BU CS 332 – Theory of Computation

https://forms.gle/1Gr9hdWCUw12UKdg7

Lecture 13:

- More decidable languages
- Universal Turing Machine
- Countability

Reading:

Sipser Ch 4.1, 4.2

HW 6 Problem 5 &
bonus delayed to HW7

Mark Bun

March 17, 2025

Last Time

Church-Turing Thesis

v1: The basic TM (and all equivalent models) capture our intuitive notion of algorithms

v2: Any physically realizable model of computation can be simulated by the basic TM

Decidable languages (from language theory)

$A_{\mathrm{DFA}} = \{\underbrace{\langle D, w \rangle}_{\text{"Encoding" of DFA } D \text{ + string } w} \mid \mathrm{DFA}\ D \text{ accepts input } w\}$, etc.

Today: More decidable languages

Are there undecidable languages? How can we prove so?

A “universal” algorithm for recognizing regular languages

$A_{\mathrm{DFA}} = \{\langle D, w\rangle \mid \text{DFA } D \text{ accepts } w\}$

Theorem: A_{DFA} is decidable

Proof: Define a (high-level) 3-tape TM M on input $\langle D, w\rangle$:

1. Check if $\langle D, w\rangle$ is a valid encoding (reject if not)
2. Simulate D on w, i.e.,
   - Tape 2: Maintain w and head location of D
   - Tape 3: Maintain state of D, update according to δ
3. Accept if D ends in an accept state, reject otherwise

Regular Languages are Decidable

Theorem: Every regular language L is decidable

Proof 1: If L is regular, it is recognized by a DFA D. Convert this DFA to a TM M. Then M decides L.

Proof 2: If L is regular, it is recognized by a DFA D. The following TM M_D decides L.

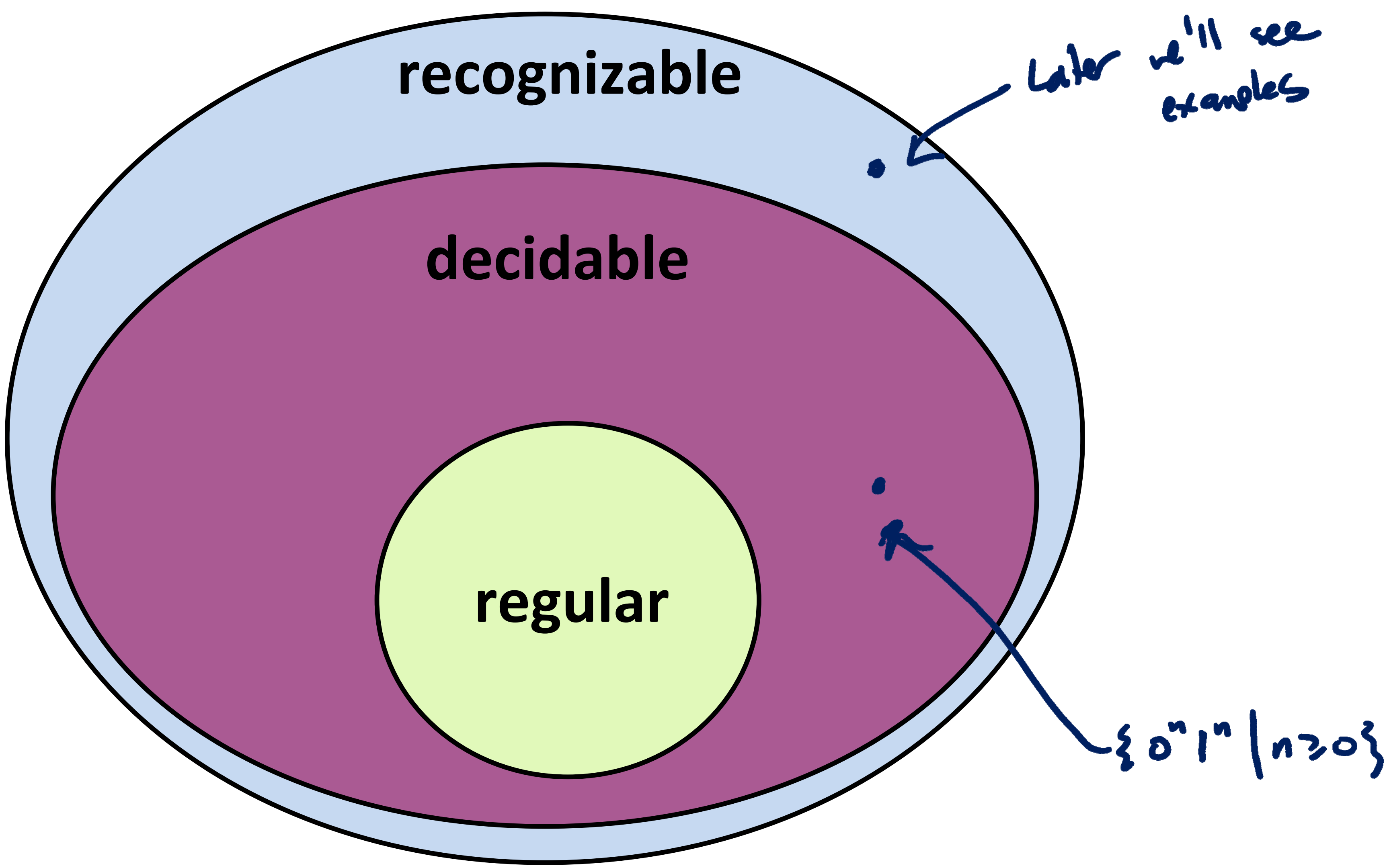
On input w:

