DISCRETE MATHEMATICS

LECTURE 12

Plan:

1. Reg Exp to NFA
2. NFA to DFA
3. Complementation

Reg Exp
↓
NFA
↓
DFA

Reference:

Section 1.2 and 1.3 of:

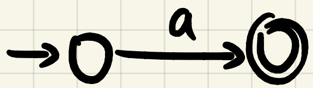
Introduction to the Theory of Computation

(third edition)

by Michael Sipser

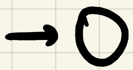
Regular expressions to NFA:

a

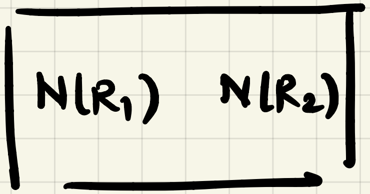


ϵ

Φ



$R_1 + R_2$



disjoint union of NFA (R_1) and NFA (R_2)

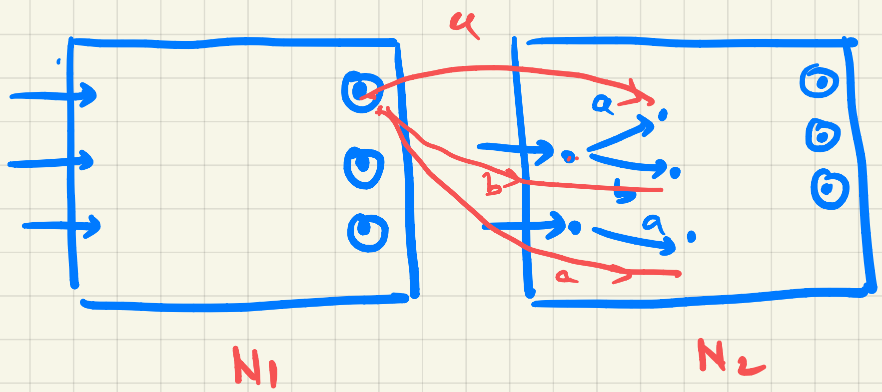
$R_1 \cdot R_2$?

R^* ?

$R_1 \cdot R_2$:

N_1 : NFA for R_1

N_2 : NFA for R_2



- 1. From every final state f of N_1 add transitions: $f \xrightarrow{a} q_1$, $f \xrightarrow{b} q_2$

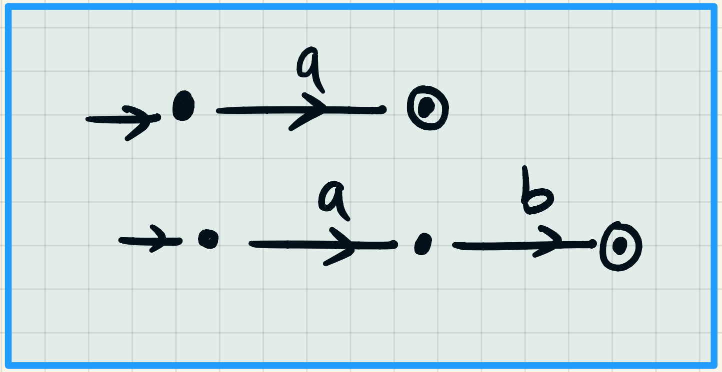
  to the 'a' and 'b' successors of every initial state of N_2

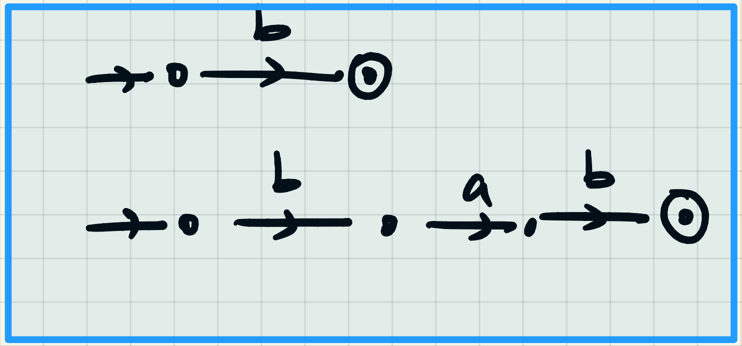
- 2. Make accept states of N_1 as non-accepting if ϵ is not in R_2

