

BU CS 332 – Theory of Computation

<https://forms.gle/xJ1wybwReq4x4dEW6>



Lecture 12:

- More decidable languages
- Universal Turing Machine
- Countability

Reading:

Sipser Ch 4.1, 4.2



Alexander Poremba & Mark Bun

March 17, 2026

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)

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

Today: More decidable languages

Are there undecidable languages? How can we prove so?

A “Universal” Algorithm for Recognizing Regular Languages

$\langle \cdot \rangle$ denotes “encoding”
i.e. string representations

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

Computational Problem:
Given DFA D , string w
does D accept input 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

All 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 .

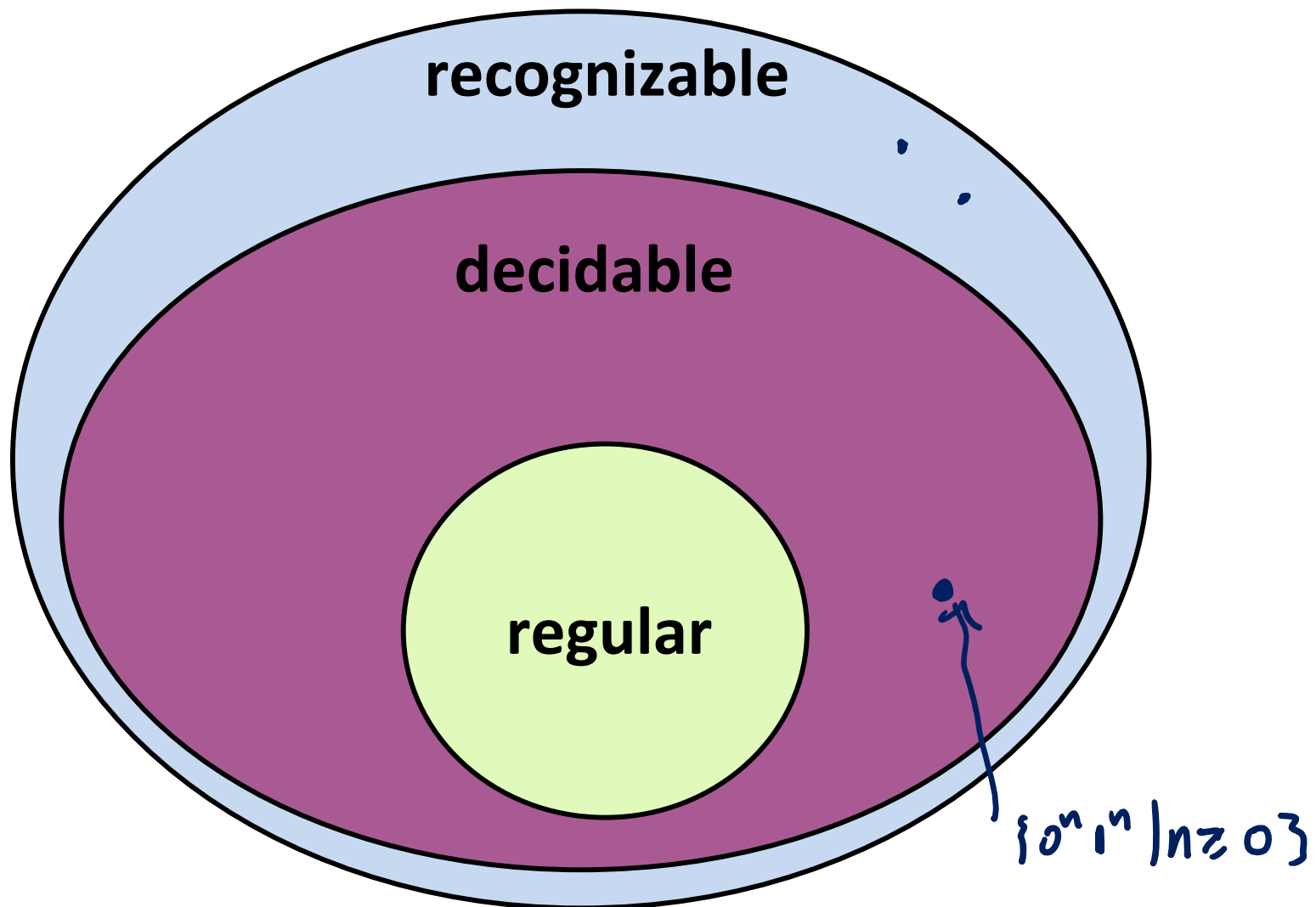
On input w :

