GPS 2: Tue, Sep 8

HW 2: Wed, Sep 9

may be clear from context

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

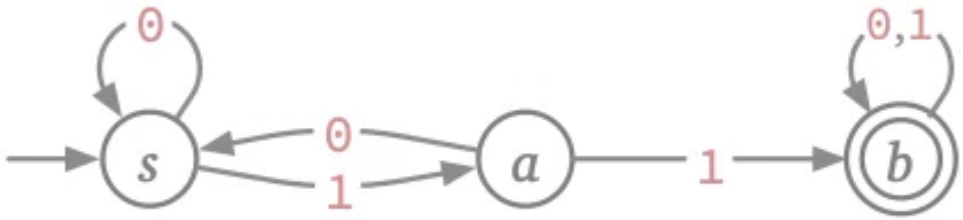
- Σ: input alphabet
- Q: finite set of states
- $\delta : Q \times \Sigma \to Q$ the transition function
- $s \in Q$ start state
- $A \subseteq Q$ accept states

Define extended transition function $\delta^* : Q \times \Sigma^* \to Q$

$$\delta^*(q, w) := \begin{cases} q & \text{if } w = \epsilon \\ \delta^*(\delta(q, a), x) & \text{if } w = ax \end{cases}$$

The language of DFA $M = (Q, \Sigma, s, A)$ is

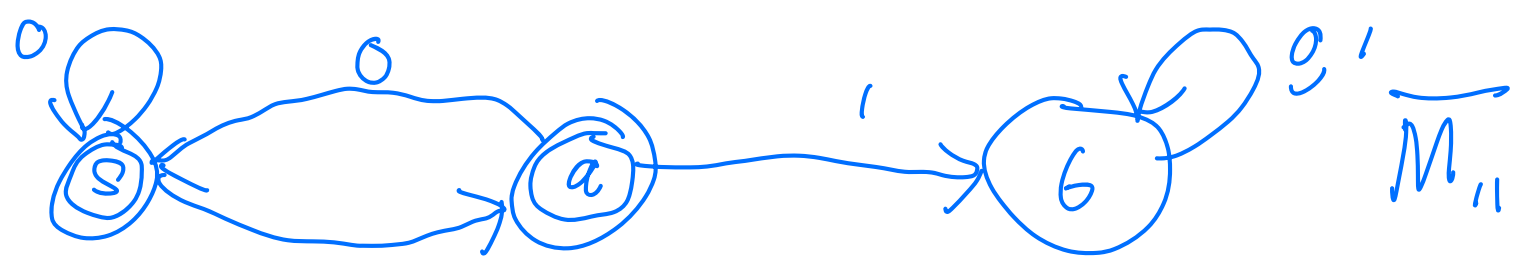
$$L(M) := \{w \mid M \text{ accepts } w\}$$
$$= \{w \mid \delta^*(s, w) \in A\}$$



M_{11}

$L(M_{11})$: binary strings with 11 as a substring

If we take the complement of the accept states but keep the rest the same...

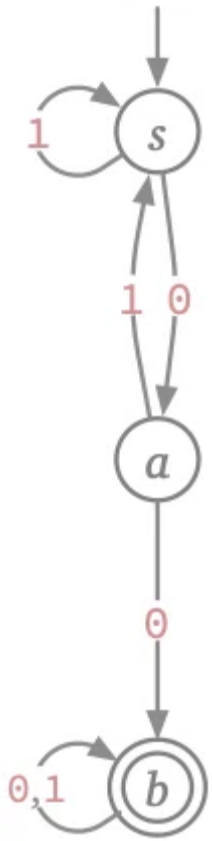


$$L(\bar{M}_{11}) = \overline{L(M_{11})} = \Sigma^* \setminus L(M_{11})$$

More generally, given $M = (Q, \delta, s, A)$,
there is a machine $\bar{M} = (Q, \delta, s, \bar{A})$

$\bar{A} = Q \setminus A$

s.t. $L(\bar{M}) = \overline{L(M)}$



M_{00}

$L(M_{00})$: binary strings with 00 as a substring.

