

Comparing Sets by Size

- A mapping, or function, from a set A to a set B is said to be a *one-to-one correspondence* if it is one-to-one and onto.
- A one-to-one correspondence associates with each element of A a unique element of B , and vice versa.
- Two sets A and B are said to be *equinumerous* (or of the same size) if, and only if, there is a one-to-one correspondence from A to B .
- For example, the two sets $\{0, 1\}$ and $\{a, b\}$ are of the same size, but $\{0, 1\}$ and $\{0, 1, 2\}$ are not.
- The set of natural numbers is equinumerous with the set of positive integers. The function f , defined by $f(n) = n + 1$, is a one-to-one correspondence.

Finite and Infinite Sets

- Finite sets are equinumerous with some initial segment, $\{0, 1, \dots, n\}$, of the natural numbers.
- If a set A is equinumerous with the set \mathbb{N} of all natural numbers it is said to be *countably infinite*.
- Examples of countably infinite sets are the natural numbers, the positive integers, the negative integers, and the set of all integers.
- A set is called *countable* if it is finite or countably infinite; and *uncountable* otherwise.
- Informally, a set is countable if its elements can be listed in a (finite or infinite) sequence,

$$a_0, a_1, a_2, \dots$$

Subsets

- **Theorem**

Every subset of a countable set is countable.

- *Proof.* Let A be a countable set and B be a subset of A .

- If B is finite, it is countable by definition.
- If B is infinite, so is A .
- Since A is countable its elements can be listed in some sequence,

$$a_0, a_1, a_2, \dots$$

- The elements of B can be listed using the same sequence, but skipping elements not in B .

Cartesian Products

- **Theorem**

The Cartesian product of two countable sets is countable.

- *Proof.* We sketch the basic idea for pairs of natural numbers, which can be listed in sequence as follows:

- Begin the list with the pair $(0,0)$ and set k to be 0.
- Repeat the following step.
- Increase k by one and list all pairs where the two components add up to k . There are only finitely many such pairs, and they can be listed in any order, e.g., according to the first component.

- This list is obtained by traversing the following table diagonal by diagonal:

$(0,0)$	$(0,1)$	$(0,2)$	$(0,3)$	$(0,4)$...
$(1,0)$	$(1,1)$	$(1,2)$	$(1,3)$	$(1,4)$...
$(2,0)$	$(2,1)$	$(2,2)$	$(2,3)$	$(2,4)$...
$(3,0)$	$(3,1)$	$(3,2)$	$(3,3)$	$(3,4)$...
\vdots	\vdots	\vdots	\vdots		

Examples

- The set of integers \mathbf{Z} is countable. Here is a one-to-one correspondence:

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

- Surprisingly, the set of rational numbers \mathbf{Q} is also countable:
 - The set of integers \mathbf{Z} is countable.
 - Hence by the above theorem, the set $\mathbf{Z} \times \mathbf{Z}$ is countable.
 - The set of rational numbers \mathbf{Q} corresponds to a subset of $\mathbf{Z} \times \mathbf{Z}$, and therefore is also countable.

Strings

- **Theorem**

The set Σ^* of all strings over an alphabet Σ is countable.

- *Proof.* We use the fact that

$$\Sigma^* = \bigcup_{k \in \mathbb{N}} \Sigma^k,$$

where Σ^k denotes the set of all strings of length k , to list strings in sequence as follows:

- Begin with the empty string ϵ and set k to be 0.
- Repeat the following step.
- Increase k by one and list all strings in Σ^k . There are only finitely many such strings, and they can be listed in any order, e.g., lexicographically.

Diagonalization

- Diagonalization is a proof technique with applications in the theory of computation.

- **The Diagonalization Principle**

Let R be a binary relation on a set A , and let D , the *diagonal set* for R , be $\{a \in A : (a, a) \notin R\}$. Furthermore, for each $a \in A$, let R_a be the set $\{b \in A : (a, b) \in R\}$.

Then the set D is distinct from each set R_a .

- For example, let R be the binary relation

$\{(a, b), (a, d), (b, b), (b, c), (c, c), (d, b), (d, c), (d, e), (e, e)\}$.

