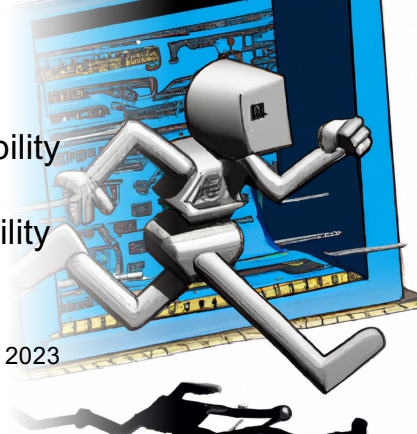


CS21

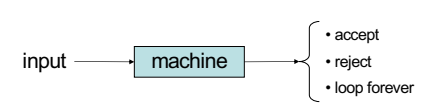
Decidability and Tractability



Lecture 2
January 6, 2023

1

What is computation?



- the set of strings that lead to “accept” is the **language recognized by this machine**
- if every other string leads to “reject”, then this language is **decided** by the machine

January 6, 2023 CS21 Lecture 2 2

2

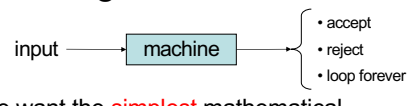
Terminology

- finite **alphabet** Σ : a set of symbols
- language** $L \subseteq \Sigma^*$: subset of strings over Σ
- a **machine** takes an input string and either
 - accepts, rejects, or
 - loops forever
- a machine **recognizes** the set of strings that lead to accept
- a machine **decides** a language L if it accepts $x \in L$ and rejects $x \notin L$

January 6, 2023 CS21 Lecture 2 3

3

What goes inside the box?



- We want the **simplest** mathematical formalization of computation possible.
- Strategy:
 - endow box with a feature of computation
 - try to **characterize** the languages decided
 - identify language we “know” real computers can decide that machine cannot
 - add new feature to overcome limits

January 6, 2023 CS21 Lecture 2 4

4

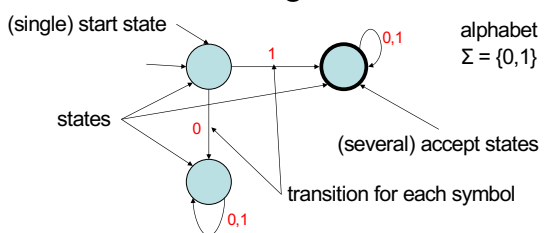
Finite Automata

- simple model of computation
- reads input from left to right, one symbol at a time
- maintains **state**: information about what seen so far (“memory”)
 - **finite** automaton has **finite** # of states: cannot remember more things for longer inputs
- 2 ways to describe: by diagram, or formally

January 6, 2023 CS21 Lecture 2 5

5

FA diagrams



(single) start state

states

(several) accept states

transition for each symbol

alphabet $\Sigma = \{0,1\}$

- read input one symbol at a time; follow arrows; accept if end in accept state

January 6, 2023 CS21 Lecture 2 6

6

FA operation

- Example of FA operation:

input: 0 1 0 1
not accepted

January 6, 2023 CS21 Lecture 2 7

7

FA operation

- Example of FA operation:

input: 1 0 1
accepted

What language does this FA decide?
 $L = \{x : x \in \{0,1\}^*, x_1 = 1\}$

January 6, 2023 CS21 Lecture 2 8

8

Example FA

- What language does this FA decide?
 $L = \{x : x \in \{0,1\}^*, x \text{ has even \# of 1s}\}$
- illustrates fundamental feature/limitation of FA:
 - “tiny” memory
 - in this example only “remembers” 1 bit of info.

January 6, 2023 CS21 Lecture 2 9

9

Example FA

$\Sigma = \{A, B, C\}$
 $\{Q, N, D\}$

Try:
AC
CBCC
AA
BBBBBB
CCBC

“35 cents”

January 6, 2023 CS21 Lecture 2 10

10

FA formal definition

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

- Q is a finite set called the **states**
- Σ is a finite set called the **alphabet**
- $\delta: Q \times \Sigma \rightarrow Q$ is a function called the **transition function**
- q_0 is an element of Q called the **start state**
- F is a subset of Q called the **accept states**

January 6, 2023 CS21 Lecture 2 11

11

FA formal definition

- Specification of this FA in formal terms:
 - $Q = \{\text{even}, \text{odd}\}$ function δ :
 - $\Sigma = \{0, 1\}$ $\delta(\text{even}, 0) = \text{even}$
 - $q_0 = \text{even}$ $\delta(\text{even}, 1) = \text{odd}$
 - $F = \{\text{even}\}$ $\delta(\text{odd}, 0) = \text{odd}$
 - $\delta(\text{odd}, 1) = \text{even}$

January 6, 2023 CS21 Lecture 2 12

12

Formal description of FA operation

finite automaton

$$M = (Q, \Sigma, \delta, q_0, F)$$

accepts a string

$$w = w_1w_2w_3\dots w_n \in \Sigma^*$$

if \exists sequence $r_0, r_1, r_2, \dots, r_n$ of states for which

- $r_0 = q_0$
- $\delta(r_i, w_{i+1}) = r_{i+1}$ for $i = 0, 1, 2, \dots, n-1$
- $r_n \in F$

January 6, 2023 CS21 Lecture 2 13

13

What now?

