Tuesday, February 1, 2022 10:05 AM

Last week:

Deterministic Finite Automata (DFA)

(Machine / Program)

- constant memory $O(1)$
- Reads i/p L to R once

Closure Properties DFA

Thm: If L_1 accepted by DFA M_1

L_2 " " " M_2

Then (i) $L_1 \cap L_2$ " " some DFA M ← Product construction

(ii) $\bar{L}_1$ " " " " M'

Corr: It follows that

$L_1 \cup L_2$

$L_1 \oplus L_2$ are all possible "easily"

$L_1 \setminus L_2$

⋮

$$L_1 \cdot L_2 = \left\{ xy \;\middle|\; \begin{array}{l} x \in L_1 \ \& \\ y \in L_2 \end{array} \right\}$$

☆ Want to show Reg languages ⟷ Languages accepted by DFA ☆

☆ Non deterministic Finite Automata (NFA)

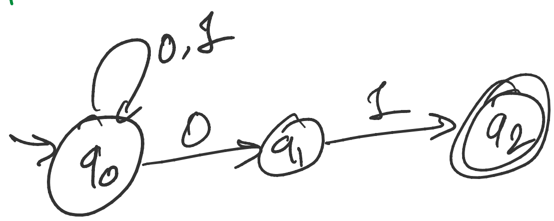
(i) allow choices

* Non-deter...

(i) allow choices

(ii) allow ε-transitions

Ex1: All strings that ends with 01.



$\delta^*(q_0, 01) = \{q_0, q_2\}$

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

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

Not a valid DFA, e.g. $\delta(q_0, 0) = q_0$ or q_1?

$\delta(q_1, 0) = ?$

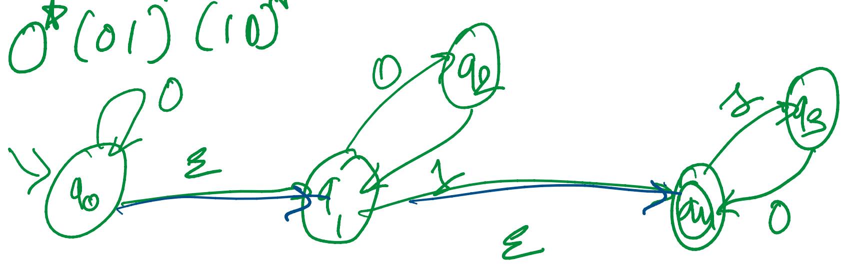
But a valid NFA!

Accept iff $\exists$ a path leading to an accept state.

eg. 00101: $q_0 \xrightarrow{0} q_0 \xrightarrow{0} q_0 \xrightarrow{1} q_0 \xrightarrow{0} q_1 \xrightarrow{1} q_2$ ✓

Reject iff $\forall$ path lead to a "non-accept" state

Ex 2 $0^*(01)^*(10)^*$



$\delta^*(q_0, 00) = \{q_0, q_1, q_4, q_2\}$ $\quad q_0 \xrightarrow{0} q_0 \xrightarrow{0} q_0 \xrightarrow{\varepsilon} q_1 \xrightarrow{\varepsilon} q_4$

A Formal Definition:

Finite set of states

start state

★ <u>Formal Definition</u>:

An NFA is $M = (Q, \Sigma, s, \delta, A)$

- Q: Finite set of states
- s: start
- Σ: set of symbols
- A: set of accept states

Like DFA EXCEPT

(In DFA $\delta : Q \times \Sigma \to Q$)

$$\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to P(Q)$$

$P(Q)$: Power set of Q. $= \{$all subsets of $Q\}$

Ex1: $\delta(q_0, 0) = \{q_0, q_1\}$

$\delta(q_1, 0) = \{\}$

Ex2: $\delta(q_0, \varepsilon) = \{q_1\}$

<u>Def</u>: Given $q \in Q$ define ε-reach(q) inductively

(i) q in ε-reach(q)