1. Run the decider for A_{DFA} on input $\langle D, w\rangle$
2. Accept if the decider accepts; reject otherwise

$= \{\langle D,w\rangle \mid \text{DFA } D \text{ accepts } w\}$

Correctness:
* If $w \in L$, D accepts $w \Rightarrow \langle D,w\rangle \in A_{DFA} \Rightarrow$ TM accepts
* If $w \notin L$, D rejects $w \Rightarrow \langle D,w\rangle \notin A_{DFA} \Rightarrow$ TM rejects

Classes of Languages



More Decidable Languages: Emptiness Testing

Theorem: $E_{\text{DFA}} = \{\langle D \rangle \mid D \text{ is a DFA such that } L(D) = \emptyset\}$ is decidable

Computational Problem: Given DFA D, does D recognize the empty language?

Proof: The following TM decides E_{DFA}

On input $\langle D \rangle$, where D is a DFA with $\boxed{k}$ states:

1. Perform $\boxed{k}$ steps of breadth-first search on state diagram of D to determine if an accept state is reachable from the start state

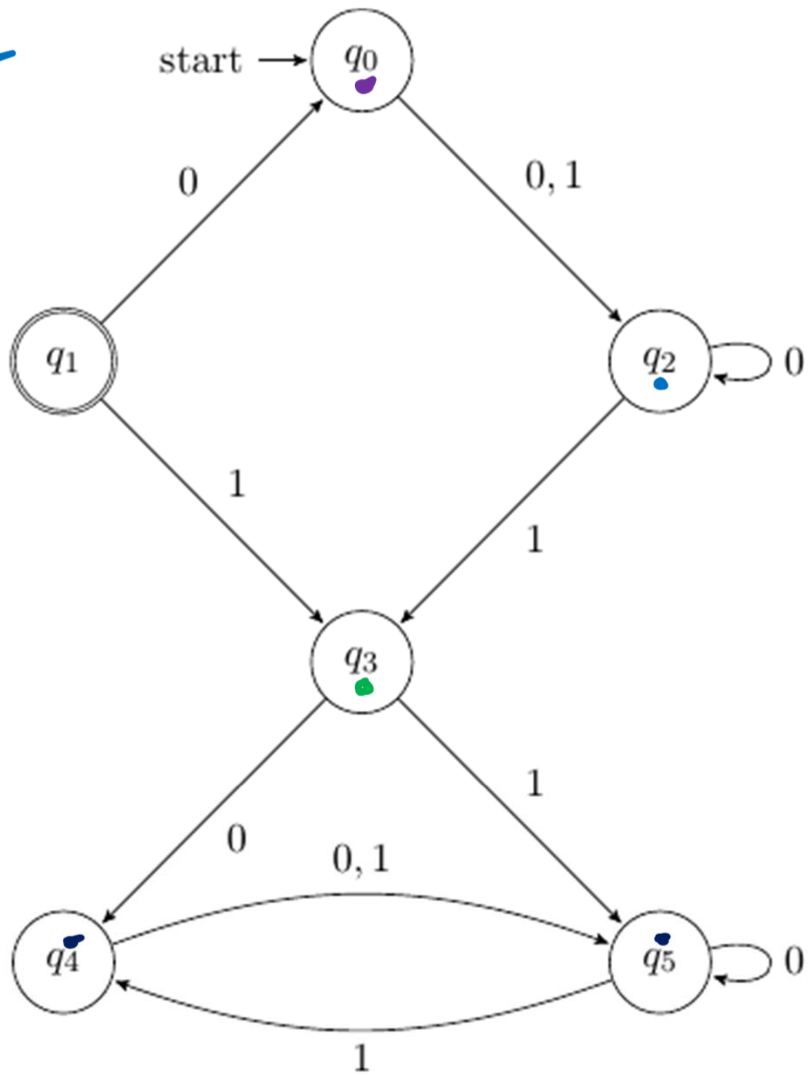
2. Reject if a DFA accept state is reachable; accept otherwise

Proof of correctness:

- If $\langle D \rangle \in E_{DFA}$, $L(D) = \emptyset$ $\Rightarrow$ not possible to reach an accept state from start state of D.
  $\Rightarrow$ not possible " " w/in k steps
  $\Rightarrow$ BFS does not reach an accept state $\Rightarrow$ TM accepts
- If $\langle D \rangle \notin E_{DFA}$, $L(D) \neq \emptyset$ $\Rightarrow$ $\exists$ a path from start state of D to an accept state
  $\Rightarrow$ $\exists$ a path from start to accept of length $\leq k$
  $\Rightarrow$ BFS finds path $\Rightarrow$ TM rejects.

E_{DFA} Example

D =



BFS Search

1) q_0

2) q_2

3) q_3

4) q_4, q_5

5) ... no more states reachable

BFS did not hit an accept state

$\Rightarrow$ TM accept (i.e., $\langle D \rangle \in E_{DFA}$)

New Deciders from Old: Equality Testing

$EQ_{\text{DFA}} = \{\langle D_1, D_2 \rangle \mid D_1, D_2 \text{ are DFAs and } L(D_1) = L(D_2)\}$

Problem: Given DFAs D_1, D_2, do they recognize the same language?

Theorem: EQ_{DFA} is decidable

Proof: The following TM decides EQ_{DFA}

On input $\langle D_1, D_2 \rangle$, where D_1, D_2 are DFAs:

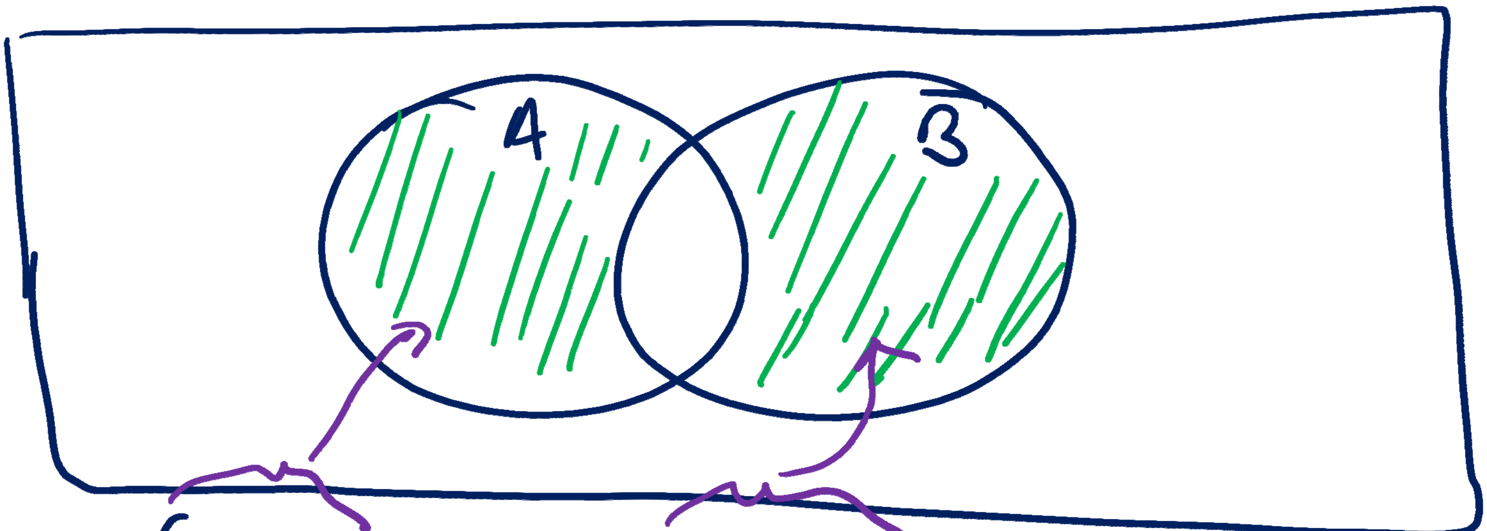
1. Construct DFA D recognizing the **symmetric difference** $L(D_1) \triangle L(D_2)$ $= \{ w \mid (w \in L(D_1) \text{ and } w \notin L(D_2)) \text{ OR } (w \in L(D_2) \text{ and } w \notin L(D_1)) \}$
   $\{ w \mid w \text{ is in exactly one of } L(D_1), L(D_2) \}$
2. Run the decider for E_{DFA} on $\langle D \rangle$ and return its output

Proof of correctness:

- If $\langle D_1, D_2 \rangle \in EQ_{\text{DFA}} \Rightarrow L(D_1) = L(D_2) \Rightarrow$ There is no string w that is in exactly one of $L(D_1)$, $L(D_2)$
  $\Rightarrow L(D_1) \triangle L(D_2) = \emptyset$
  $\Rightarrow L(D) = \emptyset$ by construction $\Rightarrow$ TM accepts
- If $\langle D_1, D_2 \rangle \notin EQ_{\text{DFA}} \Rightarrow L(D_1) \neq L(D_2) \Rightarrow \exists$ a string w in exactly one of $L(D_1)$, $L(D_2)$
  $\Rightarrow L(D_1) \triangle L(D_2) \neq \emptyset \Rightarrow$ TM rejects

Symmetric Difference

$$A \triangle B = \{w \mid w \in A \text{ or } w \in B \text{ but not both}\}$$



$$A \triangle B = (A \setminus B) \cup (B \setminus A) = (A \cap \overline{B}) \cup (B \cap \overline{A})$$

Know regular languages are closed under intersection, union, complement, and moreover, given DFAs for A, B, can construct DFAs recognizing these new languages.

