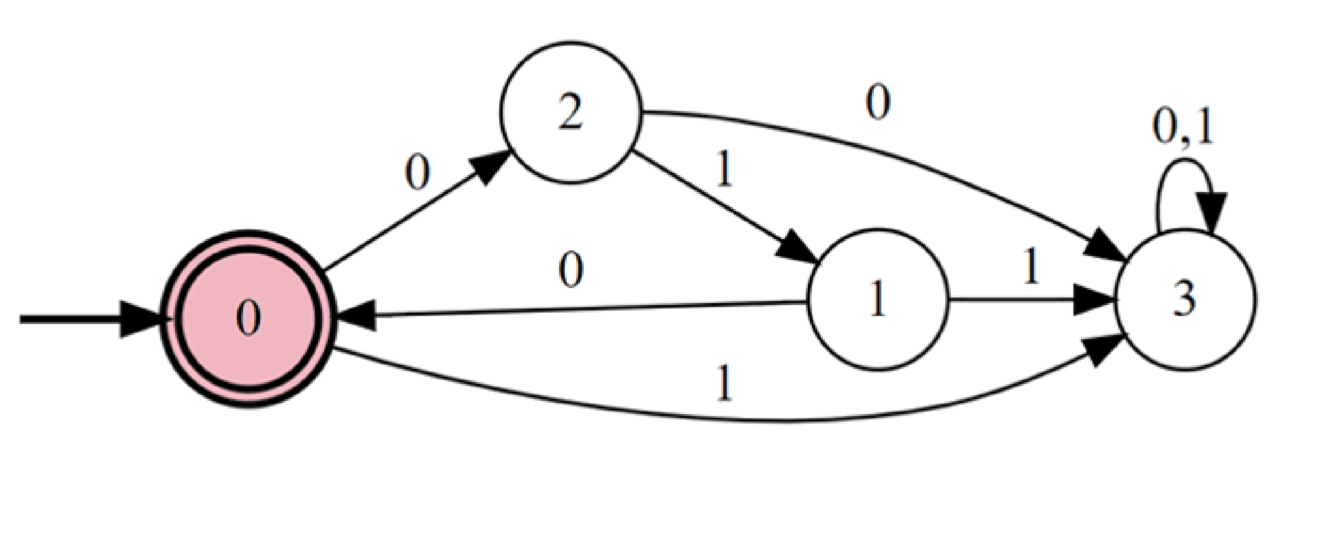
CS3120: Theory of Computation

University of Virginia, Spring 2026

HW 1 due this Friday, Jan 30 (10:00pm)

Quiz 2 coming soon



Class 4: *Finite Automata*

University of Virginia
CS3120: DMT2
https://weikailin.github.io/cs3120-toc
Wei-Kai Lin

Plan

Cardinality of Sets

Cantor’s Theorem

Functions, Problems, Languages

Computation Models

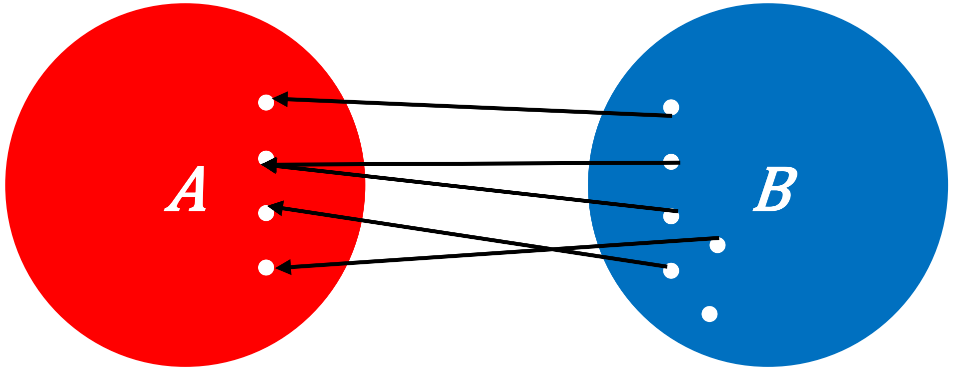
(Deterministic) Finite Automata

Today: Chapter 2.4 in the TCS book

https://introtcs.org/public/lec_02_representation.html#cantorsec

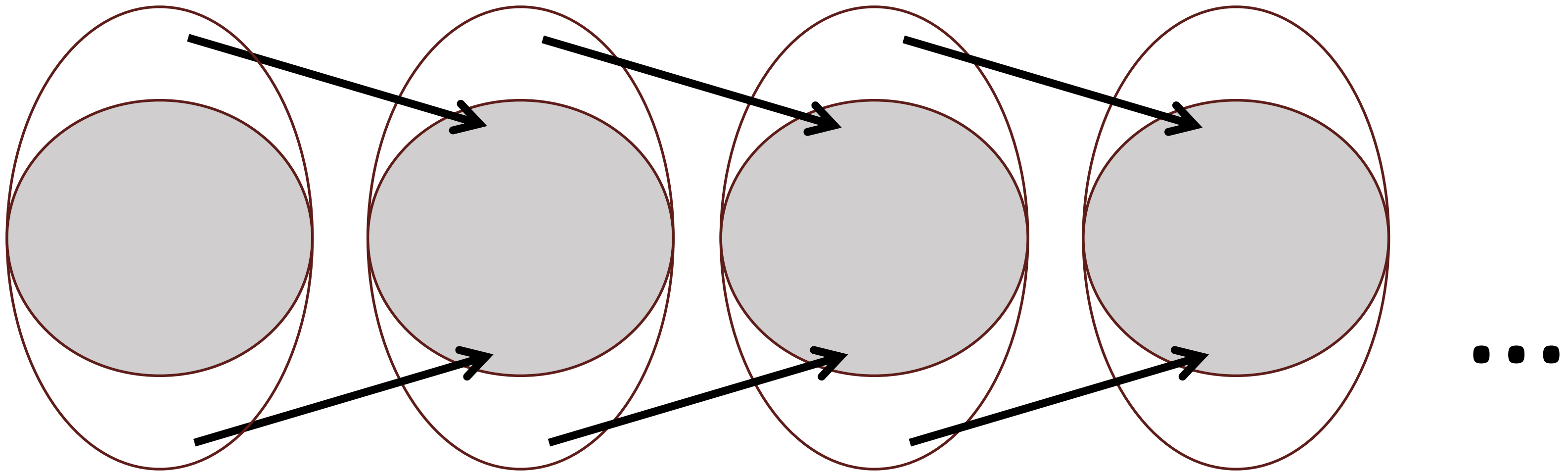
Recap: Comparing Cardinality of (Infinite) Sets

Definition. If there exists a **surjective function** from sets B to A, then we say the cardinality of B is ***greater than or equal to*** the cardinality of A.
We denote this as $|A| \leq |B|$.