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

Proof that M_D decides L :

- If $w \in L$: D accepts $w \Rightarrow \langle D, w \rangle \in A_{\text{DFA}} \Rightarrow$ Decider for A_{DFA} accepts $\Rightarrow M_D$ accepts
- If $w \notin L$: D rejects $w \Rightarrow \langle D, w \rangle \notin A_{\text{DFA}} \Rightarrow$ Decider for A_{DFA} rejects $\Rightarrow M_D$ 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 a 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 k states:

1. Run 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

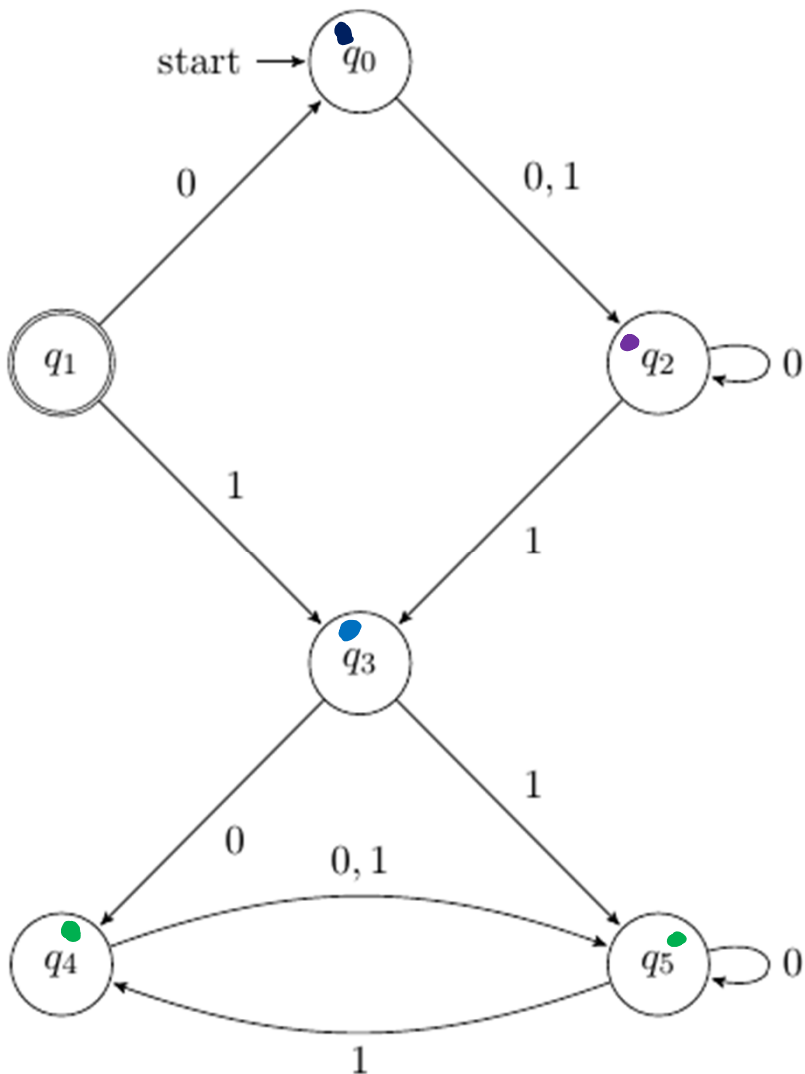
Proof of correctness:

