January 22, 2019 10:58 AM

HW1 due tomorrow 10am

HW2 available

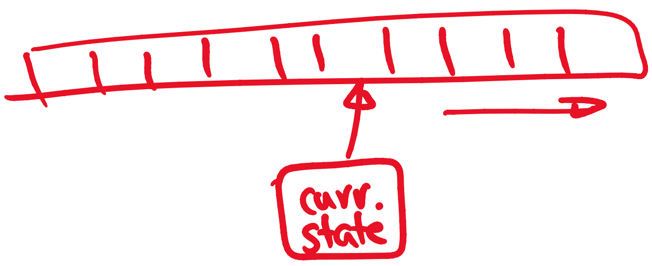
Q4: subsequence

bnn subseq of banana

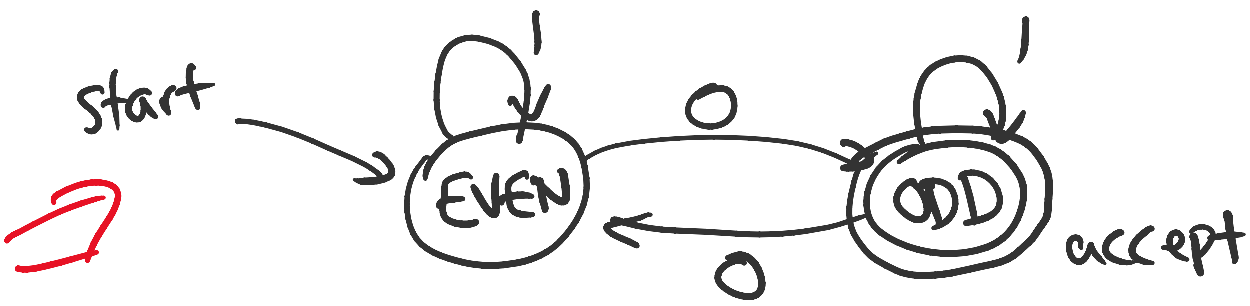
Deterministic Finite Automata (DFA)

machine/program that has
fixed amount of memory
& reads input from left to right

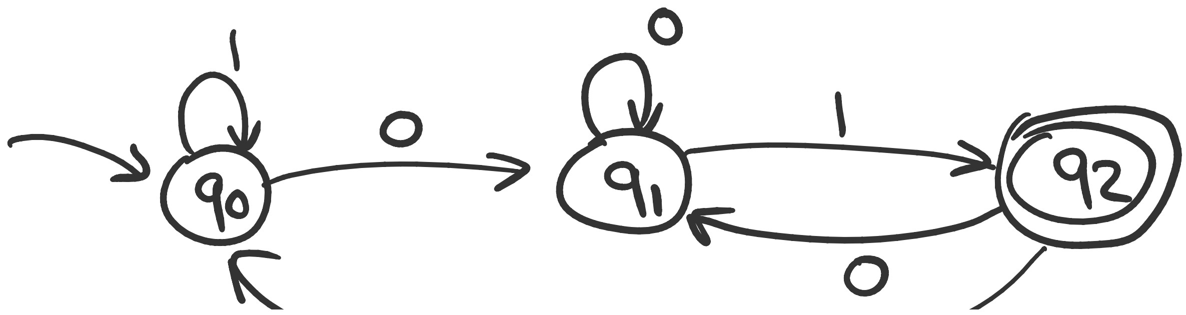
input

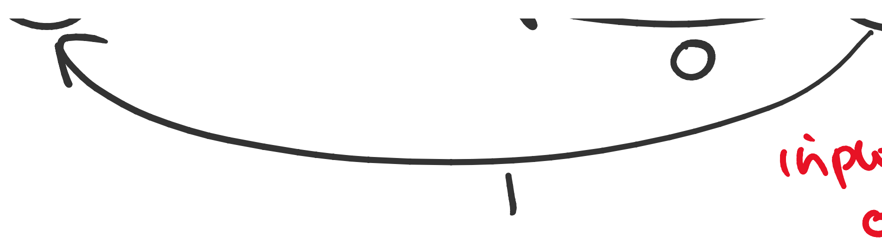


Ex0 all strings over {0,1} with
odd # of 0's



Ex1 all strings ending with 01





input: 01001

$q_0 \xrightarrow{0} q_1 \xrightarrow{1} q_2 \xrightarrow{0} q_1 \xrightarrow{0} q_1 \xrightarrow{1} q_2$