Recap: Infinite (Cardinality) Sets

Definition. A set is *Dedekind-infinite* if and only if it has the same cardinality as some **strict subset** of itself.



Recap: Countably Infinite Sets

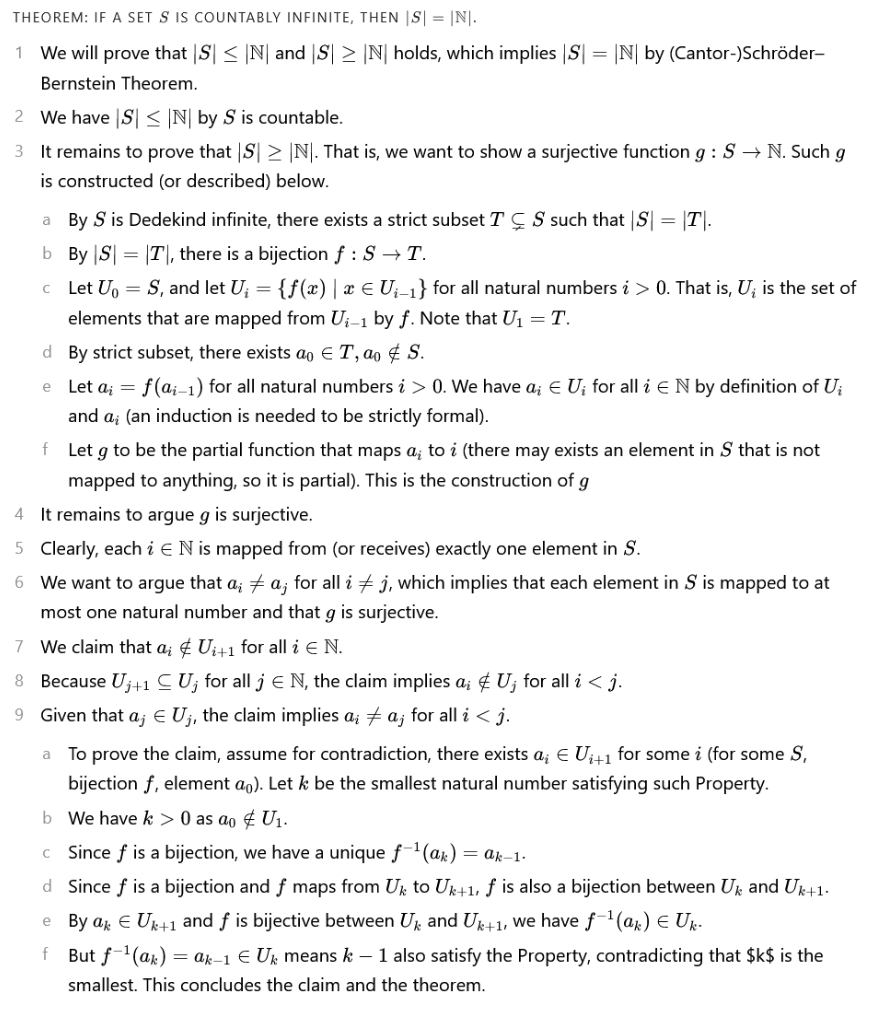
Definition: A set S is *countable* if and only if there exists a *surjective function* from $\mathbb{N}$ to S.

Theorem: A set S is *countably infinite* if and only if there exists a bijection between S and $\mathbb{N}$.

Proof.
"⇐" is easier: A bijection $f: S \rightarrow \mathbb{N}$ is a surjection $\mathbb{N} \rightarrow S$, which implies S is countable.
Also, the function composition of two bijections is still bijection.
We have a bijection $f: S \rightarrow \mathbb{N}$, a bijection $\mathbb{N}$ to EVEN, and EVEN to $T = f^{-1}(EVEN)$,
but T is a strict subset of S.
"⇒" is involved (next slide)

Theorem: A set S is *countably infinite* if and only if there exists a bijection between S and $\mathbb{N}$.

https://weikailin.github.io/cs3120-toc/#theorem-if-a-set-s-is-countably-infinite-then-cards--cardn



THEOREM: IF A SET S IS COUNTABLY INFINITE, THEN $|S| = |\mathbb{N}|$.

1 We will prove that $|S| \le |\mathbb{N}|$ and $|S| \ge |\mathbb{N}|$ holds, which implies $|S| = |\mathbb{N}|$ by (Cantor-)Schröder–Bernstein Theorem.

2 We have $|S| \le |\mathbb{N}|$ by S is countable.

3 It remains to prove that $|S| \ge |\mathbb{N}|$. That is, we want to show a surjective function $g : S \to \mathbb{N}$. Such g is constructed (or described) below.

- a By S is Dedekind infinite, there exists a strict subset $T \subsetneq S$ such that $|S| = |T|$.
- b By $|S| = |T|$, there is a bijection $f : S \to T$.
- c Let $U_0 = S$, and let $U_i = \{f(x) \mid x \in U_{i-1}\}$ for all natural numbers $i > 0$. That is, U_i is the set of elements that are mapped from U_{i-1} by f. Note that $U_1 = T$.
- d By strict subset, there exists $a_0 \in T, a_0 \notin S$.
- e Let $a_i = f(a_{i-1})$ for all natural numbers $i > 0$. We have $a_i \in U_i$ for all $i \in \mathbb{N}$ by definition of U_i and a_i (an induction is needed to be strictly formal).
- f Let g to be the partial function that maps a_i to i (there may exists an element in S that is not mapped to anything, so it is partial). This is the construction of g

