1

1/10/22

CSCE 355

Stephen Fenner

fenner@cse.sc.edu

https://cse.sc.edu/~fenner/csce355/index.html

my homepage

TA: Canyu Zhang

---

2 Midterms
1 final
Written HW
Programming project

Mathematics of Computation

- Define abstract models of computation
  - Finite Automata
  - Grammars
  - Turing Machines

1930s Foundations of math | Computability theory — what problems are computable, given unlimited resources?

(2) Computational Complexity —
difficulty

~~what is~~ which problems are feasibly computable, given reasonable but limited resources?

Represent all inputs/outputs to computations as strings (finite sequences of characters from some fixed alphabet).

Alphabets:

256 — ASCII char set
26 — lowercase letters
2 — $\boxed{\{0,1\}}$ — binary alphabet
1 — $\{0\}$ — unary alphabet

Def: An alphabet is any nonempty, finite set. If Σ is an alphabet, we call its elements symbols (or letters, characters).

③

Def: Let Σ be an alphabet. A **string over Σ** is a finite sequence of symbols from Σ. Write strings just juxtaposing the symbols.

Ex: $\Sigma = \{a, b, c\}$

Strings		length
aa	—	2
baac	—	4
ccca	—	4
b	—	1
ε	—	0 (empty string)

ε is a place holder (constant) denoting the empty string (the unique string of length 0)

ε is not part of the alphabet!

"metasymbol"

"Let $\boxed{x} = abbc$"

variable that can denote a string

$x, y, z, \ldots$ vars

Never! → "Let $a = aabc$"

4 Concatenating strings

$x = aabc$

$y = baacc$

Concatenation of x with y:

$$xy = \underbrace{aabc}_{x}\underbrace{baacc}_{y}$$

String length: If x is a string, let $|x|$ to denote the length of x

Ex: $|aabc| = 4$

$|a| = 1$

$|\varepsilon| = 0$

"Obvious": $|xy| = |x| + |y|$

$xy \neq yx$ generally (not commutative)

But

$(xy)z = x(yz)$ $(= xyz)$

Concatenation is associative

For any string x, $\varepsilon x = x\varepsilon = x$.

(5) **Def**: For alphabet Σ, let Σ^* denote the set of all strings over Σ.

Ex: $\{0\}^* = \{\varepsilon, 0, 00, 000, \ldots, 0^n, \ldots\}$

Generally if $n \geq 0$ is an integer and x a string,

$$x^n := \underbrace{xx\cdots x}_{n \text{ times}}$$

"equals by definition"

$$x^0 = \varepsilon$$
$$x^1 = x$$
$$x^2 = xx$$
$$\vdots$$

Finite automaton ("device")

takes Input string, say aabcab

reads it left to right, one symbol per step

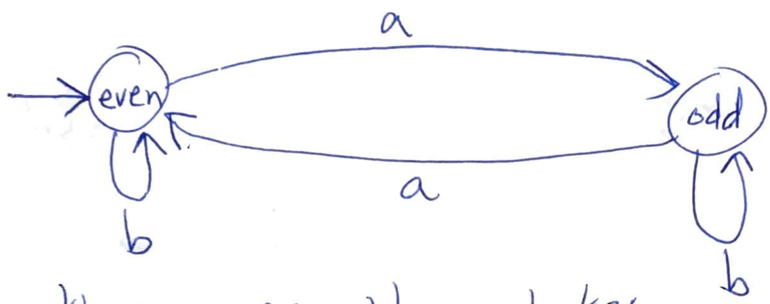
At each step, the device is in one of a fixed finite set of states.

From step to step, the device transitions

(6) to a new state, based on its current state & the scanned symbol.

Transition diagram (digraph)

$\Sigma = \{a, b\}$



Input: abbab

→ vertices are the states
transitions are directed edges
- labeled with alphabet symbols.

$\longrightarrow \bigcirc$ start state (state the device ~~s~~ starts in

Input: abbab

$\text{even} \xrightarrow{a} \text{odd} \xrightarrow{b} \text{odd} \xrightarrow{b} \text{odd} \xrightarrow{a} \text{even} \xrightarrow{b} \text{even}$

Input aaa

$\text{even} \xrightarrow{a} \text{odd} \xrightarrow{a} \text{even} \xrightarrow{a} \text{odd}$

Final state ~~tess~~ tells us whether there are an even or odd # of a's in the string.

7 Output is
- the answer to a yes/no question
- a single bit (1 or 0)
- accept or reject

"Is there an odd number ~~of~~ a's in the input?"

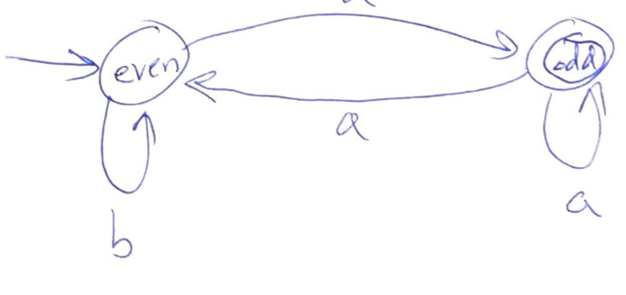
- ~~device to~~ accept means "yes"
- reject means "no"

Acceptance/rejection depends only on the final state.

Designate some states as accepting (other states are rejecting states).

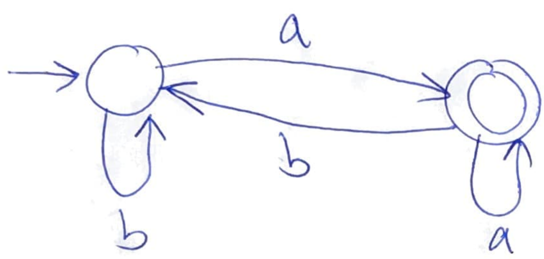
 ← accepting state

 ← rejecting state



accepts a string iff has an odd # of a's.

(8) <u>Ex</u>: "Does the input end in a?"



<u>Ex</u>: "Does the input start with a?"