$M_{11} = (Q_{11}, \delta_{11}, s_{11}, A_{11})$

$M_{00}\ (Q_{00}, \delta_{00}, s_{00}, A_{00})$

$M^{\otimes}$: states: pairs (p, q) with $p \in Q_{00}$

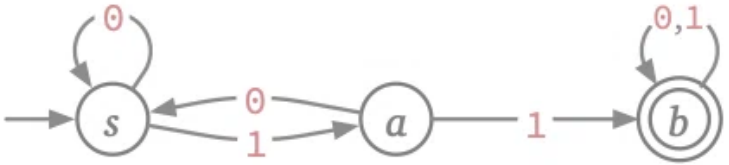
$q \in Q_{11}$

transitions $(p, q) \xrightarrow{a} (p', q')$ $\quad p' = \delta_{00}(p, a)$

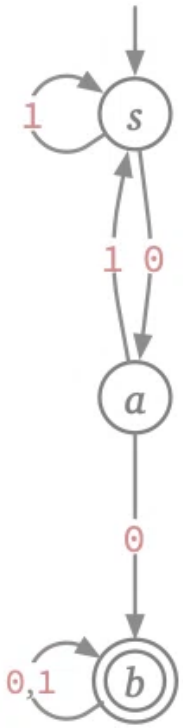
$q' = \delta_{11}(q, a)$

start at (s_{00}, s_{11})

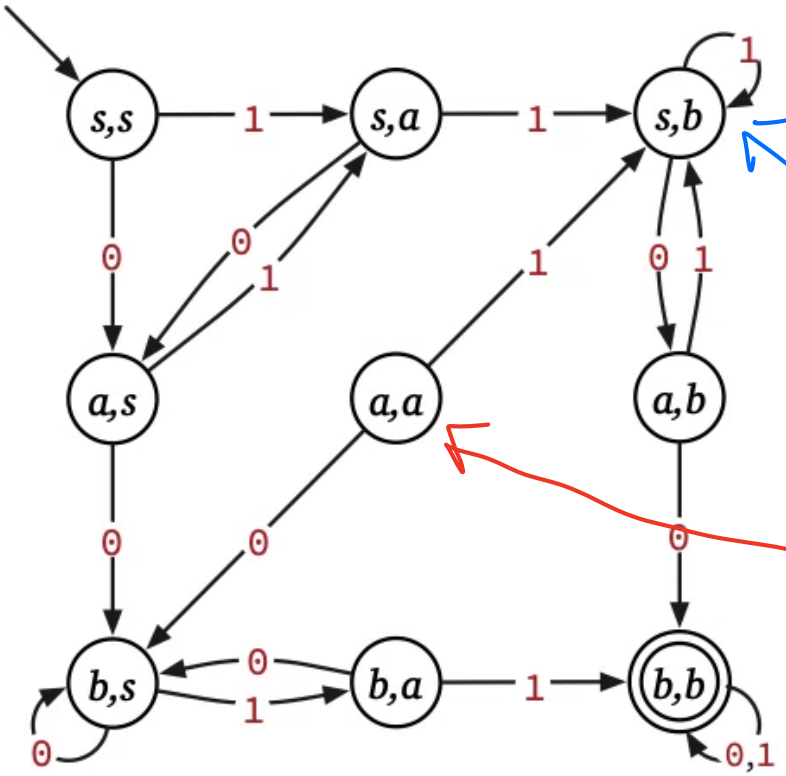
accept at $(6, 6)$ (I mean $6 \in A_{00}$ $6 \in A_{11}$)



M_{11}



M_{00}



$s \in Q_{00}$
6 from Q_{11}

$M^{\otimes}$

unreachable!
oh well

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

a <u>product construction</u> (or product) of M_1 & M_2 is a machine $M = (Q, \delta, s, A)$ s.t.

$Q = Q_1 \times Q_2 = \{(p, q) \mid p \in Q_1 \text{ & } q \in Q_2\}$,

$\delta((p, q), a) = (\delta(p, a), \delta(q, a))$ &

$s = (s_1, s_2)$

I DID NOT DEFINE A HERE!

<u>Lemma</u>: $\delta^*((p, q), w) = (\delta^*(p, w), \delta^*(q, w))$

Different choices for A give different languages...