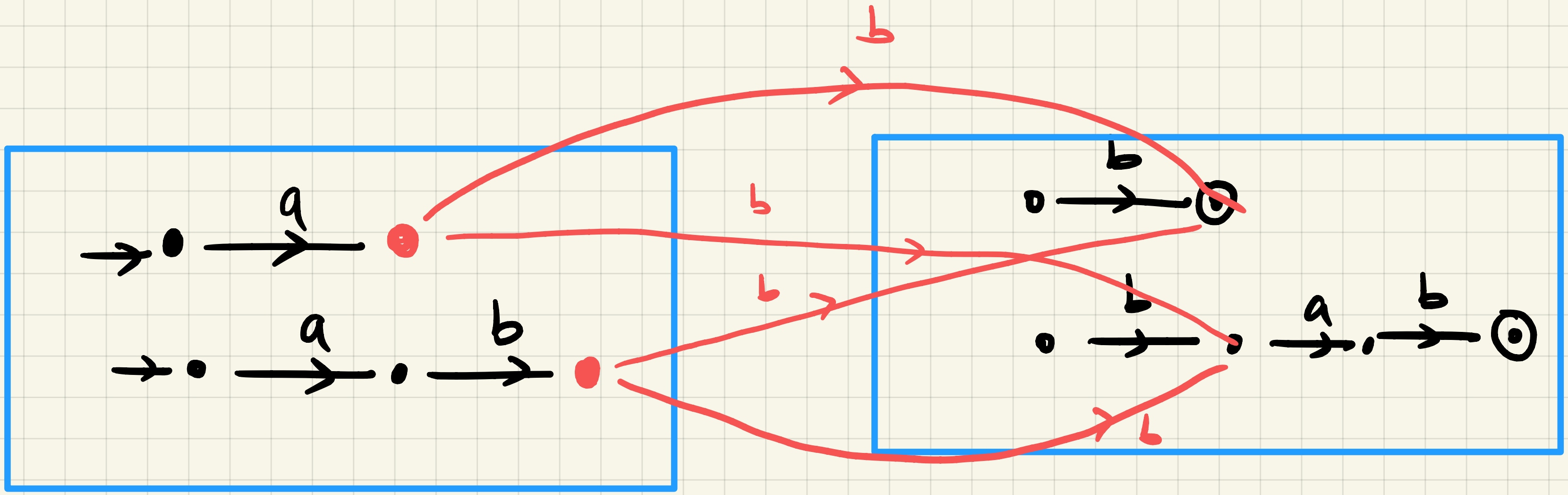
- 3. Make initial states of N_2 as non-initial if ϵ is not present in R_1.

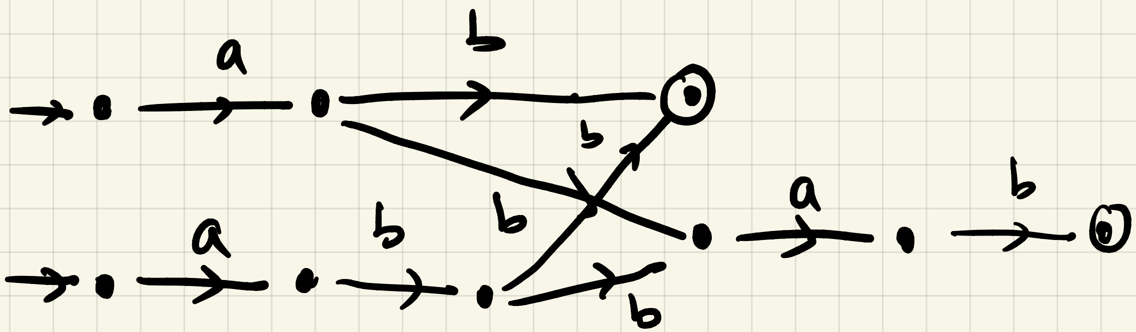
Example 1:

$R_1 : a + ab$ $\qquad$ $R_2 : b + bab$



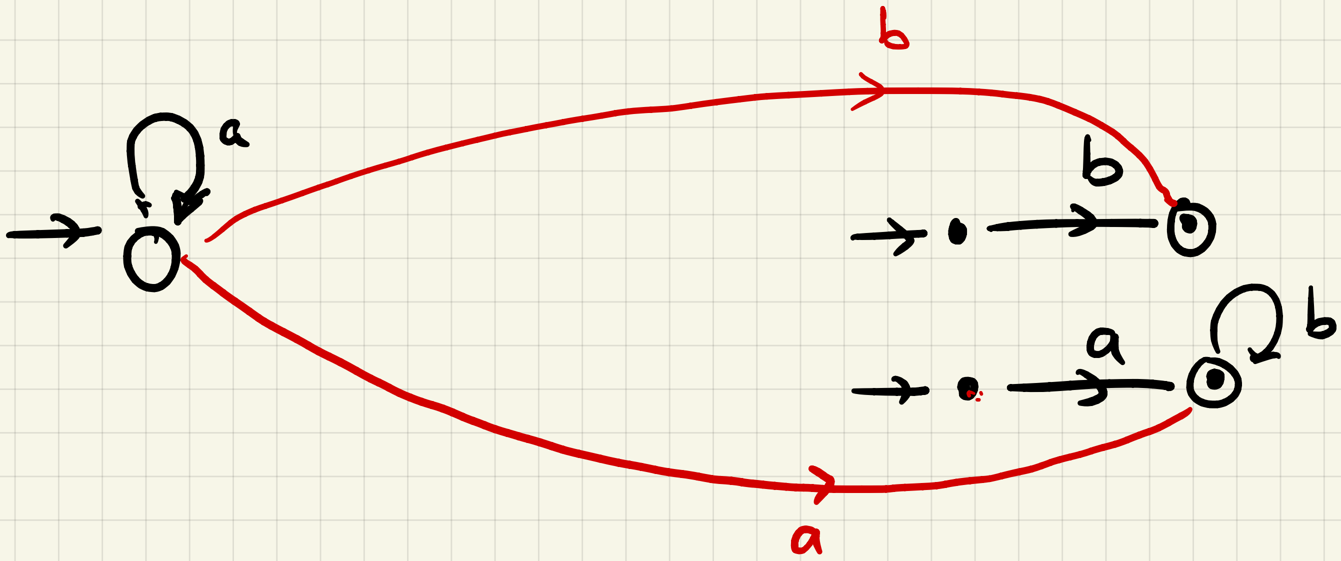




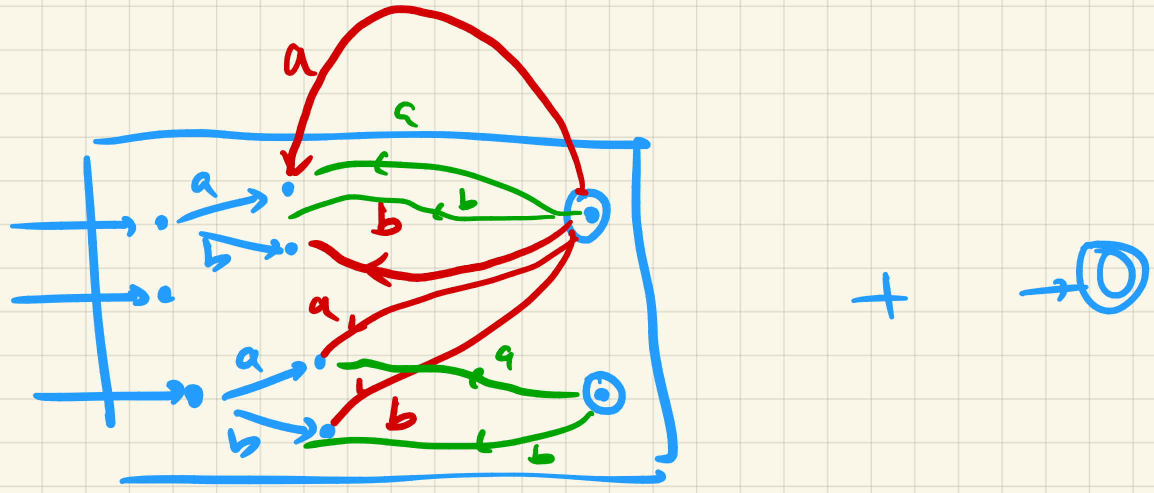


Example 2:

$R_1 : \quad a^*$ $\qquad\qquad$ $R_2 : \quad b + ab^*$



R^*:



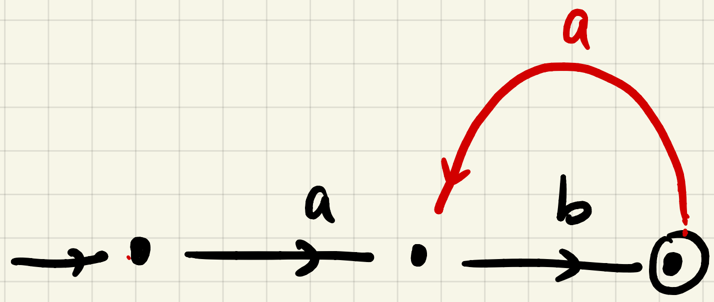
N for R

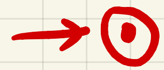
1. From every final state f of N, add transitions $f \xrightarrow{a} p$ to every a-successor of initial state

2. Add an automaton for ϵ.

Example 3:

$(ab)^*$

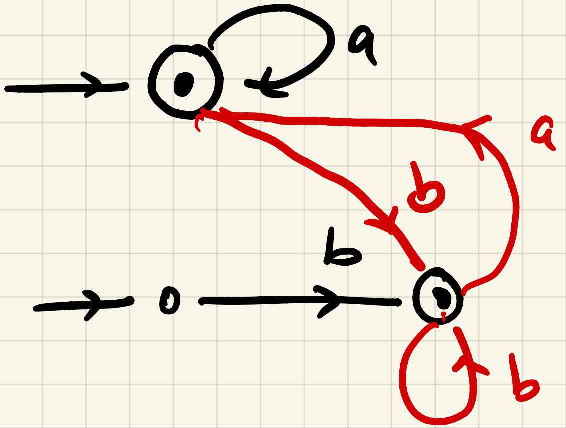




Example 4:

$(a^* + b)^*$

$L = \{ \overline{b, \epsilon, a, aa \ldots} \}^*$

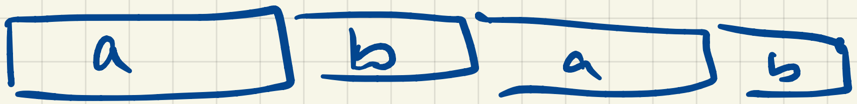


L^*

$= L^0 \cup L^1 \cup L^2 \ldots$

$\epsilon \quad L \quad \ldots \quad L^k$

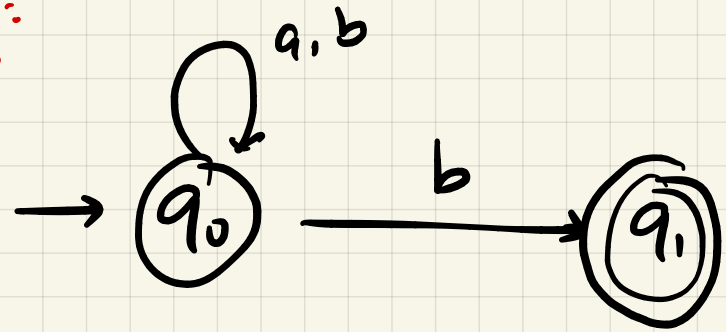
$\{ w_1 w_2 \ldots w_k \mid w_i \in L, \ k \geq 0 \}$