On a TM

Universal Turing Machine

Meta-Computational Languages

$A_{\text{DFA}} = \{\langle D, w\rangle \mid \text{DFA } D \text{ accepts } w\}$

$A_{\text{TM}} = \{\langle M, w\rangle \mid \text{TM } M \text{ accepts } w\}$

$E_{\text{DFA}} = \{\langle D\rangle \mid \text{DFA } D \text{ recognizes the empty language } \emptyset\}$

$E_{\text{TM}} = \{\langle M\rangle \mid \text{TM } M \text{ recognizes the empty language } \emptyset\}$

$EQ_{\text{DFA}} = \{\langle D_1, D_2\rangle \mid D_1 \text{ and } D_2 \text{ are DFAs}, L(D_1) = L(D_2)\}$

$EQ_{\text{TM}} = \{\langle M_1, M_2\rangle \mid M_1 \text{ and } M_2 \text{ are TMs}, L(M_1) = L(M_2)\}$

The Universal Turing Machine

$A_{\mathrm{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM that accepts input } w\}$

Theorem: A_{TM} is Turing-recognizable

- Problem: Given TM M and string w, does M accept w?

The following "Universal TM" U recognizes A_{TM}

On input $\langle M, w\rangle$:

1. Simulate running M on input w
2. If M accepts, accept. If M rejects, reject.

Proof of Correctness:

- If $\langle M, w\rangle \in A_{\mathrm{TM}}$, M accepts w $\Rightarrow$ U accepts ✓
- If $\langle M, w\rangle \notin A_{\mathrm{TM}}$, M does not accept w $\Rightarrow$ U does not accept ✓

Universal TM and A_{TM}

Why is the Universal TM **not** a decider for A_{TM}?

The following "Universal TM" U recognizes A_{TM}

TM U:

TM string

On input $\langle M, w \rangle$:

1. Simulate running M on input w
2. If M accepts, accept. If M rejects, reject.

a) It may reject inputs $\langle M, w \rangle$ where M accepts w

b) It may accept inputs $\langle M, w \rangle$ where M rejects w

c) It may loop on inputs $\langle M, w \rangle$ where M loops on w

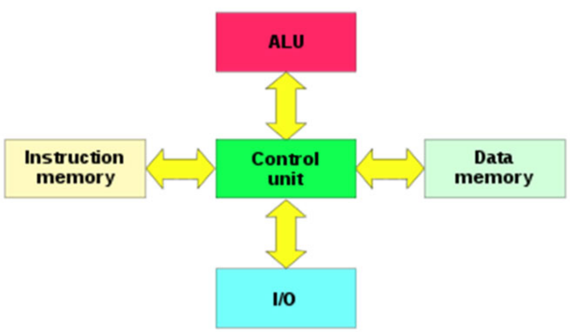
d) It may loop on inputs $\langle M, w \rangle$ where M accepts w

More on the Universal TM

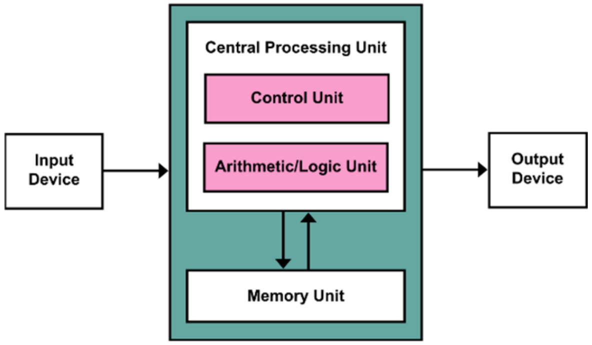
"It is possible to invent a single machine which can be used to compute any computable sequence. If this machine **U** is supplied with a tape on the beginning of which is written the S.D ["standard description"] of some computing machine **M**, then **U** will compute the same sequence as **M**."

- Turing, "On Computable Numbers…" 1936

- Foreshadowed general-purpose programmable computers
- No need for specialized hardware: Virtual machines as software



Harvard architecture:
Separate instruction and data pathways



von Neumann architecture:
Programs can be treated as data

Undecidability

A_{TM} is Turing-recognizable via the Universal TM

there is no Turing machine that decides each of these languages

…but it turns out A_{TM} (and $E_{\mathrm{TM}}, EQ_{\mathrm{TM}}$) is **undecidable**

i.e., computers cannot solve these problems no matter how much time they are given

How can we prove this?