$L(M_1) \cap L(M_2)$: Set $A = \{(p,q) \mid p \in A_1 \text{ and } q \in A_2\}$

$L(M_1) \cup L(M_2)$: Set $A = \{(p,q) \mid p \in A_1 \text{ or } q \in A_2\}$

$L(M_1) \setminus L(M_2)$: Set $A = \{(p,q) \mid p \in A_1 \text{ but } q \notin A_2\}$

$L(M_1) \oplus L(M_2)$: Set $A = \{(p,q) \mid p \in A_1 \text{ xor } q \in A_2\}$

IF DOING PRODUCT ON HW or EXAM, say what accept states are!!

A language L is "automatic" if there exists a DFA M s.t. $L = L(M)$.

regular

Thm: Let L_1 & L_2 be ~~automatic~~ languages (over the same alphabet), the following are ~~automatic~~. regular

Regular

"~~Automatic~~ languages are closed under union, ..."

- $L_1 \cup L_2$
- $L_1 \cap L_2$
- $L_1 \setminus L_2$
- $L_1 \oplus L_2$
- $\overline{L_1} = \Sigma^* \setminus L_1$

Thm (Kleene): A language L is automatic if and only if L is regular.

Cor: If M_1 & M_2 are DFAs, there is a DFA accepting

- $L(M_1) \cdot L(M_2)$
- $(L(M_1))^*$

If you want to show a language L is regular:

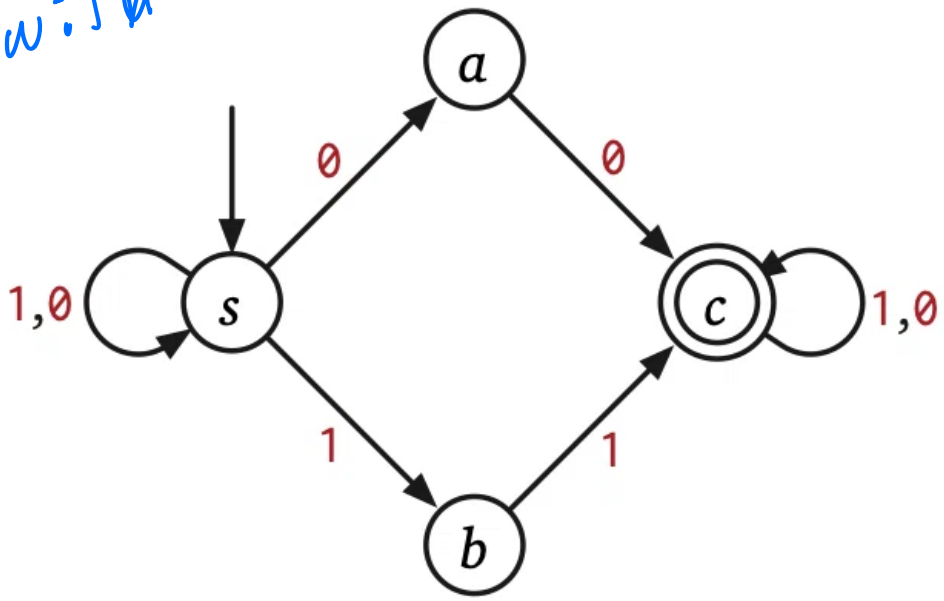
1) give a regular expression representing L
2) give a DFA accepting L

3) Build L from above operations on simpler languages

Can prove a language is <u>not</u> regular by arguing no DFA accepts it.

A DFA with choices...

Strings with 00 or 11



Ex: 0101001
Accept!

Ex: 0101
Reject!

If a sequence of choices matching the input string leads to an accept state, you accept.

nondeterministic finite-state automaton (NFA)

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

- Σ: input alphabet
- Q: finite set of states
- s: start state,
- $A \subseteq Q$: accept states
- $\delta : Q \times \Sigma \to 2^Q$ (i.e. $\mathcal{P}(Q)$)

extended transition function $\delta^* : Q \times \Sigma^* \to 2^Q$

$$\delta^*(q, w) := \begin{cases} \{q\} & \text{if } w = \varepsilon \\ \bigcup_{r \in \delta(q,a)} \delta^*(r, x) & \text{if } w = ax \end{cases}$$

$$L(M) := \{ w \mid \delta^*(s, w) \cap A \neq \emptyset \}$$

accept $w \in L(M)$

reject $w \notin L(M)$

Ex: Ends with 01101001