4 It remains to argue g is surjective.

5 Clearly, each $i \in \mathbb{N}$ is mapped from (or receives) exactly one element in S.

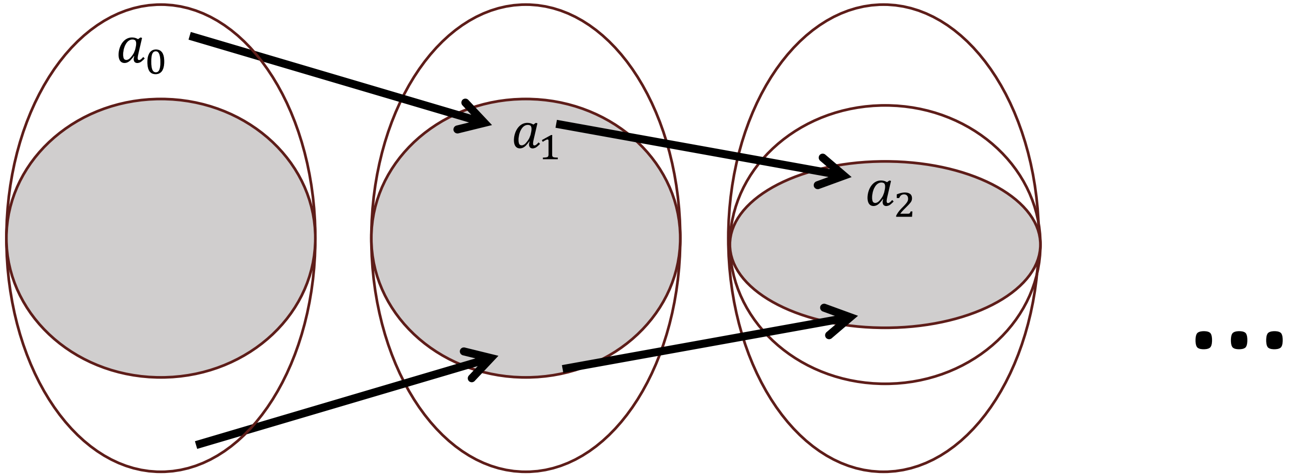
6 We want to argue that $a_i \ne a_j$ for all $i \ne j$, which implies that each element in S is mapped to at most one natural number and that g is surjective.

7 We claim that $a_i \notin U_{i+1}$ for all $i \in \mathbb{N}$.

8 Because $U_{j+1} \subseteq U_j$ for all $j \in \mathbb{N}$, the claim implies $a_i \notin U_j$ for all $i < j$.

9 Given that $a_j \in U_j$, the claim implies $a_i \ne a_j$ for all $i < j$.

- a To prove the claim, assume for contradiction, there exists $a_i \in U_{i+1}$ for some i (for some S, bijection f, element a_0). Let k be the smallest natural number satisfying such Property.
- b We have $k > 0$ as $a_0 \notin U_1$.
- c Since f is a bijection, we have a unique $f^{-1}(a_k) = a_{k-1}$.
- d Since f is a bijection and f maps from U_k to U_{k+1}, f is also a bijection between U_k and U_{k+1}.
- e By $a_k \in U_{k+1}$ and f is bijective between U_k and U_{k+1}, we have $f^{-1}(a_k) \in U_k$.
- f But $f^{-1}(a_k) = a_{k-1} \in U_k$ means $k - 1$ also satisfy the Property, contradicting that k is the smallest. This concludes the claim and the theorem.



Idea:

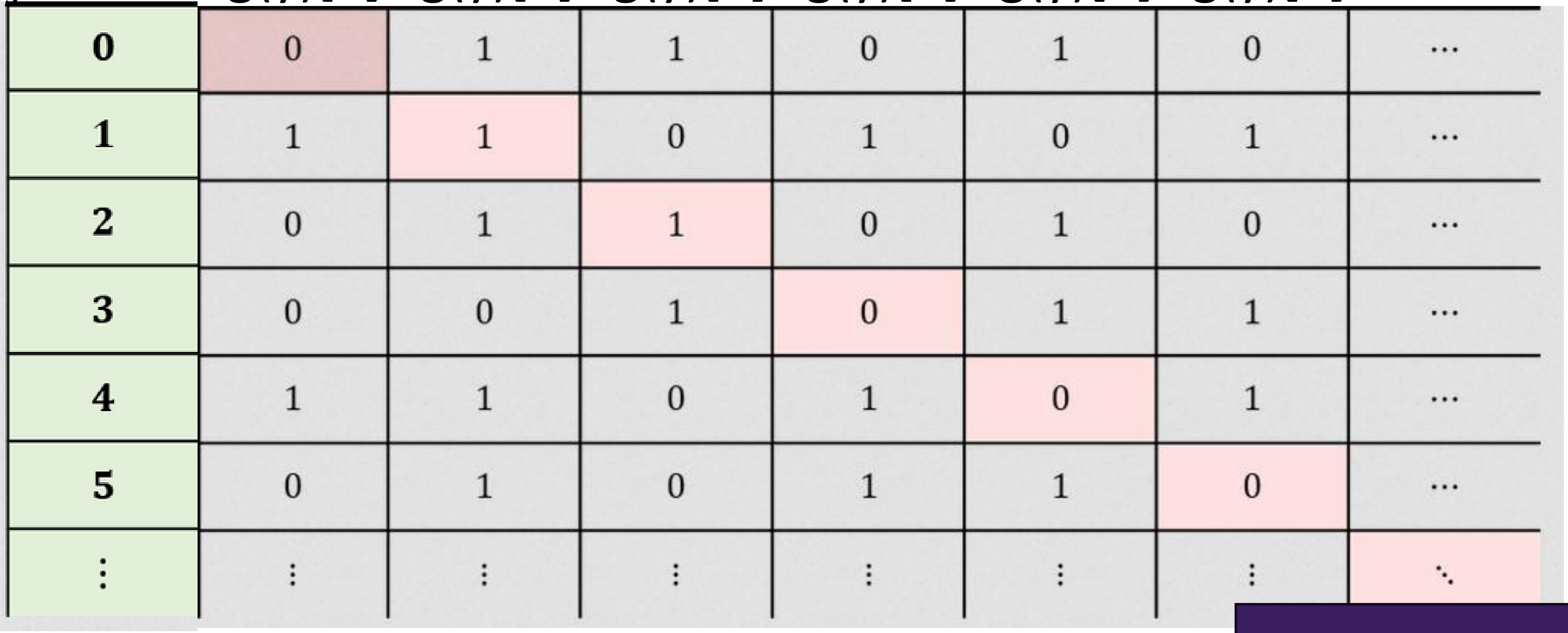
$$a_1 \ne a_0$$
$$a_2 \ne a_1, a_0$$
$$a_3 \ne a_2, a_1, a_0$$

…

$\mathbf{\{0,1\}^{\infty}}$

Recap: *Infinite* Binary Strings is uncountable

Assume $g\colon \mathbb{N} \to \{0,1\}^{\infty}$ a bijection.



$y \in \mathbb{N}$	g(y)[0]	g(y)[1]	g(y)[2]	g(y)[3]	g(y)[4]	g(y)[5]	
0	0	1	1	0	1	0	⋯
1	1	1	0	1	0	1	⋯
2	0	1	1	0	1	0	⋯
3	0	0	1	0	1	1	⋯
4	1	1	0	1	0	1	⋯
5	0	1	0	1	1	0	⋯
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱

Idea: Negate the diagonal

Let s := (¬g(0)[0], ¬g(1)[1], ¬g(2)[2], ¬g(3)[3], …).

$s \in \{0,1\}^{\infty}$ by definition of $\{0,1\}^{\infty}$.