… but first, a math interlude

Countability and Diagonalization

What's your intuition?

Which of the following sets is the "biggest"?

a) The natural numbers: $\mathbb{N} = \{1, 2, 3, \dots\}$

$POW2 \subseteq E \subseteq \mathbb{N}$

b) The even numbers: $E = \{2, 4, 6, \dots\}$

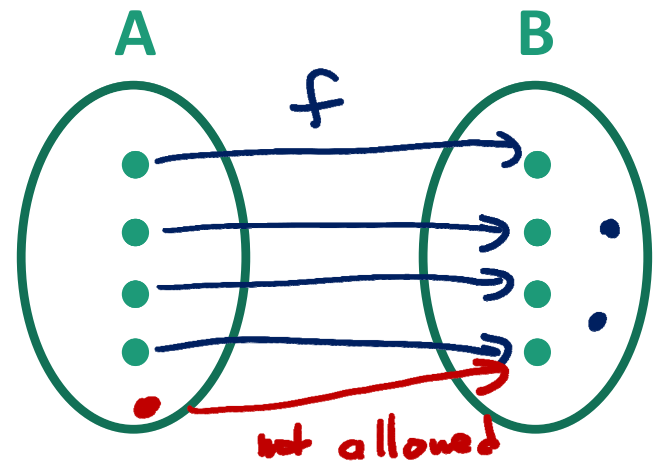
c) The positive powers of 2: $POW2 = \{2, 4, 8, 16, \dots\}$

d) They all have the same size

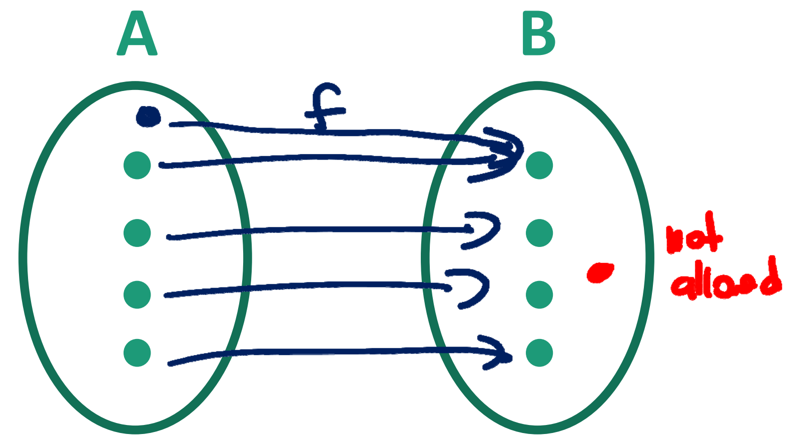
Set Theory Review

A function $f : A \to B$ is

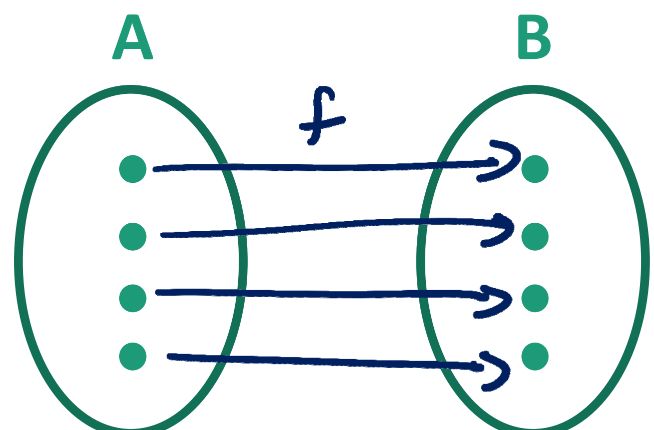
- 1-to-1 (injective) if $f(a) \neq f(a')$ for all $a \neq a'$



- onto (surjective) if for all $b \in B$, there exists $a \in A$ such that $f(a) = b$



- a correspondence (bijective) if it is 1-to-1 and onto, i.e., every $b \in B$ has a unique $a \in A$ with $f(a) = b$



How can we compare sizes of infinite sets?