(ii) if $q' \in \varepsilon$-reach(q) & $q'' \in \delta(q', \varepsilon)$ Then q'' in ε-reach(q)

$$q' \xrightarrow{\varepsilon} q''$$

eg. ε-reach(q_0) in Ex2 is $\{q_0, q_1, q_4\}$

Define Extended transition Fun^c inductively

$$\delta^* : Q \times \Sigma^* \to P(Q)$$

ε-reach(q)

$\delta^*: Q \times \Sigma^* \rightarrow \mathcal{P}(Q)$

(i) $\delta^*(q, \varepsilon) = \varepsilon\text{-reach}(q)$

(ii) $\delta^*(q, x) = \bigcup_{q' \in \varepsilon\text{-reach}(q)} \; \bigcup_{q'' \in \delta(q', a)} \delta^*(q'', y)$

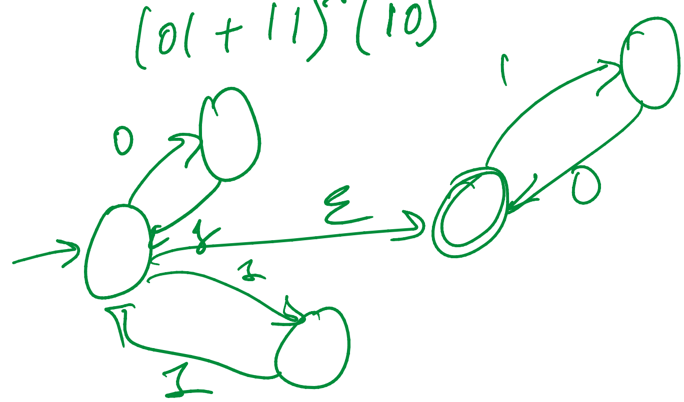
$x = ay$

$a \in \Sigma$

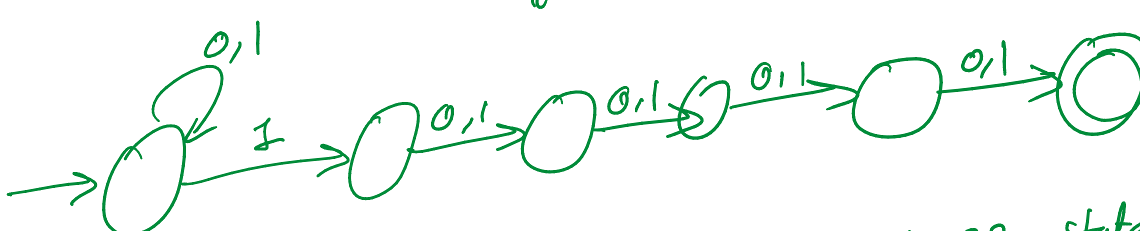
$y \in \Sigma^*$

Ex1: $\delta^*(q_0, 01) = \{q_0, q_1\}$

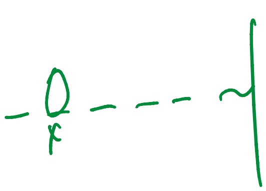
Ex a) $(01 + 11)^*(10)^*$



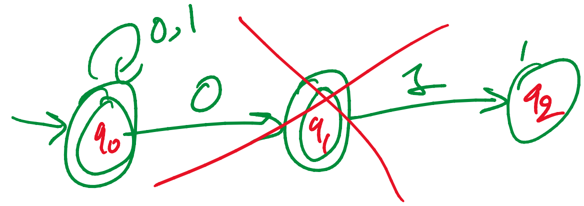
b) all strings whose 5^{th} character from right is a 1.



(A DFA has atleast 32 states!)



c) all strings not ending 01



WRONG

01 $\quad q_0 \xrightarrow{0} q_0 \xrightarrow{1} q_1$

$\{0, \cdot, *\}$

★ Regular Language $\longrightarrow$ NFA.