q_1 : just seen 0

q_2 : just seen 01

q_0 : none of above

Program:

```
state = q0;
while (not end of input) {
    c = next input symbol;
    if (state == q0 && c == 0) state=q1;
    else if (state==q0 && c==1) state=q0;
        ⋮
}
if (state == q2) output yes; else no;
```

Formal Def A DFA is specified by 5 things

$M = (Q, \Sigma, s, A, \delta)$ where

- Q is a finite set of states
- Σ is finite alphabet
- $s \in Q$ is the start state
- $A \subseteq Q$ is the set of accepting states
- $\delta : Q \times \Sigma \to Q$ is the transition fn

Ex1

$Q = \{q_0, q_1, q_2\}$

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

$s = q_0$

$A = \{q_2\}$

q	$\delta(q,0)$	$\delta(q,1)$
q_0	q_1	q_0
q_1	q_1	q_2

$s = q_0$

$A = \{q_2\}$

q_1	q_1	q_2
q_2	q_1	q_0

Def Define the <u>extended</u> transition fn
$\delta^* : Q \times \Sigma^* \to Q$ inductively:

(i) $\delta^*(q, \varepsilon) = q$

(ii) $\delta^*(q, x) = \delta^*(\delta(q,a), y)$ $\quad x = ay$, $a \in \Sigma$, $y \in \Sigma^*$

Ex 1
$$\begin{aligned}\delta^*(q_0, 01) &= \delta^*(\delta(q_0, 0), 1)\\ &= \delta^*(q_1, 1)\\ &= \delta^*(\delta(q_1, 1), \varepsilon)\\ &= \delta^*(q_2, \varepsilon) = q_2\end{aligned}$$

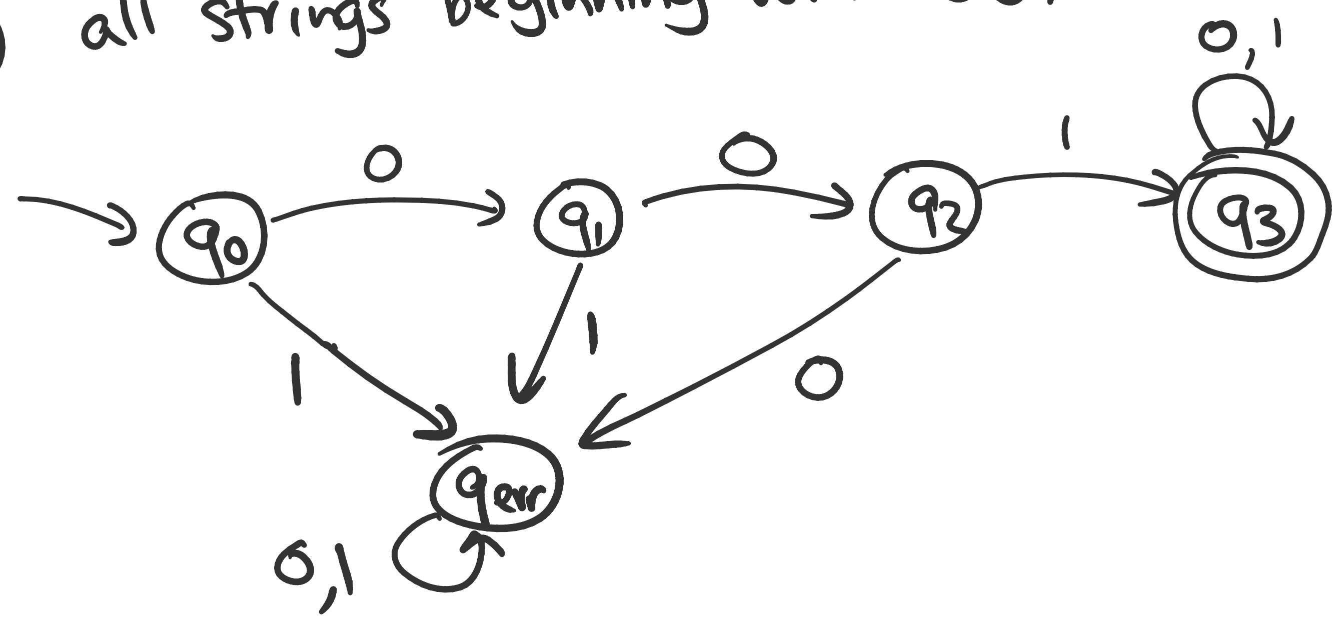
$\delta^*(q_0, 01001) = q_2$, $\quad 01001 \in L(M)$