- We have

$$R_a = \{b, d\}$$

$$R_b = \{b, c\}$$

$$R_c = \{c\}$$

$$R_d = \{b, c, e\}$$

$$R_e = \{e\}$$

and $D = \{a, d\}$.

Example

- The relation R above can be represented by a table:

	a	b	c	d	e
a		x		x	
b		x	x		
c			x		
d		x	x		x
e					x

- The sets R_x correspond to the rows of the table.
- The set D represents the complement of the diagonal (from upper left to lower right):

x			x	
---	--	--	---	--

- Evidently, the complement of the diagonal is different from each row.

Uncountable Sets

- **Theorem**

The powerset of \mathbf{N} is uncountable.

- *Proof (by contradiction).*

- Suppose $\mathcal{P}(\mathbf{N})$ is countable. Then there is a one-to-one correspondence f between \mathbf{N} and $\mathcal{P}(\mathbf{N})$.

- Define a binary relation

$$R = \{(i, j) \in \mathbf{N} \times \mathbf{N} : j \in f(i)\}.$$

- The set R_i , as defined in the statement of the Diagonalization Principle, is equal to the set $f(i)$. In other words, each subset of \mathbf{N} is equal to one of the sets R_i .

- Now consider the diagonal set for R ,

$$D = \{n \in \mathbf{N} : n \notin R_n\}.$$

By the Diagonalization Principle, the set D is distinct from each set R_i .

- But D is a subset of \mathbf{N} , and since f is a one-to-one correspondence between \mathbf{N} and $\mathcal{P}(\mathbf{N})$, we must have $D = R_k$ for some k .

- In short, the assumption that $\mathcal{P}(\mathbf{N})$ is countable leads to a contradiction. Thus, $\mathcal{P}(\mathbf{N})$ is uncountable.

Formal Languages

- **Theorem**

The set $\mathcal{P}(\Sigma^*)$ of all formal languages is uncountable.

- *Proof.* The set $\mathcal{P}(\Sigma^*)$ is equinumerous with the uncountable set $\mathcal{P}(\mathbf{N})$.

- **Theorem**

The set of Turing-recognizable languages is countable.

- *Proof.* A language is Turing-recognizable if it is accepted by a Turing machine, but the set of all Turing machines is countable.

- **Corollary**

The set of recursive languages is countable.

The Halting Problem

- The acceptance problem for Turing machines is represented by the following formal language:

$$A_{TM} = \{\langle M, w \rangle : M \text{ is a TM that accepts } w\}.$$

- The set A_{TM} is Turing-recognizable, or recursively enumerable: a universal Turing machine U accepts A_{TM} .
- A closely related problem is the *Halting Problem*, which is represented by the following set:

$$HALT_{TM} = \{\langle M, w \rangle : M \text{ is a TM that halts for } w\}.$$

- The Halting Problem is also recursively enumerable.

An Undecidable Problem

- **Theorem**

The set A_{TM} is not recursive.

- *Proof.* The proof uses diagonalization.

- Suppose A_{TM} is recursive and let H be a decider for A_{TM} .
- The TM H accepts $\langle M, w \rangle$ if M accepts w , and rejects $\langle M, w \rangle$ if M does not accept w .
- Let D be a TM that, for input $\langle M \rangle$, runs H on input $\langle M, \langle M \rangle \rangle$ and outputs the opposite of what H outputs.
- Thus, D accepts $\langle M \rangle$ if M does not accept $\langle M \rangle$; and D rejects $\langle M \rangle$ if M accepts $\langle M \rangle$.
- Consequently, D accepts $\langle D \rangle$ if it does not accept $\langle D \rangle$; and rejects $\langle D \rangle$ if it accepts $\langle D \rangle$.
- This is a contradiction. We conclude that A_{TM} is not recursive.

Recursive Enumerability

- **Theorem**

The class of recursive languages is a proper subset of the class of recursively enumerable languages.

- *Proof.* The set A_{TM} is recursively enumerable, but not recursive.

- **Theorem**

A set is recursive if, and only if, both the set and its complement are recursively enumerable.

- **Theorem**

The class of recursively enumerable languages is not closed under complement.

- *Proof.* The set A_{TM} is recursively enumerable, but not recursive. Consequently, its complement cannot be recursively enumerable.