Let $y \in \mathbb{N}$ s.t. $g(y) = s$. But $g(y)[y] = s[y] = \neg g(y)[y]$, a contradiction.

Georg Cantor
(1845-1918)

Georg Cantor's Shocking Result

(~1874)

For **all** sets S, $|pow(S)| > | S |$.

Note: this isn't what the TCS book calls *Cantor's Theorem* but is what most people call "Cantor's Theorem". Cantor came up with the diagonalization argument we will see Tuesday. The proof we'll see soon of Cantor's Theorem is believed to have been first done by Hessenberg (1906).

Cantor’s Theorem:
For all sets S, $|pow(S)| > |S|$.

Proof. Assume for contradiction, exists A s.t. $|pow(A)| \leq |A|$.
Let $g: A \to pow(A)$ be surjective.

$y \in A$	$a_0 \in g(a_0)$	$a_1 \in g(a_0)$	$a_2 \in g(a_0)$	$a_3 \in g(a_0)$	…
a_0	0	1	1	0	…
a_1	1	1	0	1	…
a_2	0	1	1	0	…
a_3	0	0	1	0	…
a_4	0	1	1	0	…
…	…	…	…	…	…

This table is only for intuition as
A may not be ordered nor countable

Let $T = \{y \in A \mid y \notin g(y)\}$, ie, negate of the diagonal. $T \subseteq pow(A)$.
Let $u \in A$ s.t. $g(u) = T$. But $u \in g(u)$ iff $u \in T$ iff $u \notin g(u)$, a contradiction.

Proof. For all sets S, $|pow(S)| > | S |$.

Towards a contradiction, **assume** $\exists \boldsymbol{A}.\, |\boldsymbol{pow}(\boldsymbol{A})| \leq |\boldsymbol{A}|$.
By the definition of $\leq$, there must exist a *surjective function* g from $A \rightarrow pow(A)$.

Define $T = \{a \ \in A \mid \ a \notin g(a)\}$ (⋆).

$T \in pow(A)$. (Obviously, its a subset of A.)
Since g is surjective, $\exists\ u \in A$ such that $g(u) = T$.

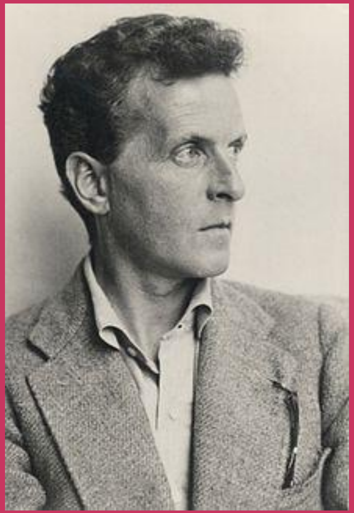
(1) If $u \in g(u)$, then $u \notin T$ by ⋆.
But $T = g(u)$, so $u \notin g(u)$.

(2) If $u \notin g(u)$, then $u \in T$ by ⋆.
But $T = g(u)$, so $u \in g(u)$.

Contradiction! So, there must not exist any A such that $|\,pow(A)| \ \leq |A|$.

Georg Cantor
(1845-1918)

“corruptor of youth”
Leopold Kronecker



“utter nonsense”
Ludwig Wittgenstein

“grave disease”
Henri Poincaré

Georg Cantor
(1845-1918)

My theory stands as firm as a rock; every arrow directed against it will return quickly to its archer. How do I know this? Because I have studied it from all sides for many years; because I have examined all objections which have ever been made against the infinite numbers; and above all, because I have followed its roots, so to speak, to the first infallible cause of all created things.

Georg Cantor, 1887 Letter to K. F. Heman

Any set bigger than $\mathbb{N}$?

Yes:$|pow(\mathbb{N})| = |\{0,1\}^{\infty}| = |[0,1]|$

Any set bigger than $\mathbb{N}$?

Yes: $|pow(\mathbb{N})| = |\{0,1\}^\infty| = |[0,1]|$

- $a \in pow(\mathbb{N})$
- $f_a: \mathbb{N} \to \{0,1\}$ such that $f_a(i) = \begin{cases} 0 \;\; if \;\; i \notin a \\ 1 \;\; if \;\; i \in a \end{cases}$
- $(b_0, b_1, b_2, \dots) \in \{0,1\}^\infty$ such that $b_i = \begin{cases} 0 \;\; if \;\; i \notin a \\ 1 \;\; if \;\; i \in a \end{cases}$
- $0.b_0b_1b_2 \dots \in [0,1]$ in base 2

And also $|pow(\mathbb{N})| = |[0,1]^2| = |\mathbb{R}| = |\mathbb{R}^2| = |\mathbb{C}|$

Any set bigger than $\mathbb{N}$?

Yes:

$$|pow(\mathbb{N})| = |\{0,1\}^{\infty}| = |[0,1]| = |[0,1]^2| = |\mathbb{R}| = |\mathbb{R}^2| = |\mathbb{C}|$$

Any set bigger than [0, 1]?

yes: For **all** sets S, $|pow(S)| > |\ S\ |$

Aleph-Naught

$$\aleph_0 = |\mathbb{N}|$$

“smallest infinite cardinal number”

$$\aleph_1 = ?$$

“*second* smallest infinite cardinal number”

$$\aleph_0 = |\mathbb{N}|$$

“smallest infinite cardinal number”

$$\aleph_1 = ?$$

“*second sm*… …*ite* card… …er”

Is there any set with card… …ween $\mathbb{N}$ and $pow(\mathbb{N})$?

It se… $pow(\mathbb{N})| = |\mathbb{R}| = \aleph_1$

First of Hilbert's 23 problems presented in 1900

…ntor’s Continuum Hypothesis

https://en.wikipedia.org/wiki/Continuum_hypothesis
https://en.wikipedia.org/wiki/Aleph_number#Continuum_hypothesis

To conclude…

Infinities are not Intuitive, at least at first

From the paradise, that
Cantor created for us,
no-one can expel us.
David Hilbert

Why care uncountable?

- How many “problems” are there?
- This course: problem = *function*, eg,

$$f: \{0,1\}^* \to \{0,1\}$$

- Binary string $x \in \{0,1\}^*$ is an instance of the problem f, and $f(x)$ is the answer
- Is the set $\{f: \{0,1\}^* \to \{0,1\}\}$ countable?