- If $\langle D \rangle \in E_{\text{DFA}} \Rightarrow L(D) = \emptyset \Rightarrow$ impossible to reach an accept state in D from its start state
 $\Rightarrow k$ steps of BFS fails to reach an accept state of D
 \Rightarrow TM accepts overall
- If $\langle D \rangle \notin E_{\text{DFA}} \Rightarrow L(D) \neq \emptyset \Rightarrow \exists$ a path from start to an accept state in D
 $\Rightarrow \exists$ a path from start to accept w/ length $\leq k$
 \Rightarrow BFS finds an accept state within k steps
 \Rightarrow TM rejects

E_{DFA} Example

$L(0) \in E_{DFA}$

$D =$



BFS

1) q_0

2) q_2

3) q_3

4) q_4, q_5

5) No new states to explore

No accept states reachable by BFS

\Rightarrow TM accepts

New Deciders from Old: Equality Testing

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

Theorem: EQ_{DFA} is decidable Computational Problem:
Given DFAs D_1, D_2 , do they recognize the same language?

Proof: The following TM decides EQ_{DFA}

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

1. Construct DFA D recognizing the **symmetric difference**

$$L(D_1) \Delta L(D_2) = \{ w \mid w \in L(D_1) \text{ or } w \in L(D_2) \text{ but not both.} \}$$

2. Run the decider for E_{DFA} on $\langle D \rangle$ and return its output

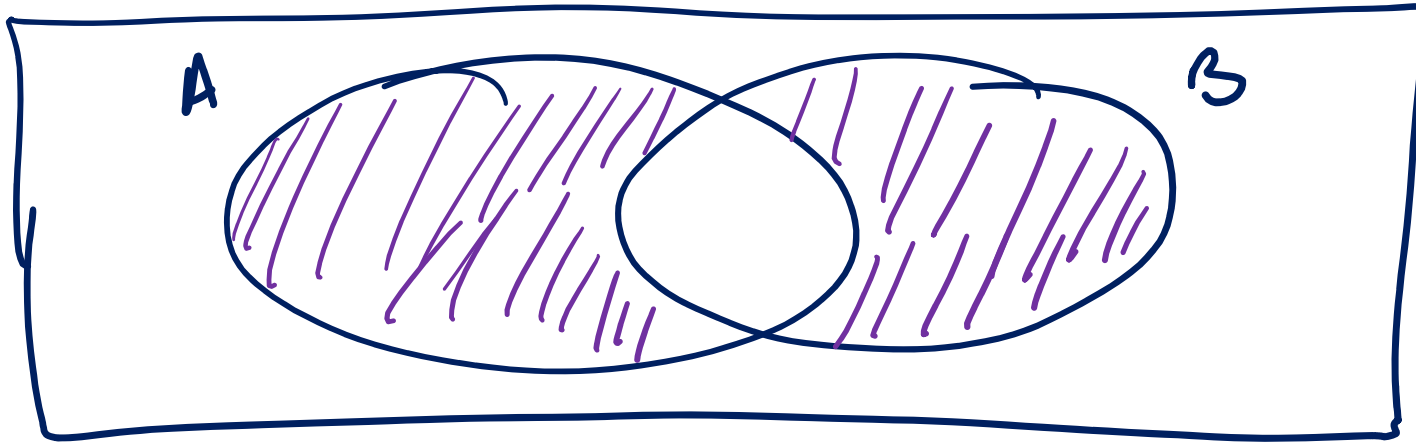
Correctness:

• If $\langle D_1, D_2 \rangle \in EQ_{DFA}$: $L(D_1) = L(D_2) \Rightarrow L(D_1) \Delta L(D_2) = \emptyset$
 $\Rightarrow L(D) = \emptyset$
 \Rightarrow Decider for E_{DFA} accepts $\Rightarrow M$ accepts