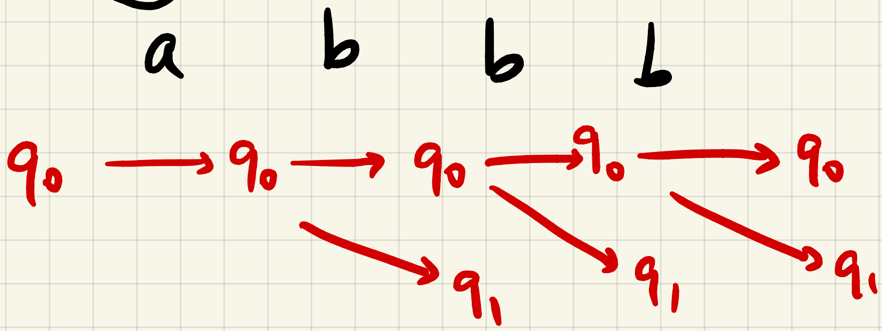


NFA to DFA:

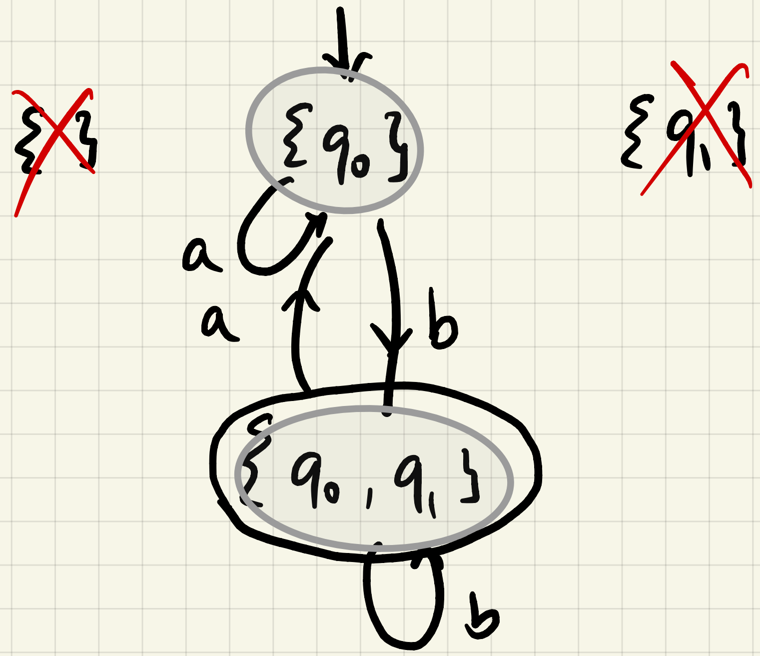
Example 5:

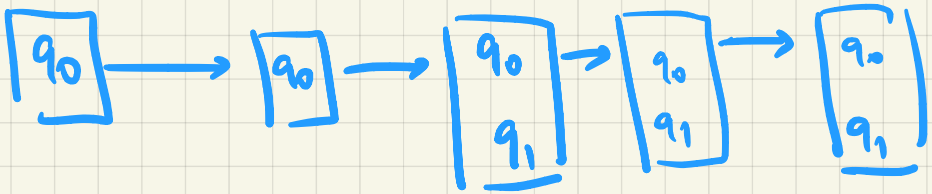
NFA:

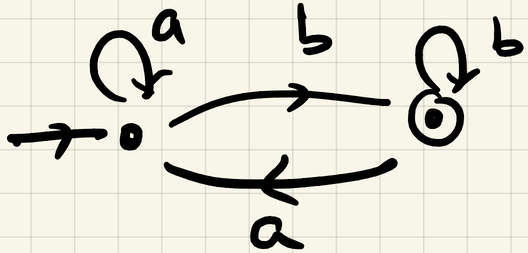




DFA: $\{q_0, q_1\}$

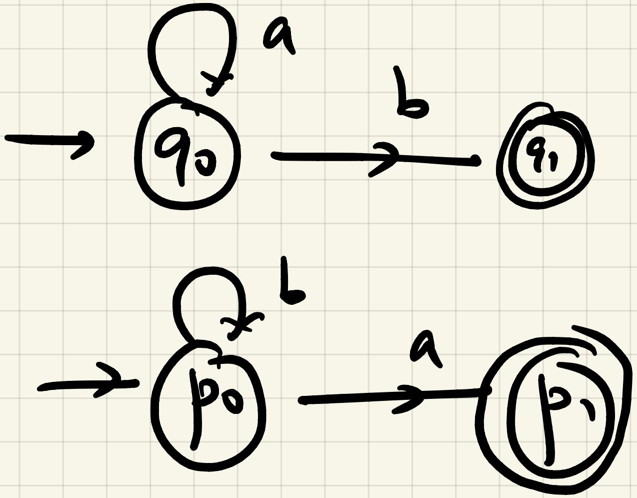






DFA.

