable, there is at least one injective function

  $$f_n : \Sigma_n \to \mathbb{N}$$

  For instance, one could look at the *alphabetical ordering* of strings in Σ_n. This is well defined since Σ_n is finite.

  And now we are ready to define the injective mapping h from Σ^* to $\mathbb{N}$ using the same idea as in rationals. Given any $\sigma \in \Sigma^*$, define

  $$h : \sigma \mapsto 2^{|\sigma|} \cdot 3^{f_{|\sigma|}(\sigma)}$$

  That is, if $|\sigma| = n$ where $n \in \mathbb{N} \cup \{0\}$, then we map σ to $2^n \cdot 3^{f_n(\sigma)}$.

To see which this is an injection, let us select $\sigma \neq \sigma'$ in Σ^*.

We claim this is an injection. To see this, take $\sigma \neq \sigma'$.

Case 1: $|\sigma| \neq |\sigma'|$. In this case the largest power of 2 dividing $g(\sigma)$ and $g(\sigma')$ are different, and thus the two numbers must be different.

Case 2: $|\sigma| = |\sigma'| = n$. In this case, both lie in Σ_n implying $f_n(\sigma) \neq f_n(\sigma')$. Thus, the largest power of 3 dividing $g(\sigma)$ and $g(\sigma')$ are different, and thus the two numbers must be different.

- **Where are we headed?** The fact that $\mathcal{P}$ is countable will lead us to the notion of "uncomputable" functions. What does that mean? For this we need to define what a computable function is. We will do so rather informally (and please take CS39 to get the rigorous version of computability) by saying

  A function $f : \mathbb{N} \to \{0,1\}$ is computable if there is a python code C taking input an number and outputting 0 or 1, such that for every $n \in \mathbb{N}$, we have $C(n) = f(n)$.

Theorem 6. If every function in $\mathcal{F}$ were computable, then $\mathcal{F}$ would be a countable set.

Proof. We describe an injective map from $\mathcal{F}$ to $\mathcal{P}$; we would be done by Theorem 5.

Indeed, gven a function $f \in \mathcal{F}$, since it is computable, there is a code $C \in \mathcal{P}$ which computes it. We claim for two $f \neq f' \in \mathcal{F}$ we can't have the same code C. Indeed, if $f \neq f'$, there exists some $n \in \mathbb{N}$ such that $f(n) \neq f'(n)$. But both are $C(n)$. Contradiction. □

Next lecture, we show $\mathcal{F}$ is ***uncountable***. And thus, there must exist uncomputable functions.