• If $\langle D_1, D_2 \rangle \notin EQ_{DFA}$: $L(D_1) \neq L(D_2) \Rightarrow L(D_1) \Delta L(D_2) \neq \emptyset$
 $\Rightarrow L(D) \neq \emptyset$
 \Rightarrow Decider for E_{DFA} rejects $\Rightarrow M$ rejects

Symmetric Difference

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



$$A \Delta B = (A \setminus B) \cup (B \setminus A) = (A \cap \bar{B}) \cup (B \cap \bar{A})$$

If A and B are recognized by DFAs, can apply closure

constructions for union, intersection, complement to give a DFA

recognizing $A \Delta B$

→ can be implemented on a TM.

Universal Turing Machine

Meta-Computational Languages

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

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

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

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

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

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

The Universal Turing Machine

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



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

Theorem: A_{TM} is Turing-recognizable

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**.

Correctness:

$\langle M, w \rangle \in A_{\text{TM}} : \text{TM } M \text{ accepts } w \Rightarrow U \text{ accepts}$

$\langle M, w \rangle \notin A_{\text{TM}} : \text{TM } M \text{ does not accept } w \Rightarrow U \text{ does not accept}$

Universal TM and A_{TM}



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

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

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**.

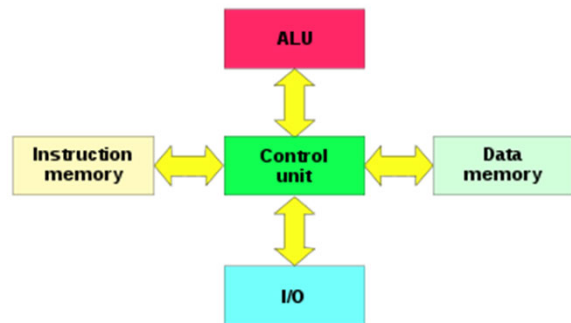
- a) It may reject inputs $\langle M, w \rangle$ where M accepts w
- b) It may accept inputs $\langle M, w \rangle$ where M rejects w
- c) It may loop on inputs $\langle M, w \rangle$ where M loops on w
- d) It may loop on inputs $\langle M, w \rangle$ where M accepts w

More on the Universal TM

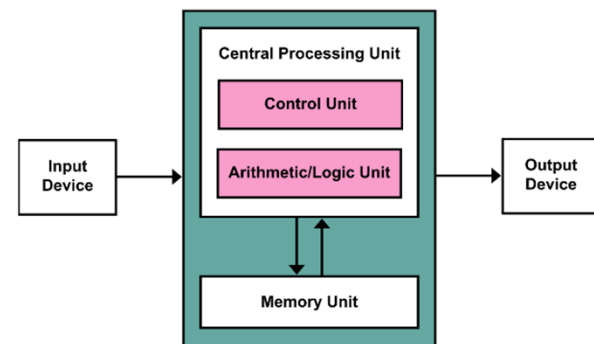
"It is possible to invent a single machine which can be used to compute any computable sequence. If this machine **U** is supplied with a tape on the beginning of which is written the S.D ["standard description"] of some computing machine **M**, then **U** will compute the same sequence as **M**."

- 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

...but it turns out A_{TM} (and E_{TM}, EQ_{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: $\mathbb{N} = \{1, 2, 3, \dots\}$