Example 6:



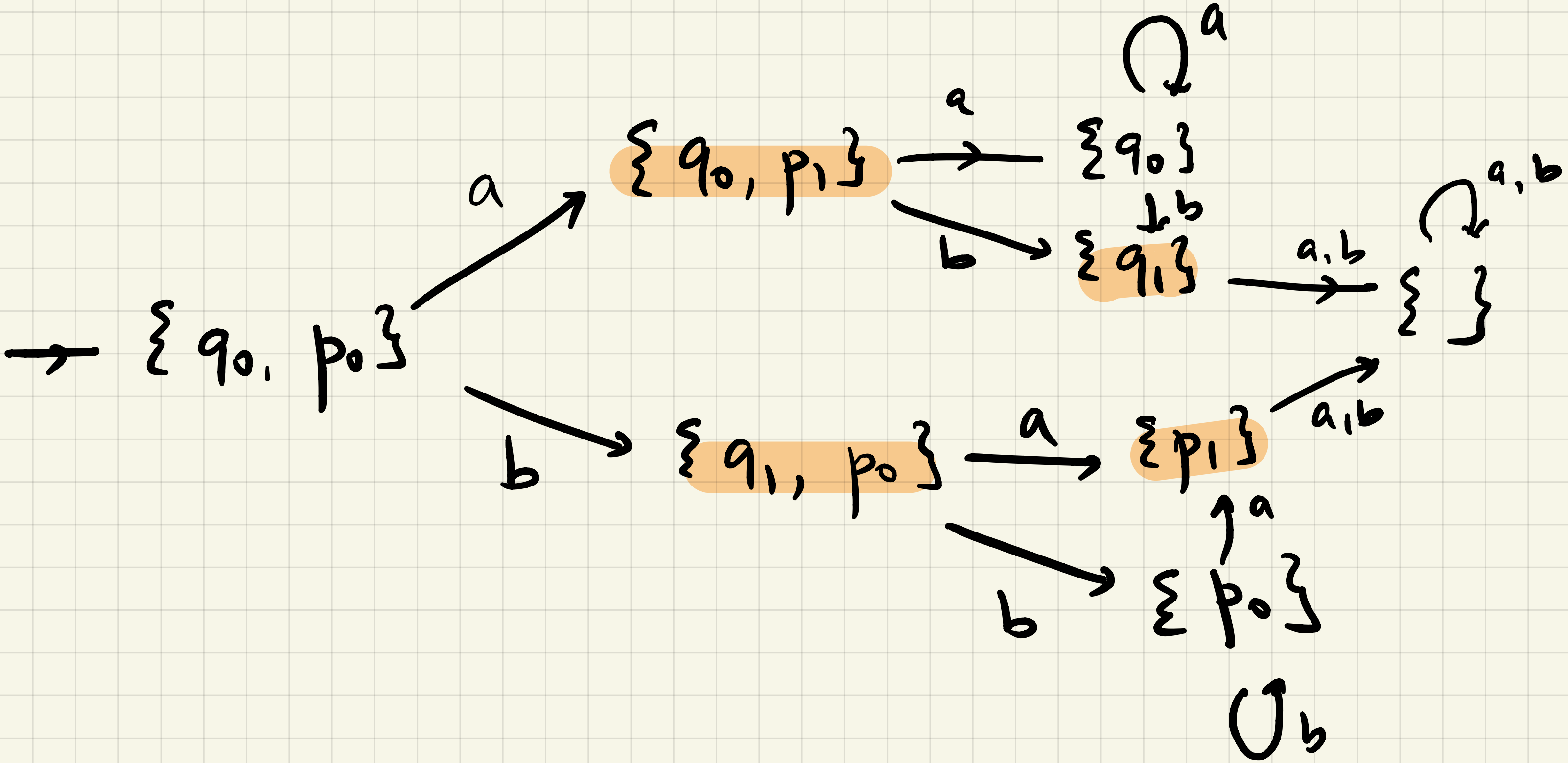
$\{q_0, p_0\}$

$q_0 \xleftarrow{a}$ q_0, $p_0 \xrightarrow{a} p_1$

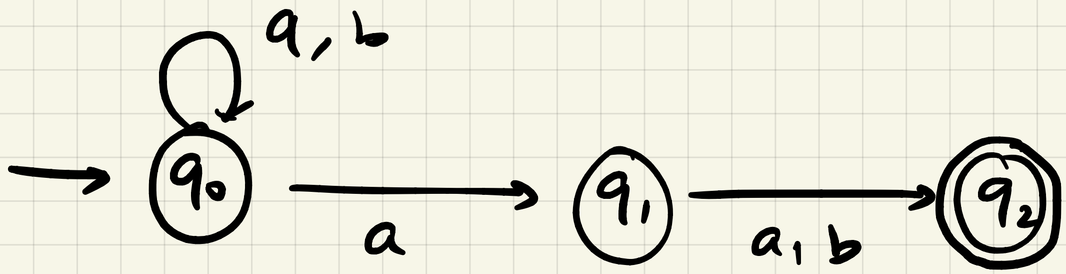
$\{q_0, p_1\}$

$q_0 \xrightarrow{a} q_0$, p_1: $\times$

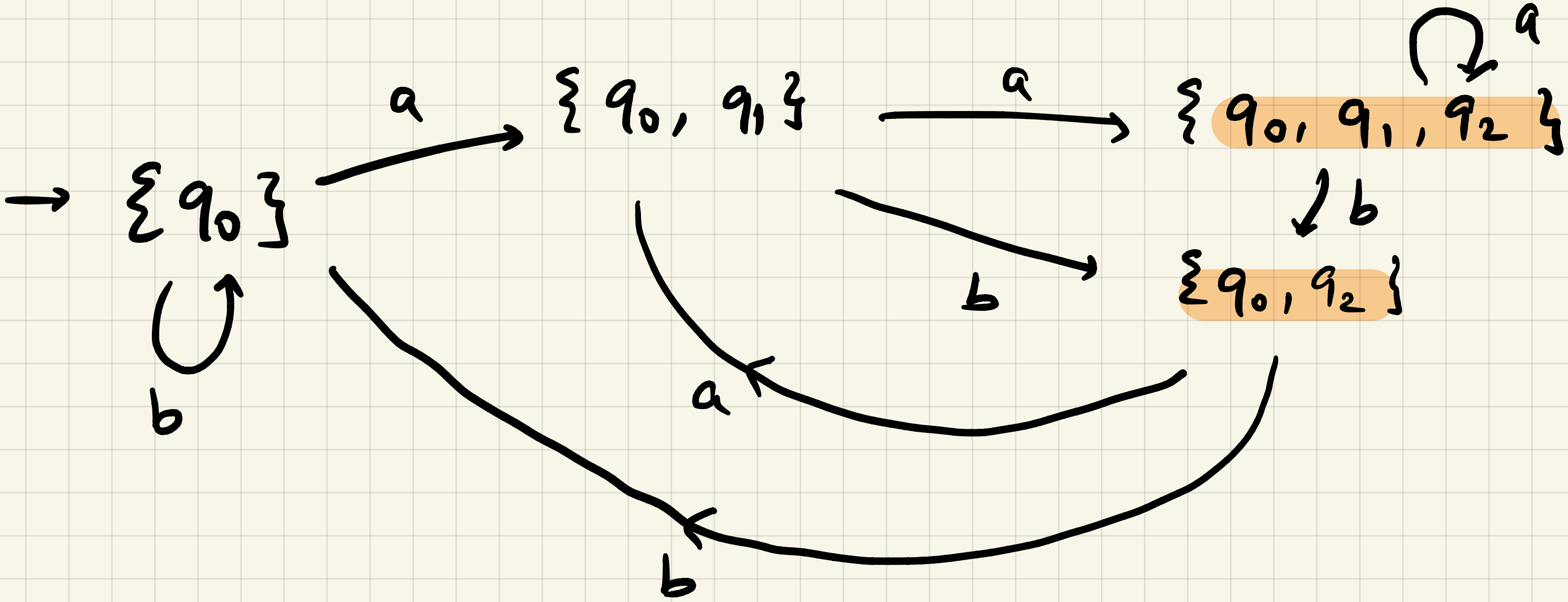
$\{q_0, q_1, p_0, p_1\}$



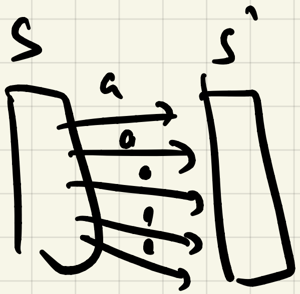
Example 7:



NFA.



$$S \xrightarrow{a} s' \quad = \quad s' = \{ q \mid p \xrightarrow{a} q \text{ for some } p \in s \}$$



NFA $\rightarrow$ DFA: Subset construction:

NFA $\mathcal{A}$: $(Q, \Sigma, Q_0, \Delta, F)$

Equivalent DFA $\mathcal{B} = (S, \Sigma, s_0, \delta, S_F)$ is given as follows:

- States S = all subsets of Q - Power set $\mathcal{P}(s)$
- Initial state s_0 = the subset Q_0
- Final states $S_F = \{ T \in \mathcal{P}(s) \mid T \cap F \neq \phi \}$
- Transitions: $T \xrightarrow{a} T'$

  if $T' = \{ q' \mid \exists q \in T \text{ with } q \xrightarrow{a} q' \}$

So far:

$$\text{Reg Exp} \downarrow \text{NFA} \xrightarrow{\text{(subset construction)}} \text{DFA}$$

Reg Exp

↓

NFA

↓ (subset construction)

