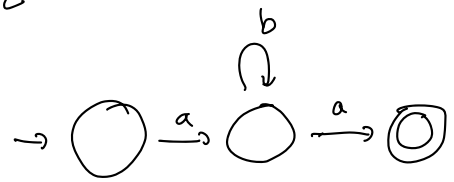
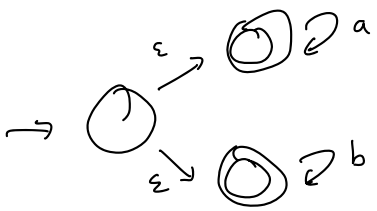
$\{ab^n a\}$



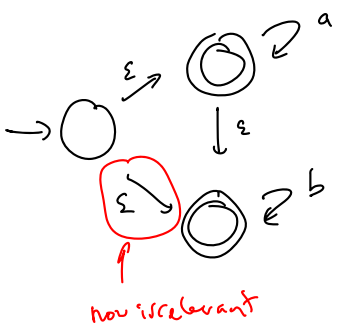
(an NFA)

An NFA is also allowed to make a spontaneous transition or "ε-transition"

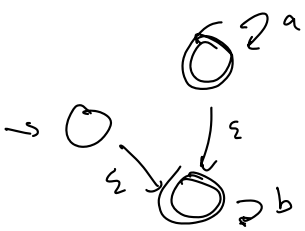
Where a transition happens without reading any letter.



Accepts any string like a^n or b^n.



Accepts $a^n b^m$



Formally, an NFA is same as a DFA

except:

- NFA allows ε-transitions.

$\delta(q, a)$
$\delta(q, b)$
$\delta(q, \varepsilon)$ ← ε is allowed here.

- NFA allows several arrows per letter,
  so $\delta(q, a)$ is a set of states.

In a DFA: $\delta : Q \times \Sigma \to Q$

in a NFA: $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to \mathcal{P}(Q)$

(↑ also allow ε; ↑ needs to be sets of states.)

It needs to be $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \longrightarrow \mathcal{P}(Q)$

δ takes a pair: (state, letter or ε),
the answer is a subset of Q.

Def An NFA is a 5-tuple:

$$M = (Q, \Sigma, \delta, q_0, F)$$

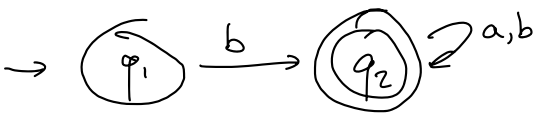
where:

Q is a finite set of states
Σ is a finite alphabet
$\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to \mathcal{P}(Q)$ is the trans. fcn.

("power set of Q" = set of all subsets.)

$q_0 \in Q$ is the start state

$F \subseteq Q$ is the set of accepting states.

Ex 

write the NFA formally

$$M = (\{q_1, q_2\}, \{a, b\}, \delta, q_1, \{q_2\})$$

where δ is:

$\delta(q_1, a) = \emptyset$
$\delta(q_1, b) = \{q_2\}$
$\delta(q_1, \varepsilon) = \emptyset$

$\delta(q_2, a) = \{q_2\}$
$\delta(q_2, b) = \{q_2\}$
$\delta(q_2, \varepsilon) = \emptyset$

The language accepted by a NFA?

For DFA: $L(M) = \{x \in \Sigma^* \mid \delta^*(q_0, x) \in F\}$

In a NFA $\delta^*(q, x)$ when x is a string,

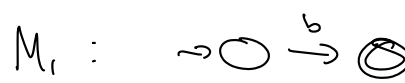
$\delta^*(q, x)$ is the set of all states reachable from q when we read x.

$L(M) = \{x \in \Sigma^* \mid \delta^*(q_1, x)$ contains an accepting state $\}$

Earlier:

$$L(M) = \{ x \in \Sigma^* \mid \delta^*(q_1, x) \cap F \neq \emptyset \}$$

NFAs can be combined using ε's.

M_1: 

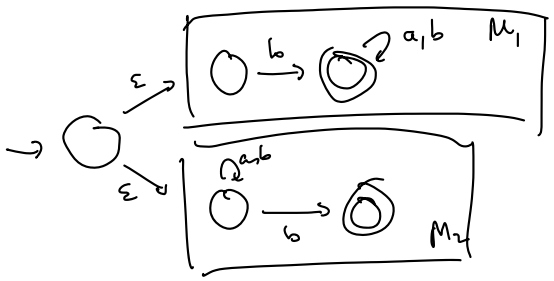
— accepts all strings starting with b

M_2: 

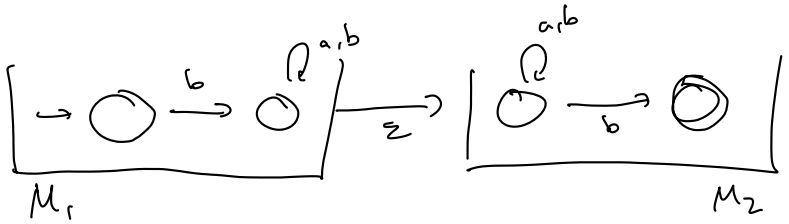
ending with b.

To get starting with b OR ending with b

we can combine like:



We can also do:



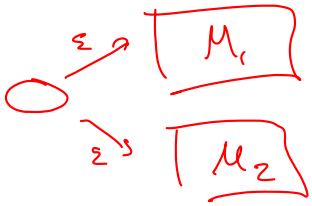
This is the concatenation machine.

it accepts strings like xy

where $x \in L(M_1)$ & $y \in L(M_2)$

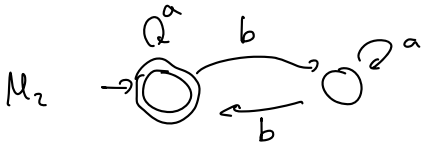
Make a NFA which accepts when length is even, or when # of b's is even.

union these

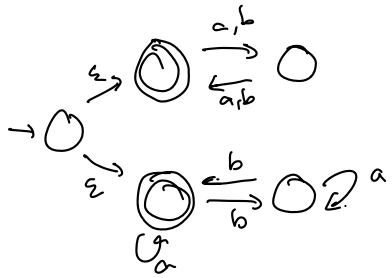


M_1: length is even:

M_2: # b's is even



My final machine:



ababa

bbb

DFAs can accept any regular lang.

Which languages can be accepted by NFA?

Any reg. lang can be computed by NFA.

Are there new langs that we can do on NFA but not DFA?