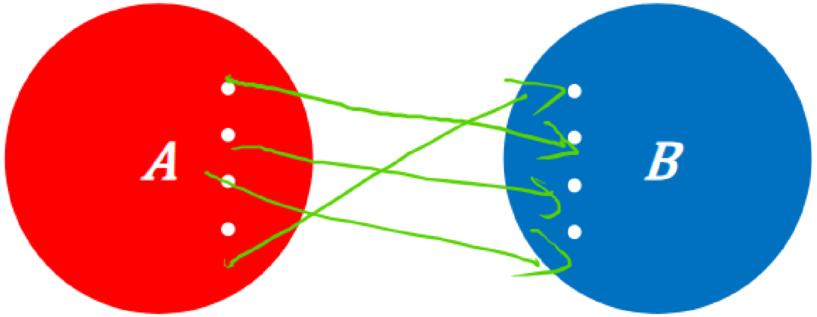
Functions, Problems, Languages

http://commons.wikimedia.org/wiki/File:S70-34902.jpg

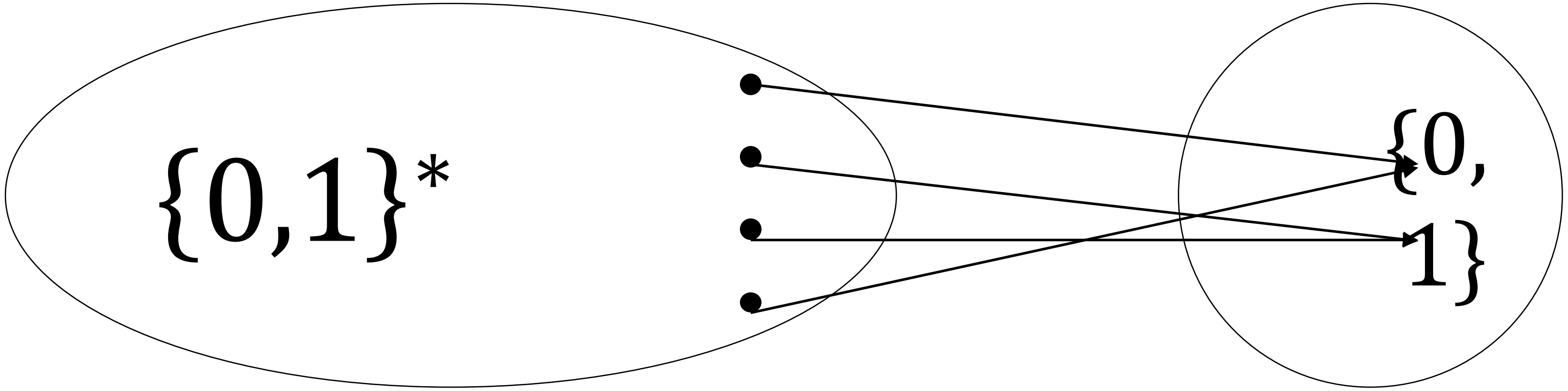
Apollo 13, 1995.
The meaning of “problem” differs in this course.

Boolean Functions

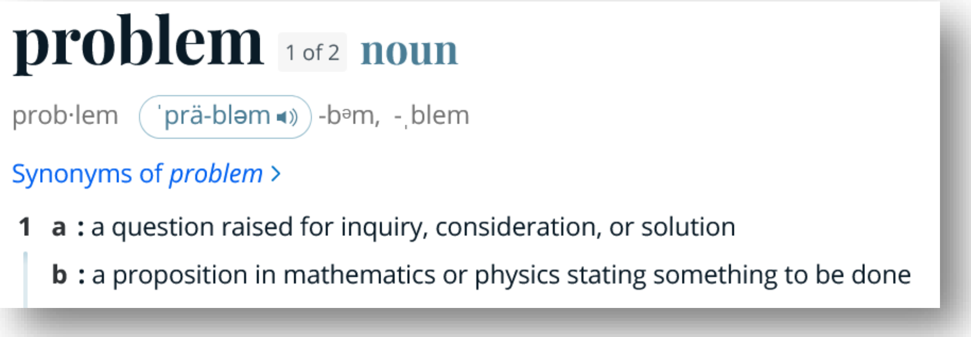
We've talked a lot in this class.



Spoiler: This course, Theory of Computation, mostly considers (total) *Boolean functions*, $f: \{0,1\}^* \to \{0,1\}$



Problems



problem 1 of 2 noun

prob·lem ˈprä-bləm -bᵊm, -ˌblem

Synonyms of *problem* >

1 **a** : a question raised for inquiry, consideration, or solution

b : a proposition in mathematics or physics stating something to be done

A “problem” is often a sentence in math and science.

In Theory of Computation, a problem is: each binary string $x \in \{0,1\}^*$ is answered either ‘Yes’ or ‘No.’

eg, Does $x \in \{0,1\}^*$ represent 3120 as a natural number?

eg, Does $x \in \{0,1\}^*$ represent an English sentence (in Unicode)?

Note: Each x must be Y or N, no neither

Problems and Instances

Definition. A *problem* is a subset $P \subseteq \{0,1\}^*$ of binary strings.

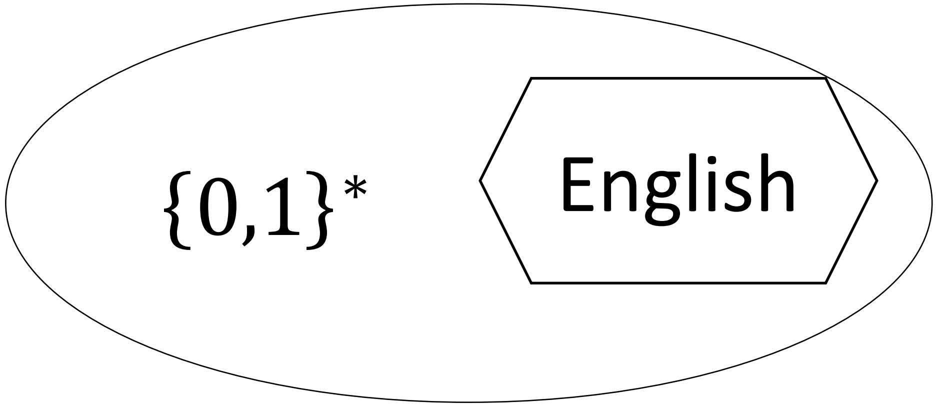
Intuition: Only Yes-No questions are considered.
A problem is a template;
a diff binary string gives a diff Yes/No answer.

Definition. An *instance* (of a problem) is a binary string $x \subseteq \{0,1\}^*$.

Named by linguists such as Noam Chomsky

(Formal) Languages

A language L is a subset of binary strings, $L \subseteq \{0,1\}^*$.



No other requirements!