- We have a **model of computation**
(Maybe this is it. Maybe everything we can do with real computers we can do with FA...)
- try to **characterize** the languages FAs can recognize
 - investigate closure under certain operations
- show that some languages not of this type

January 6, 2023 CS21 Lecture 2 14

14

Characterizing FA languages

- We will show that the set of languages recognized by FA is **closed** under:
 - union " $C = (A \cup B)$ "
 - concatenation " $C = (A \circ B)$ "
 - star " $C = A^*$ "
- Meaning: if A and B are languages recognized by a FA, then C is a language recognized by a FA

January 6, 2023 CS21 Lecture 2 15

15

Characterizing FA languages

- union " $C = (A \cup B)$ "
 $(A \cup B) = \{x : x \in A \text{ or } x \in B \text{ or both}\}$
- concatenation " $C = (A \circ B)$ "
 $(A \circ B) = \{xy : x \in A \text{ and } y \in B\}$
- star " $C = A^*$ " (note: ϵ always in A^*)
 $A^* = \{x_1x_2x_3\dots x_k : k \geq 0 \text{ and each } x_i \in A\}$

January 6, 2023 CS21 Lecture 2 16

16

Concatenation attempt

$$(A \circ B) = \{xy : x \in A \text{ and } y \in B\}$$

What label do we put on the new transitions?

January 6, 2023 CS21 Lecture 2 17

17

Concatenation attempt

- Need it to happen "for free": label with ϵ (?)
- allows construct with multiple transitions with the same label (!?)

January 6, 2023 CS21 Lecture 2 18

18

Nondeterministic FA

- We will make life easier by describing an additional feature (**nondeterminism**) that helps us to “program” FAs
- We will **prove** that FAs with this new feature can be **simulated** by ordinary FA
 - same spirit as programming constructs like procedures
- The concept of **nondeterminism** has a significant role in TCS and this course.

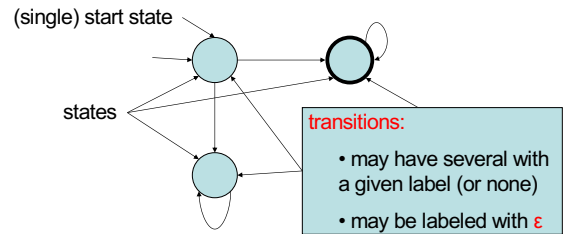
January 6, 2023

CS21 Lecture 2

19

19

NFA diagrams



- At each step, **several** choices for next state – if *possible* to reach accept, then input accepted

January 6, 2023

CS21 Lecture 2

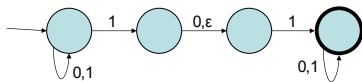
20

20

NFA operation

- Example of NFA operation:

alphabet $\Sigma = \{0,1\}$



input: 0 1 0

not accepted

January 6, 2023

CS21 Lecture 2

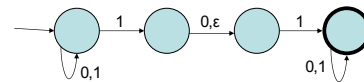
21

21

NFA operation

- Example of NFA operation:

alphabet $\Sigma = \{0,1\}$



input: 1 1 0

accepted

January 6, 2023

CS21 Lecture 2

22

22

NFA operation

- One way to think of NFA operation:
- string $x = x_1x_2x_3\dots x_n$ accepted if and only if
 - there exists a way of inserting ϵ 's into x

$$x_1\epsilon x_2x_3\dots\epsilon x_n$$
 - so that there exists a path of transitions from the start state to an accept state

January 6, 2023

CS21 Lecture 2

23

23

NFA formal definition

A nondeterministic FA

$(Q, \Sigma, \delta$

“powerset of Q”:
the set of all
subsets of Q

- Q is a finite set called the **states**
- Σ is a finite set called the **alphabet**
- $\delta: Q \times (\Sigma \cup \{\epsilon\}) \rightarrow \mathcal{P}(Q)$ is a function called the **transition function**
- q_0 is an element of Q called the **start state**
- F is a subset of Q called the **accept states**

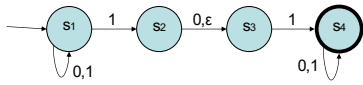
January 6, 2023

CS21 Lecture 2

24

24

NFA formal definition



- Specification of this NFA in formal terms:

- $Q = \{s_1, s_2, s_3, s_4\}$	$\delta(s_1, 0) = \{s_1\}$	$\delta(s_3, 0) = \{s_3\}$
- $\Sigma = \{0, 1\}$	$\delta(s_1, 1) = \{s_1, s_2\}$	$\delta(s_3, 1) = \{s_4\}$
- $q_0 = s_1$	$\delta(s_1, \epsilon) = \{s_1\}$	$\delta(s_3, \epsilon) = \{s_3\}$
- $F = \{s_4\}$	$\delta(s_2, 0) = \{s_3\}$	$\delta(s_4, 0) = \{s_4\}$
	$\delta(s_2, 1) = \{s_3\}$	$\delta(s_4, 1) = \{s_4\}$
	$\delta(s_2, \epsilon) = \{s_3\}$	$\delta(s_4, \epsilon) = \{s_4\}$

January 6, 2023

CS21 Lecture 2

25