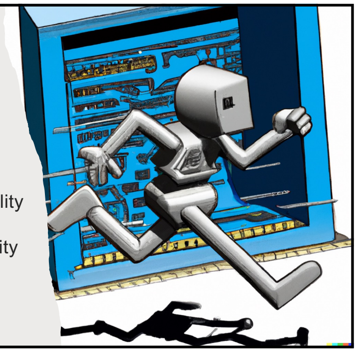
CS21
Decidability
and
Tractability

Lecture 2
January 5, 2024



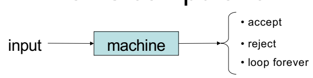
1

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 $x \notin L$

2

What is computation?



- 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

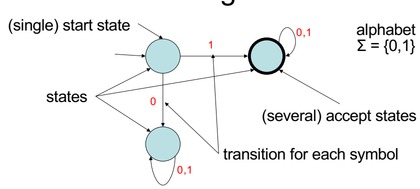
3

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

4

FA diagrams

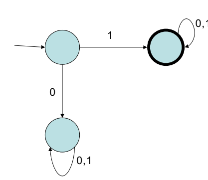


- read input one symbol at a time; follow arrows; accept if end in accept state

5

FA operation

- Example of FA operation:



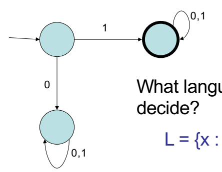
input: 0 1 0 1

not accepted

6

FA operation

- Example of FA operation:



input: 1 0 1

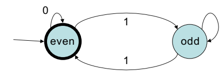
accepted

What language does this FA decide?

L = {x : x ∈ {0,1}*, x_1 = 1}

7

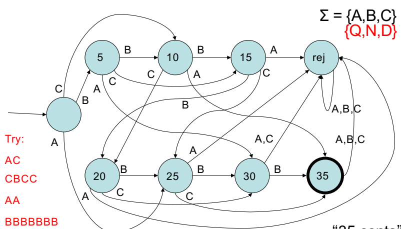
Example FA



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

8

Example FA



Try:
AC
CBCC
AA
BBBBBBB
CCBC

"35 cents"

9

FA formal definition

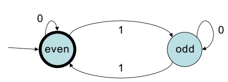
A finite automaton is a 5-tuple

(Q, Σ, δ, q_0, F)

- Q is a finite set called the states
- Σ is a finite set called the alphabet
- δ:Q x Σ → 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

10

FA formal definition



- Specification of this FA in formal terms:
  - Q = {even, odd}
  - Σ = {0,1}
  - q_0 = even
  - F = {even}

function δ:
δ(even, 0) = even
δ(even, 1) = odd
δ(odd, 0) = odd
δ(odd, 1) = even

11

Formal description of FA operation

finite automaton

M = (Q, Σ, δ, q_0, F)

accepts a string

w = $w_1 w_2 w_3 \ldots w_n$ ∈ Σ*

if ∃ sequence $r_0, r_1, r_2, \ldots, r_n$ of states for which

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

12

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

13

Characterizing FA languages

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

14

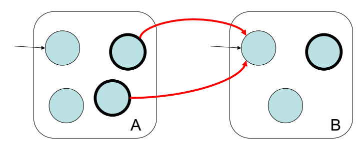
Characterizing FA languages

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

15

Concatenation attempt

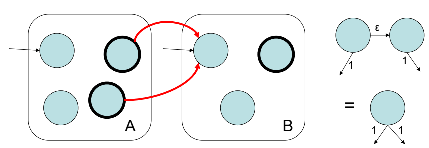
$(A \circ B) = \{xy : x \in A \text{ and } y \in B\}$



What label do we put on the new transitions?

16

Concatenation attempt



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

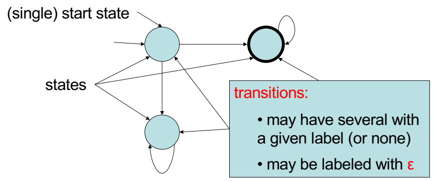
17

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.

18

NFA diagrams

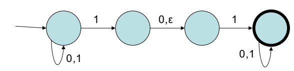


- At each step, several choices for next state
  - if *possible* to reach accept, then input accepted

NFA operation

- Example of NFA operation:

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



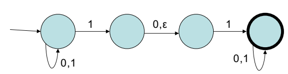
input: 0 1 0

not accepted

NFA operation

- Example of NFA operation:

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



input: 1 1 0

accepted

NFA operation

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

    $x_1\varepsilon\varepsilon x_2\, x_3\ldots\varepsilon x_n$
  - so that there exists a path of transitions from the start state to an accept state

NFA formal definition

A nondeterministic FA

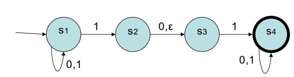
(Q, Σ, δ

transit labeled alpha symbols or ε

"powerset of Q": the set of all subsets of Q

- Q is a finite set called
- Σ is a finite set called the alphabet
- $\delta: Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow \mathcal{P}(Q)$ 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

NFA formal definition



- Specification of this NFA in formal terms:
  - $Q = \{s_1, s_2, s_3, s_4\}$
  - $\Sigma = \{0,1\}$
  - $q_0 = s_1$
  - $F = \{s_4\}$

$\delta(s_1, 0) = \{s_1\}$	$\delta(s_3, 0) = \{\}$
$\delta(s_1, 1) = \{s_1, s_2\}$	$\delta(s_3, 1) = \{s_4\}$
$\delta(s_1, \varepsilon) = \{\}$	$\delta(s_3, \varepsilon) = \{\}$
$\delta(s_2, 0) = \{s_3\}$	$\delta(s_4, 0) = \{s_4\}$
$\delta(s_2, 1) = \{\}$	$\delta(s_4, 1) = \{s_4\}$
$\delta(s_2, \varepsilon) = \{s_3\}$	$\delta(s_4, \varepsilon) = \{\}$

Formal description of NFA operation

NFA $M = (Q, \Sigma, \delta, q_0, F)$
accepts a string $w = w_1w_2w_3\ldots w_n \in \Sigma^*$
if w can be written (by inserting ε's) as:

$$y = y_1y_2y_3\ldots y_m \in (\Sigma \cup \{\varepsilon\})^*$$

and ∃ sequence $r_0,r_1,\ldots,r_m$ of states for which

- $r_0 = q_0$
- $r_{i+1} \in \delta(r_i, y_{i+1})$ for $i = 0,1,2, \ldots, m-1$
- $r_m \in F$