Eg, no “meaning”

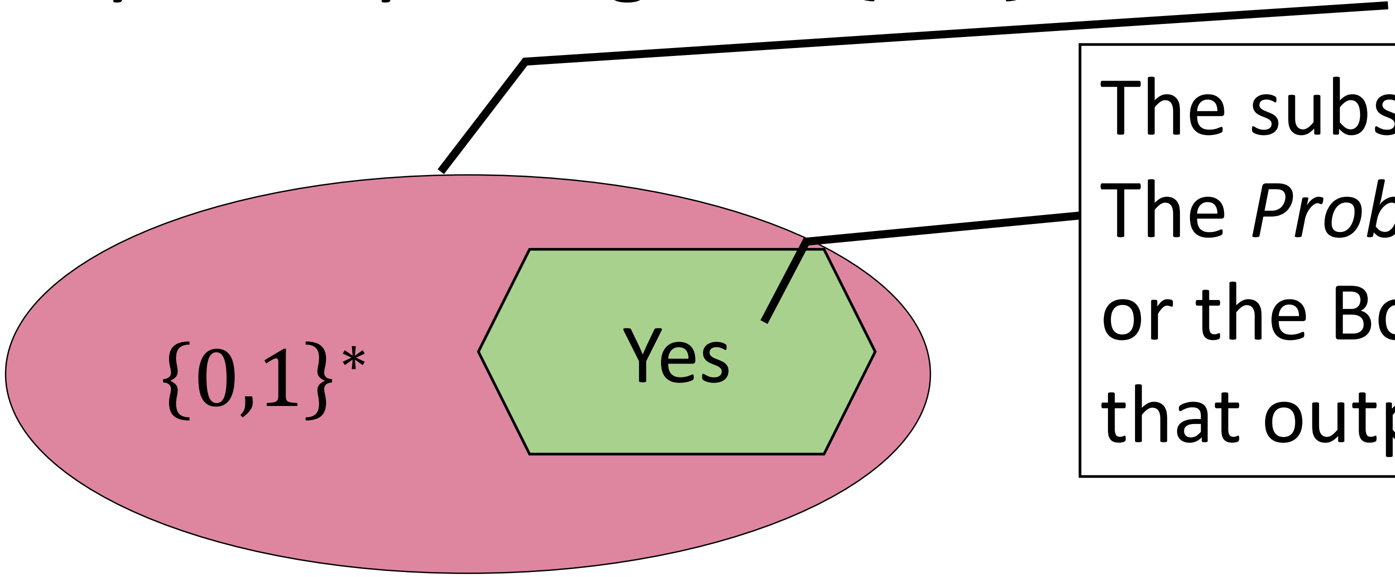
Easy to say “x in language L” or not.

Same as Boolean functions and problems ☺

Convention (ie, Definition)

Problem = Language = Binary Function

Any binary string $x \in \{0,1\}^*$ is an *instance*.



The subset of Yes instances:
The *Problem*, or the *Language,*
or the Boolean function
that outputs 1

Central Question of CS 3120:

Given a Boolean function $f\colon \{0,1\}^* \to \{0,1\}$,
Can we compute $f(x)$
for all $x \in \{0,1\}^*$?

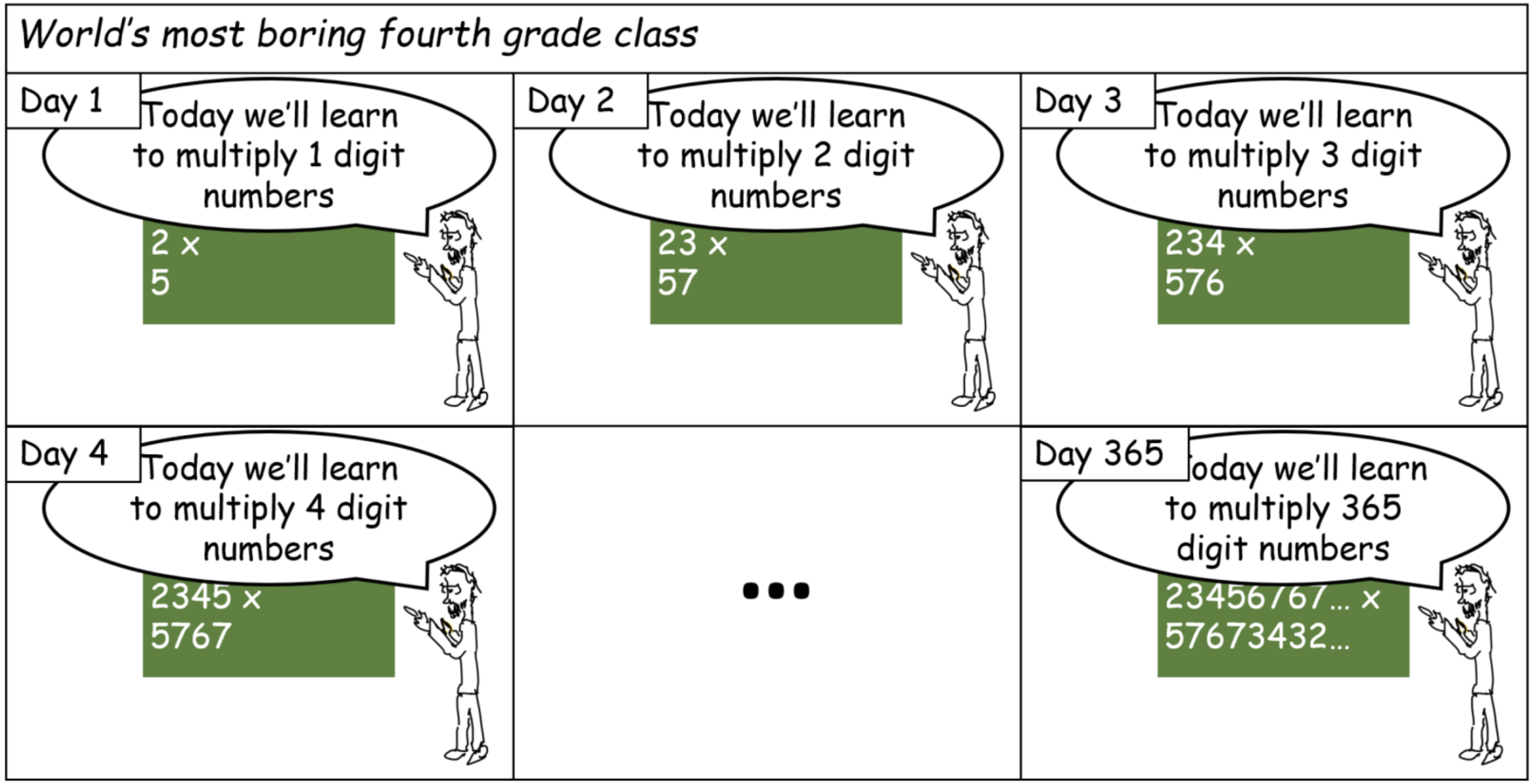
If so, how much resource?