DFA

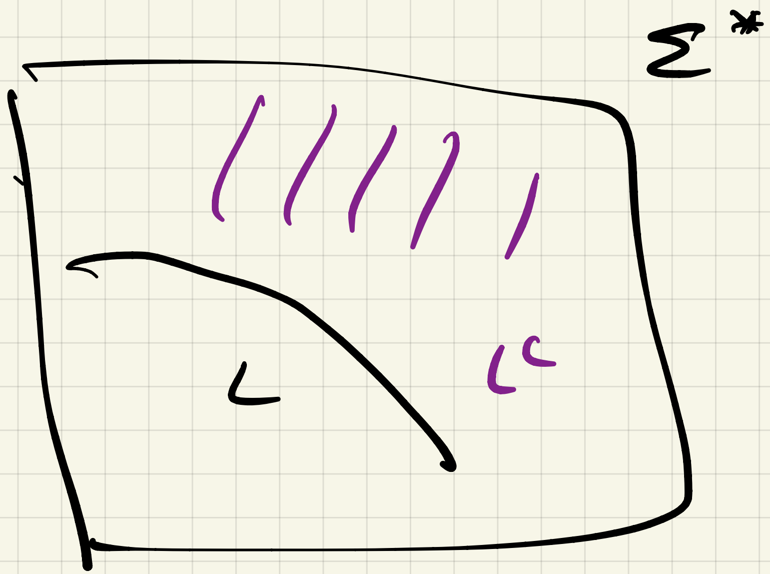
Remark: It is also possible to convert every NFA into a regular expression.

→ Will not do this in this course.

COMPLEMENTATION:

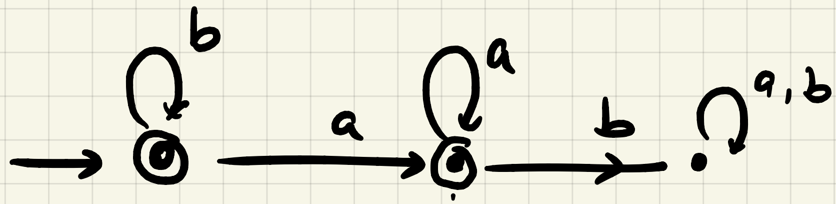
Operations on languages seen so far: union, concatenation, star.

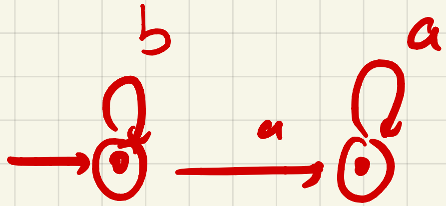
$$L^c = \{ w \in \Sigma^* \mid w \notin L \}$$



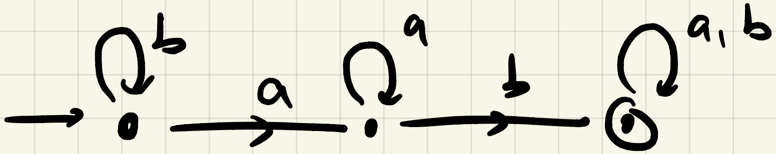
Example: Draw a DFA for

$$\{ w \in (a+b)^* \mid w \text{ does not contain 'ab'} \}.$$

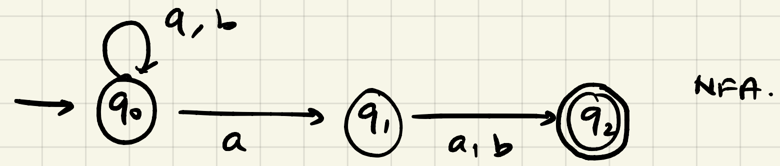


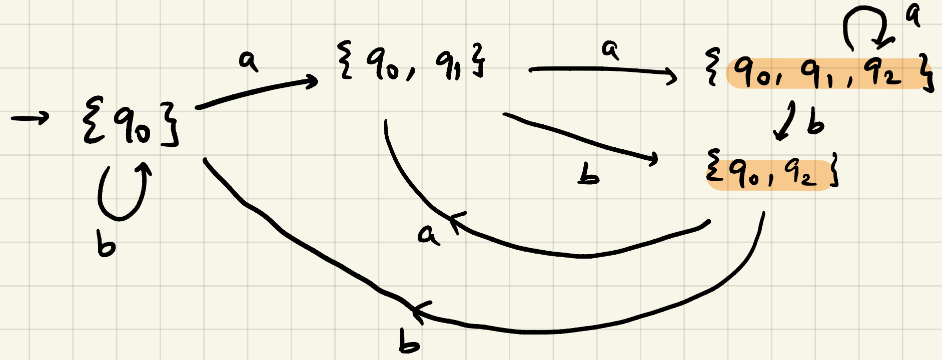


DFA for $\{ w \mid w \text{ contains 'ab'} \}$



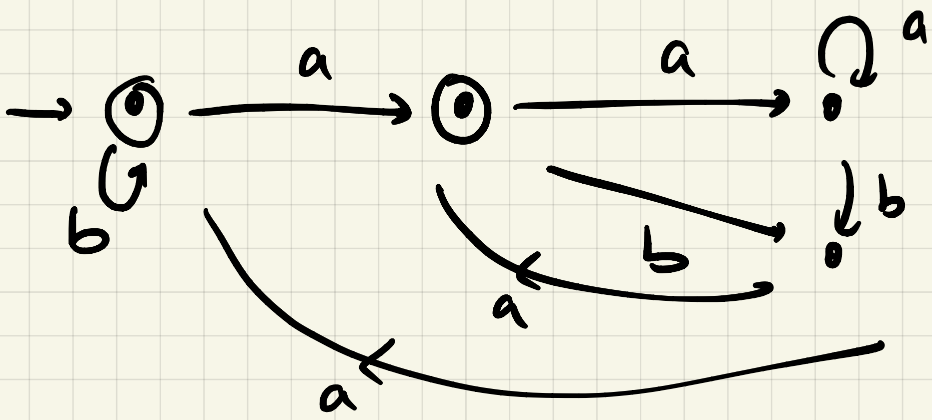
Example:





L: Second last letter is an 'a'

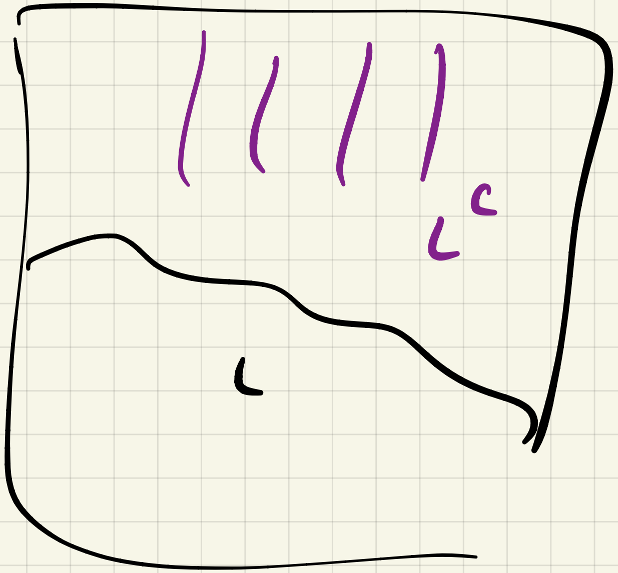
L^c: $\{\epsilon, a, b\}$ $\cup$ { second last letter is a 'b' }

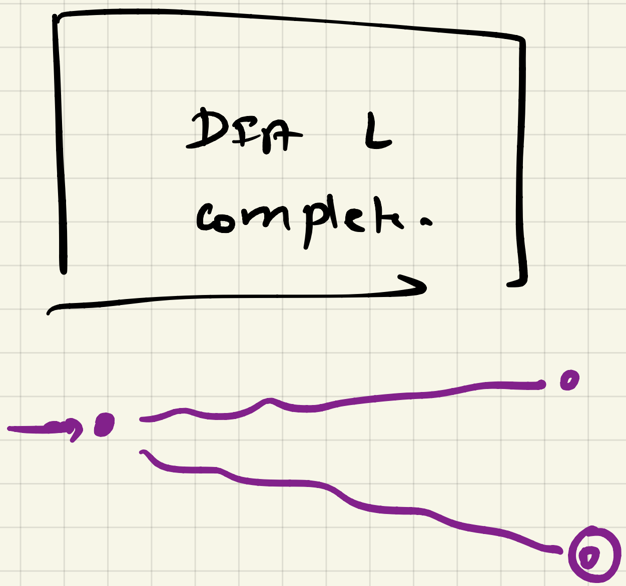


Complementing a language:

- L : Take a complete DFA for L
- Interchange accepting and non-accepting states.
- This gives the DFA for L^c.

Σ^*





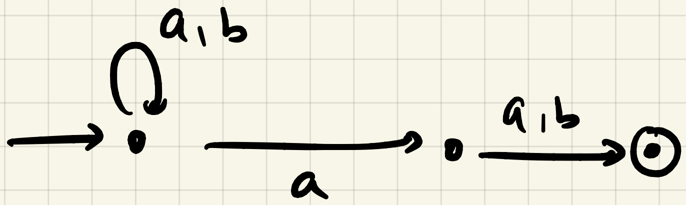
Why this works for complete DFA?

- For every word $w \in \Sigma^*$, there is a unique run that reads the whole word.

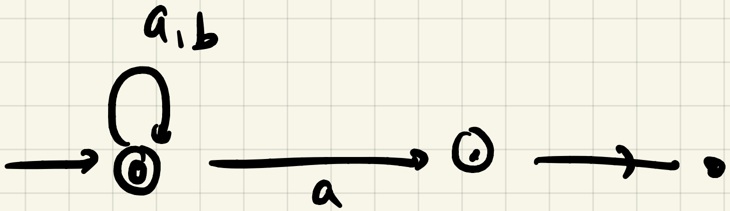
- For words in the language, the last state is accepting. For words in the complement, the last state is non-accepting.

- So interchanging accept and non-accepting states gives the complement of the language.

Why such a construction does not work for NFA?



L:



interchanging acc. & reject gives a different language, not the complement.

- Consider an NFA for L. For a word $w \in L$, there could be some runs that are accepting and some that are rejecting. Therefore, even after interchanging accept and non-accept states, word w may get accepted.

Summary:

- Regular expressions, NFAs, DFA.
- Reg. Exp $\longrightarrow$ NFA
- NFA $\longrightarrow$ DFA
- Complementing languages given by NFA/DFA.