Thm: If L_1 accepted by NFA M_1

L_2 " " " M_2

Then

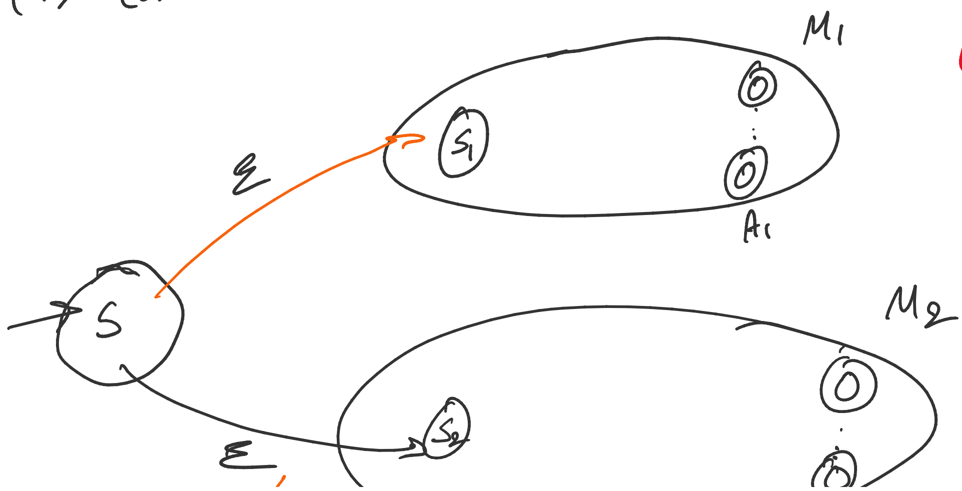
(i) $L_1 \cup L_2$ " " some NFA M

(ii) $L_1 \cdot L_2$ " " " " M'

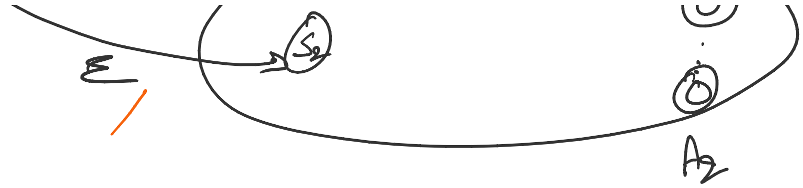
(iii) L_1^* " " " " M''

pf: (By picture)

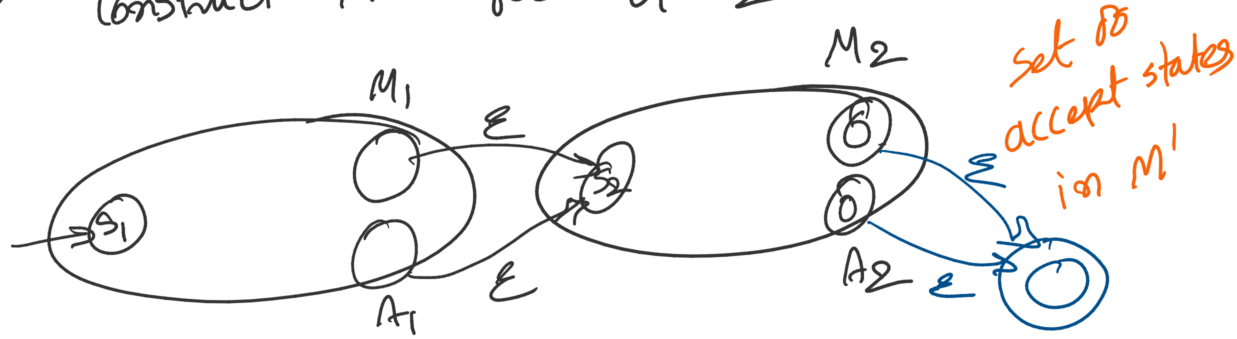
(i) Construct M for $L_1 \cup L_2$



$Q = Q_1 \cup Q_2 \cup \{s\}$



(ii) Construct M' for $L_1 \cdot L_2$



(iii) M'' for L_1^*