Def Define $L(M) = \{x \in \Sigma^* \mid \delta^*(s, x) \in A\}$

language accepted by M

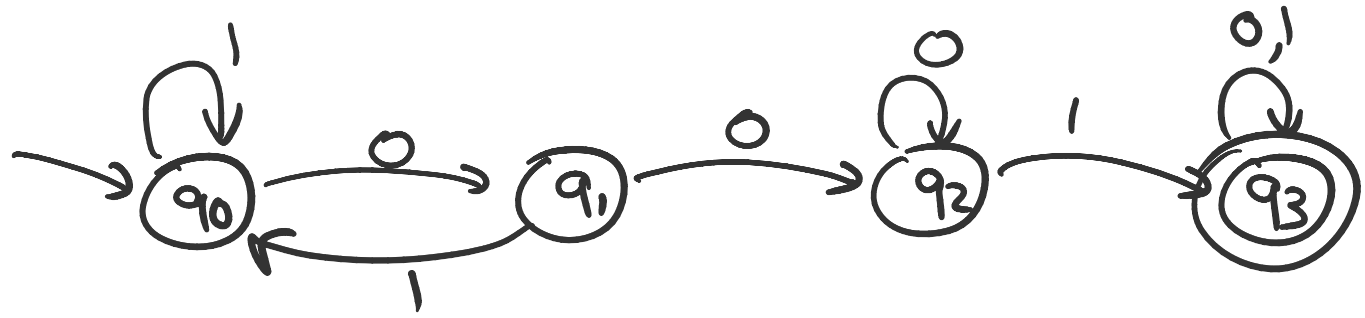
Exs $(\Sigma = \{0, 1\})$

a) all strings beginning with 001



b) all strings containing 001 as substring

b) all strings containing 001 as substring

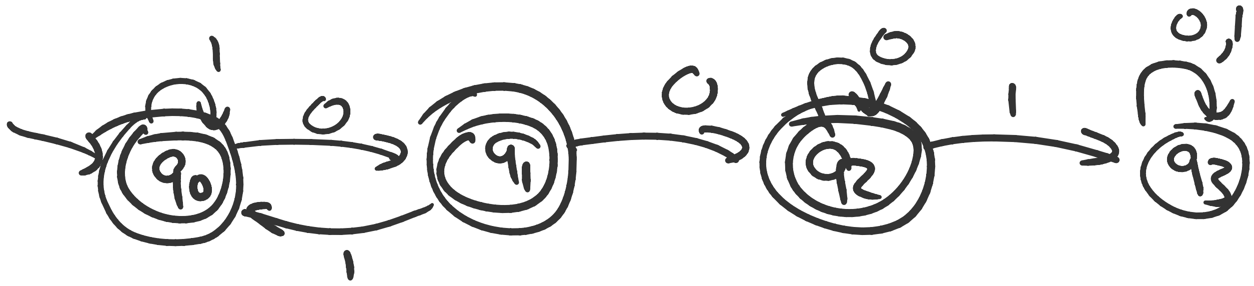


$\Rightarrow$

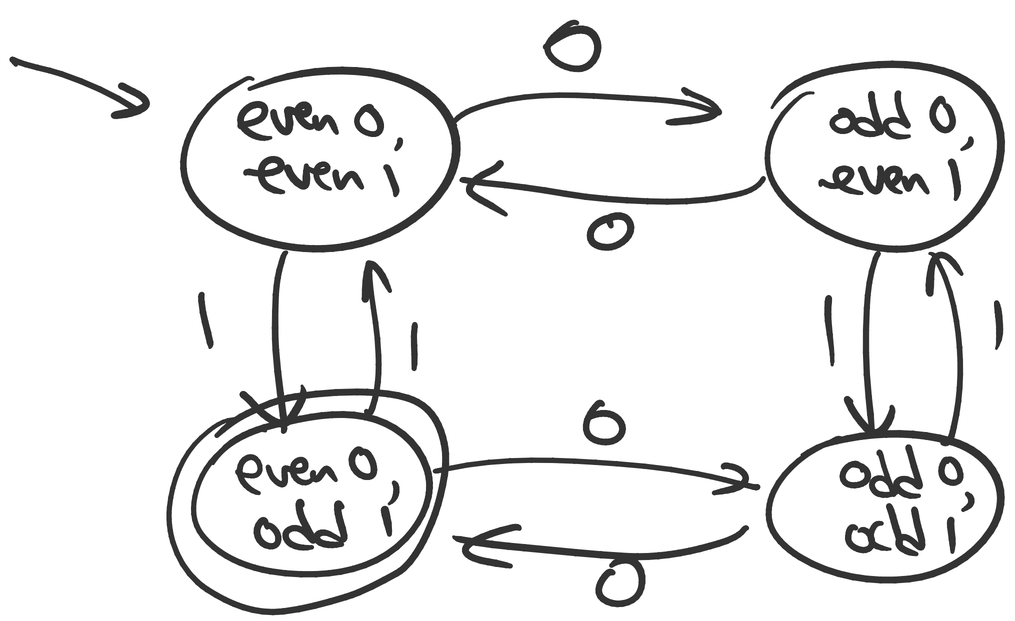
q_3: found 001
q_2: just seen 00 (but found 001)
q_1: just seen 0 but not 00
q_0: none of above

note: generalizes to other pattern strings
$\Rightarrow$ pattern matching algm in $O(n)$ time

c) all strings <u>not</u> containing 001



d) all strings with even # 0's <u>and</u> odd # 1's



Closure Properties

Thm If L is accepted by DFA M,
then $\overline{L}$ is accepted by some DFA M'.

Pf: idea - complement the accepting states

Given $M = (Q, \Sigma, s, A, \delta)$
construct $M' = (Q, \Sigma, s, A', \delta)$
where $A' = Q \setminus A$

Then $x \in L(M') \iff \delta^*(s,x) \in A'$
$\iff \delta^*(s,x) \notin A$