(Deterministic) Finite Automata

A model of computation

We want to compute on Unbounded-Length inputs (instances)

The input length is any $n \in \mathbb{N}$.


World's most boring fourth grade class
Day 1
Today we'll learn to multiply 1 digit numbers
2 ×
5
Day 2
Today we'll learn to multiply 2 digit numbers
23 ×
57
Day 3
Today we'll learn to multiply 3 digit numbers
234 ×
576
Day 4
Today we'll learn to multiply 4 digit numbers
2345 ×
5767
…
Day 365
Today we'll learn to multiply 365 digit numbers
23456767… ×
57673432…

A simple computing model

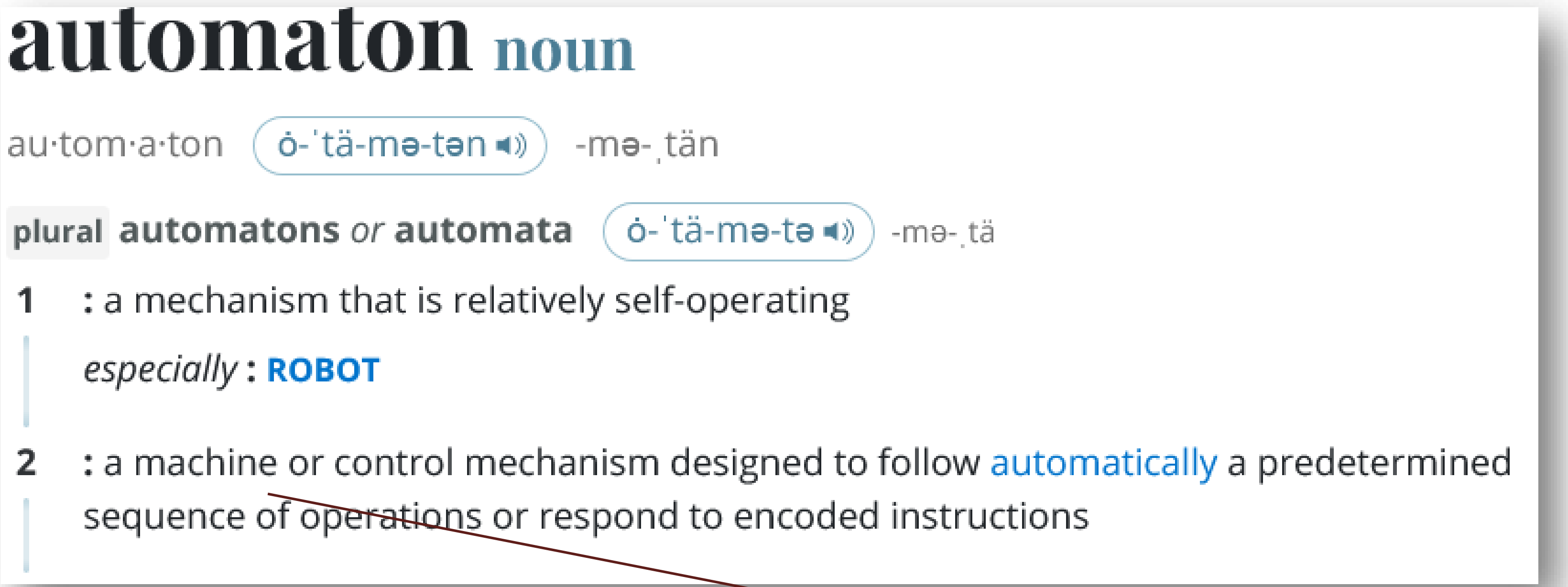
Reading input $x = x_1 \ldots x_n$ as a stream of bits (once)

Compute if $f(x) = 1$ or not at the end

The same “automata” works for all input length n

Convention: Since f is a Boolean function, we use *compute* and *decide* interchangeably.

What’s Automata?



automaton noun

au·tom·a·ton ȯ-ˈtä-mə-tən -mə-ˌtän

plural **automatons** *or* **automata** ȯ-ˈtä-mə-tə -mə-ˌtä

1 **:** a mechanism that is relatively self-operating

especially **: ROBOT**

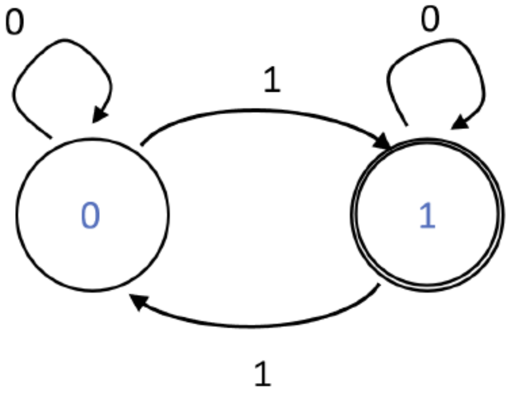
2 **:** a machine or control mechanism designed to follow automatically a predetermined sequence of operations or respond to encoded instructions