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

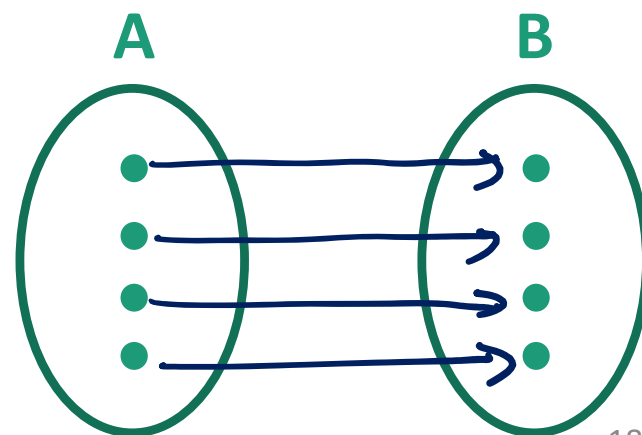
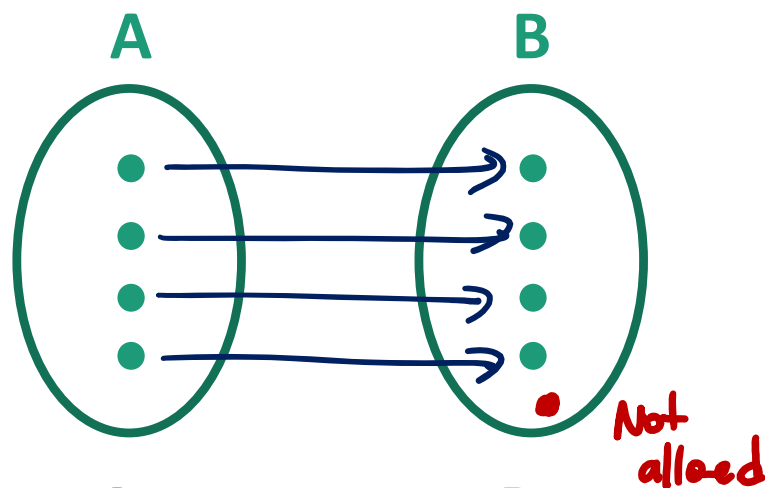
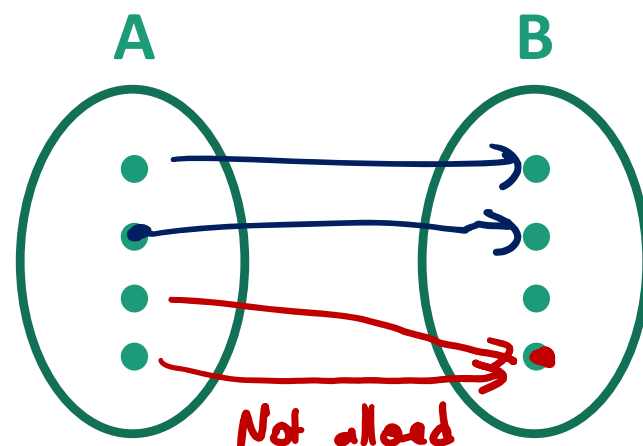
c) The positive powers of 2: $POW2 = \{2, 4, 8, 16, \dots\}$

d) They all have the same size

Set Theory Review

A function $f: A \rightarrow B$ 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

"countably infinite"

Examples of Countable Sets

- \emptyset
 - $\{0,1\}$
 - $\{0, 1, 2, \dots, 8675309\}$
- Finite \Rightarrow countable