$\iff x \notin L(M)$
$\iff x \in \overline{L}$. $\square$

Thm If L_1 is accepted by DFA M_1,
L_2 is " " " M_2,
then $L_1 \cap L_2$ is accepted by some DFA M.

Pf: idea - remember a pair of states

called product construction

Given $M_1 = (Q_1, \Sigma, s_1, A_1, \delta_1)$,
$M_2 = (Q_2, \Sigma, s_2, A_2, \delta_2)$

Construct $M = (Q, \Sigma, s, A, \delta)$
where $Q = Q_1 \times Q_2$

$s = (s_1, s_2)$

$A = A_1 \times A_2$

$(A \times B = \{(a,b) \mid a \in A, b \in B\}$

$|A \times B| = |A| \cdot |B|$

$\delta : Q \times \Sigma \to Q$

$$\delta((q_1,q_2),a) = (\delta_1(q_1,a),\ \delta_2(q_2,a))$$

Lemma: $\delta^*((q_1,q_2),x) = (\delta_1^*(q_1,x),\ \delta_2^*(q_2,x))$

Pf: by boring induction ...

Then $x \in L(M) \iff \delta^*((s_1,s_2),x) \in A$

$\iff (\delta_1^*(s_1,x),\ \delta_2^*(s_2,x)) \in A_1 \times A_2$

$\iff \delta_1^*(s_1,x) \in A_1$ <u>and</u>
$\delta_2^*(s_2,x) \in A_2$

$\iff x \in L_1 \cap L_2$. $\square$

Ex all strings containing 001
<u>and</u> having odd # 0's