Machine

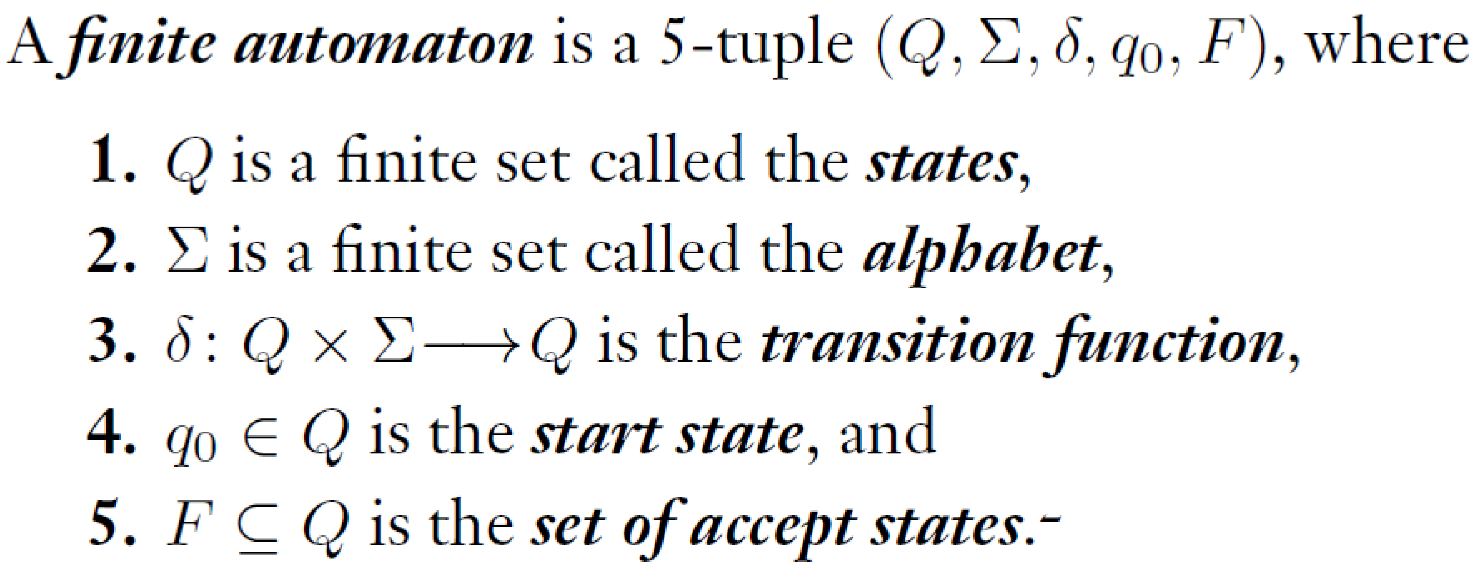
Example: XOR

For $x = x_1 \dots x_n$ let $XOR(x) = x_1 \oplus x_2 \dots \oplus x_n$,
where $\oplus$ is the 2-input exclusive OR

As we read through the bits, we only need one bit of memory b that determines the XOR so far.



Formal Definition of FA

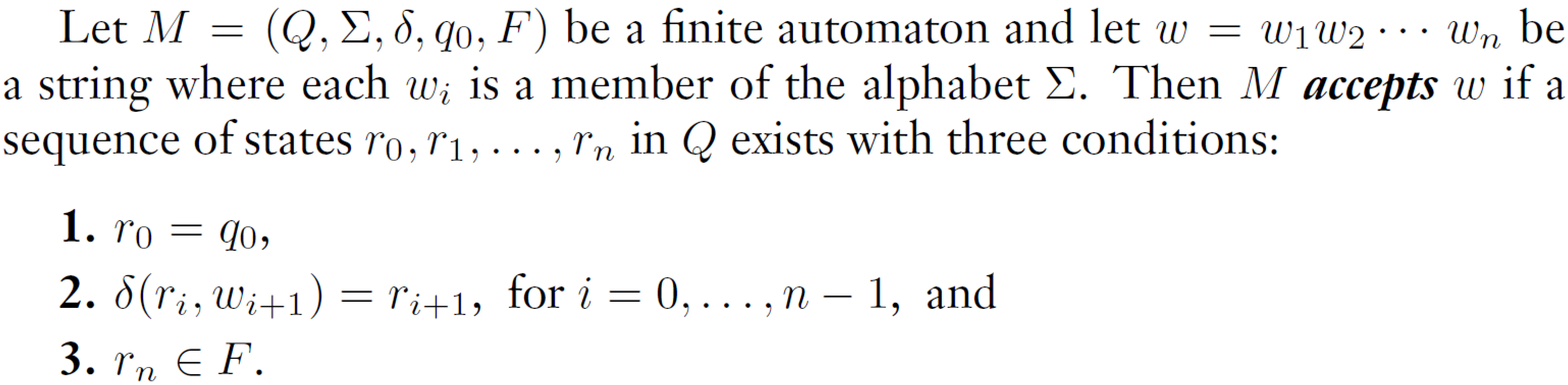


A ***finite automaton*** is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where

1. Q is a finite set called the ***states***,
2. Σ is a finite set called the ***alphabet***,
3. $\delta \colon Q \times \Sigma \longrightarrow Q$ is the ***transition function***,
4. $q_0 \in Q$ is the ***start state***, and
5. $F \subseteq Q$ is the ***set of accept states***.

Introduction to the Theory of Computation. Sipser 2006.

Formal definition of FAs accepting inputs



Let $M = (Q, \Sigma, \delta, q_0, F)$ be a finite automaton and let $w = w_1 w_2 \cdots w_n$ be a string where each w_i is a member of the alphabet Σ. Then M ***accepts*** w if a sequence of states $r_0, r_1, \ldots, r_n$ in Q exists with three conditions:

1. $r_0 = q_0$,

2. $\delta(r_i, w_{i+1}) = r_{i+1}$, for $i = 0, \ldots, n-1$, and

3. $r_n \in F$.

Constant-size memory: only needs r_i
Efficient: read each w_i only once

Example:

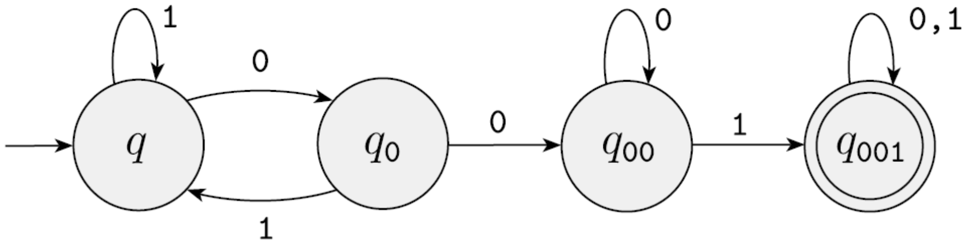


FIGURE 1.22
Accepts strings containing 001

1. $Q =$ □
2. $\Sigma = \{0,1\}$,
3. δ is described as □
4. □ is the start state, and
5. $F =$ □

Example: prefix and suffix

Alphabet $\Sigma = \{0,1\}$. All strings with:

Prefix:

Suffix:

Another example: set of even numbers

Example: Set of multiples of 3 in basis 10:

$$\boldsymbol{\Sigma} = \{\mathbf{0}, \ldots, \mathbf{9}\}$$

Terminology (= definition)

Let M be a FA
Let f be the function: $f(x) = 1$ iff M accepts x
We say that M “computes” or “decides” f

Let L be the language $x \in L$ iff $f(x) = 1$
We might say M “recognizes” (or “decides”) the language L (or function f)

Both f and L ignore the computation’s details, but M cares about it

Complexity class: DFA-Comp

DFA-Comp $= \{f \mid f$ is computed by a DFA $M\}$

More generally “Complexity class” for **set of algorithms $\boldsymbol{X}$**:
= all languages/functions computable with an algorithm in the set X.

DFA-Comp = Complexity class of “FAs”

Why DFAs?

Simple to build (even mechanical)

Easy to use (good HCI)

Low computation time & space

Regular Expressions

A (seemingly) completely different way of dealing with infinite languages (i.e., functions on infinite input sets)

Logistics

- Midterm 0 covers Module 1 (Feb 3, 20 min.)
- Quiz 2 will be posted in 24 hours

Wrap up

Today: Chapter 2.4 in the TCS book

Cardinality of Sets

Countable and Infinite

Power Set

Cantor’s Theorem