Definition: Two sets have the same size if there is a bijection between them

A set is countable if either

- it is a finite set, or
- it has the same size as $\mathbb{N}$, the set of natural numbers

Set is "countably infinite"

Examples of countable sets

- $\emptyset$
- $\{0,1\}$
- $\{0, 1, 2, \ldots, 8675309\}$

finite sets

- $E = \{2, 4, 6, 8, \ldots\}$
- $SQUARES = \{1, 4, 9, 16, 25, \ldots\}$
- $POW2 = \{2, 4, 8, 16, 32, \ldots\}$

countably infinite

$f: \mathbb{N} \to E$ defined by $f(n) = 2n$ is a bijection

$f: \mathbb{N} \to SQUARES$ defined by $f(n) = n^2$ is a bijection

$$|E| = |SQUARES| = |POW2| = |\mathbb{N}|$$

How to show that $\mathbb{N} \times \mathbb{N}$ is countable?

$= \{(x,y) \mid x \in \mathbb{N},\ y \in \mathbb{N}\}$

$f(1)=$ $f(2)=$ $f(4)=$

$f(3)=$ $f(5)=$

$f(6)=$

$(1,1)$	$(2,1)$	$(3,1)$	$(4,1)$	$(5,1)$	…
$(1,2)$	$(2,2)$	$(3,2)$	$(4,2)$	$(5,2)$	…
$(1,3)$	$(2,3)$	$(3,3)$	$(4,3)$	$(5,3)$	…
$(1,4)$	$(2,4)$	$(3,4)$	$(4,4)$	$(5,4)$	…
$(1,5)$	$(2,5)$	$(3,5)$	$(4,5)$	$(5,5)$	
⋮					⋱

$f: \mathbb{N} \to (\mathbb{N} \times \mathbb{N})$

How to argue that a set S is countable

- Describe how to "list" the elements of S, usually in stages:

Ex: Stage 1) List all pairs (x, y) such that $x + y = 2$ $f(1) = (1,1)$

Stage 2) List all pairs (x, y) such that $x + y = 3$ $f(2) = (2,1)$ $f(3) = (1,2)$

...

Stage n) List all pairs (x, y) such that $x + y = n + 1$

$(n,1)$ $(n-1,1)$ .... $(1,n)$

...

- Explain why every element of S appears in the list

Ex: Any $(x, y) \in \mathbb{N} \times \mathbb{N}$ will be listed in stage $x + y - 1$

- Define the bijection $f: \mathbb{N} \to S$ by $f(n) =$ the n'th element in this list (ignoring duplicates if needed)

automatically makes f 1-1

More examples of countable sets

Proof 1: $f : \mathbb{N} \to \{0,1\}^*$ $f(n) =$ binary representation of n almost works.... but have to be careful to ensure 01, 001, ... are hit by f

- $\{0,1\}^*$

  Proof 2: $\{\underbrace{\varepsilon}_{0}, \underbrace{0, 1}_{1}, \underbrace{00, 01, 10, 11}_{2}, \ldots\}$

  Could enumerate in stages

  In stage i (for $i = 0, 1, \ldots$), enumerate all strings of length i
- $\{\langle M \rangle \mid M$ is a Turing machine$\}$
  - Can encode TMs as binary strings, i.e. in $\{0,1\}^*$
  - $\{\langle M \rangle \mid M$ is a TM$\} \subseteq \{0,1\}^*$
- $\mathbb{Q} =$ {rational numbers}

  Same as proof that $\mathbb{N} \times \mathbb{N}$ countable, interpreting fractions $\frac{x}{y}$ as (x, y)
- If $A \subseteq B$ and B is countable, then A is countable
- If A and B are countable, then $A \times B$ is countable
- Nonempty S is countable (finite or countably infinite) if and only if there exists a surjection (an onto function) $f : \mathbb{N} \to S$