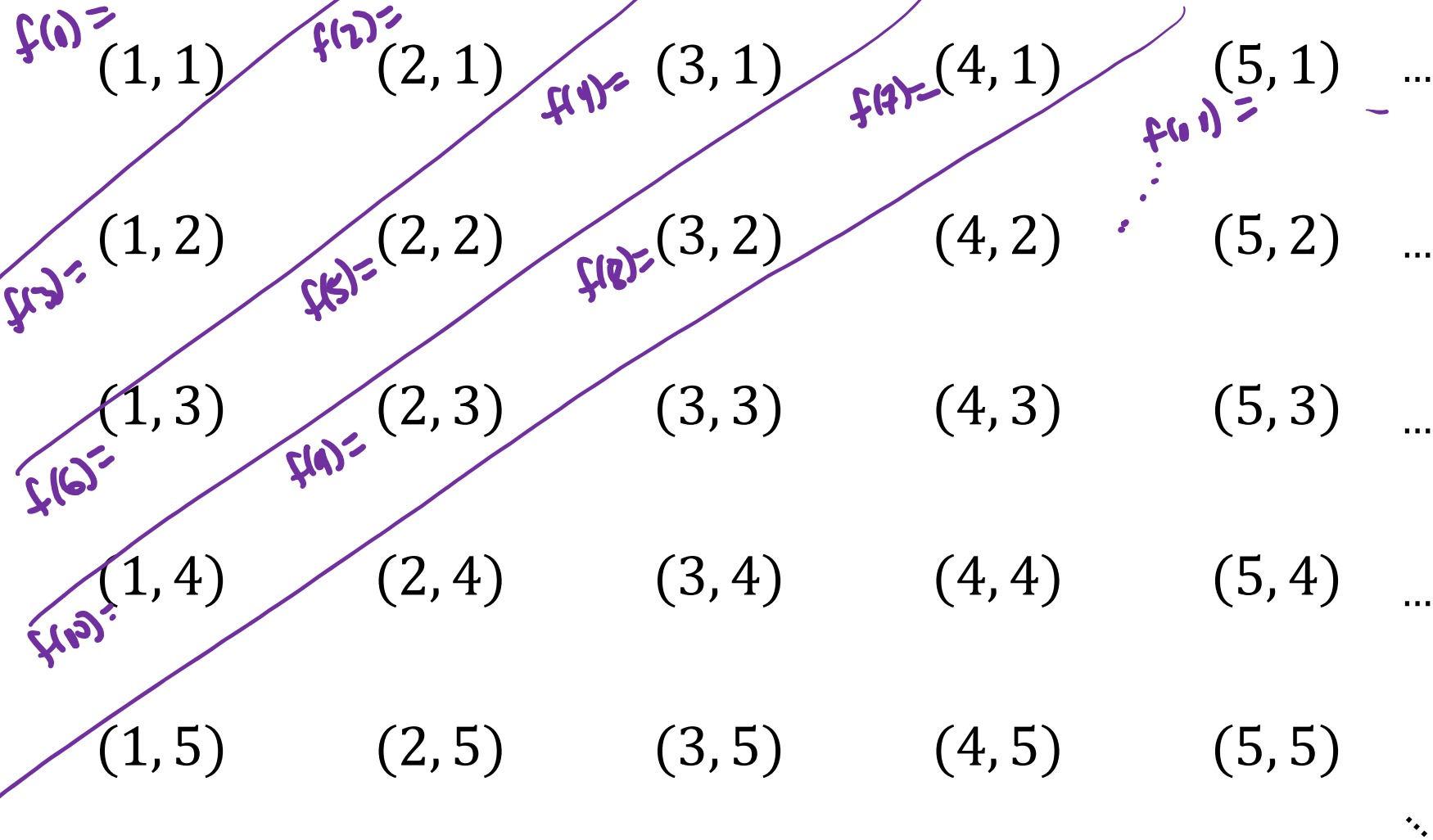
- $E = \{2, 4, 6, 8, \dots\}$ $f: \mathbb{N} \rightarrow E$ $f(n) = 2n$ is a bijection
 - $SQUARES = \{1, 4, 9, 16, 25, \dots\}$ $f(n) = n^2$
 - $POW2 = \{2, 4, 8, 16, 32, \dots\}$ $f(n) = 2^n$
- Countably infinite

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

Why is $\mathbb{N} \times \mathbb{N}$ Countable?

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

Goal: Construct bijection $f: \mathbb{N} \rightarrow \mathbb{N} \times \mathbb{N}$



How to Argue That a Set S is Countable

- Describe how to “list” the elements of S , usually in stages: *finite*

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

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

...

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

...
(n, 1) *(n-1, 2)* ... *(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 $f: \mathbb{N} \rightarrow S$ by $f(n) =$ the n 'th element in this list (ignoring duplicates if needed)

*automatically ensures
 $f \ni$ injective*

