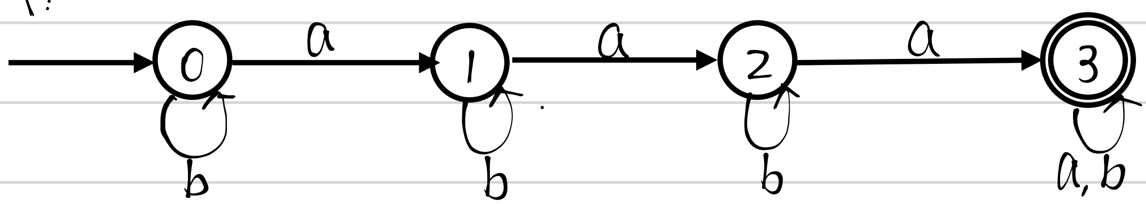
02-05 Lecture Introduction to DFAs

Deterministic Finite automaton:

Conceptually, a machine that takes in an input string $x \in \Sigma^*$, and consumes symbols in x one by one from left to right, and outputs a Yes (Accept) or No (Reject) answer.

M:



○ : states   a, b : symbols   ⟶ : transitions.

Formally, a DFA M is a structure consisting of 5 components.

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

Q : a finite set of states

Σ : the input alphabet
(a finite set of single symbols)

s : the start state (alway a single start state)

F : $F \subseteq Q$, a set of final/accept states

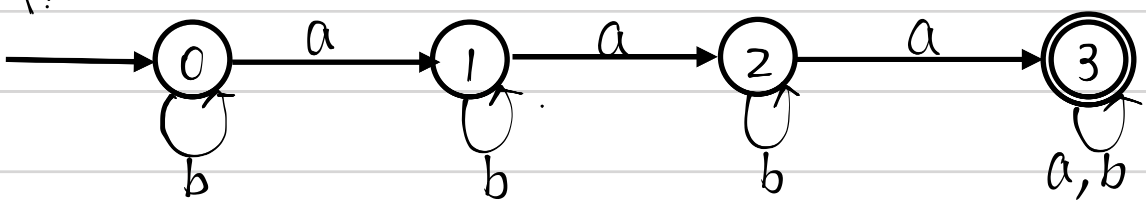
may have multiple final states

δ : $Q \times \Sigma \to Q$ transition function
maps an ordered pair to a state.
δ tells which state to move to in response to an input
$\delta(q, a) = r$
if M is currently in state q and the current input symbol to consume is a, then M consumes the symbol and moves to state r.

$|Q| \times |\Sigma|$ transitions: one transition out of each $q \in Q$ on each $a \in \Sigma$

$Q = \{0, 1, 2, 3\}$
$\Sigma = \{a, b\}$
$s = 0$
$F = \{3\}$

M:



$\delta(0,a)=1$, $\delta(1,a)=2$, $\delta(2,a)=3$, $\delta(3,a)=3$
$\delta(0,b)=0$, $\delta(1,b)=1$, $\delta(2,b)=2$, $\delta(3,b)=3$

Specify a DFA

① list all the components of the DFA

② use a table

write. $\rightarrow$ to specify the start state

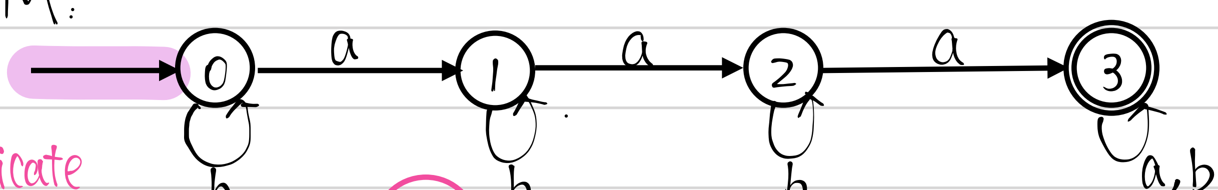
write F besides each final state.

	$Q \backslash \Sigma$	a	b
$\rightarrow$	0	1	0
	1	2	1
	2	3	2
F	3	3	3

$\delta(0,a)=1$

③ Use a Diagram. (For homeworks)

M:



indicate start state

double circle indicate final states

Operation mechanism.

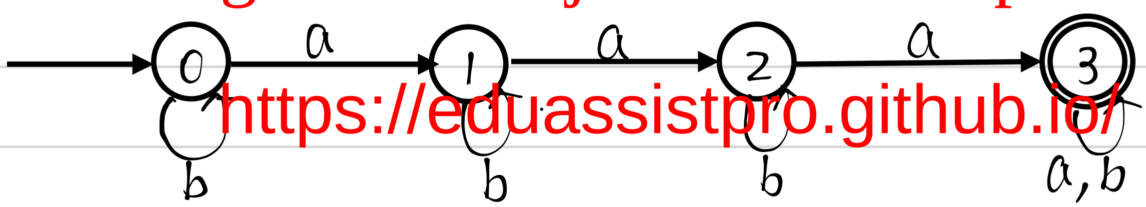
① An input can be any string $x \in \Sigma^*$

② Starting from the start state s, the DFA M consumes symbols in x one by one from left to right and moves to states according to the transition function δ

③ When the machine consumes the last symbol in x and lands on some state p,

- x is accepted (Yes) if $p \in F$
- x is rejected (No) if $p \notin F$

Example



input string baabb

$$\underset{\overset{\uparrow}{0}}{\text{baabb}} \xrightarrow[\delta(0,b)=0]{} \underset{\overset{\uparrow}{0}}{\text{aabb}} \xrightarrow[\delta(0,a)=1]{} \underset{\overset{\uparrow}{1}}{\text{abb}} \xrightarrow[\delta(1,a)=2]{} \underset{\overset{\uparrow}{2}}{\text{bb}}$$

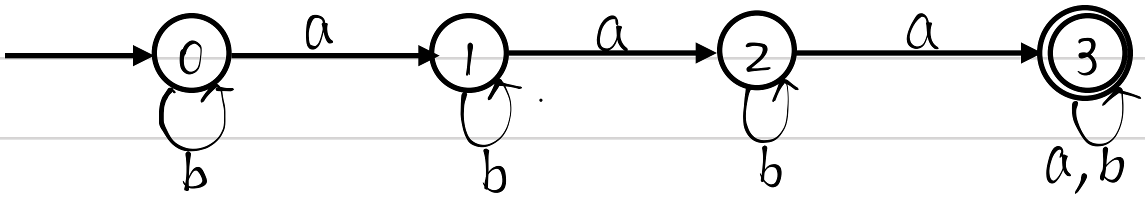
$$\xrightarrow[\delta(2,b)=2]{} \underset{\overset{\uparrow}{2}}{\text{b}} \xrightarrow[\delta(2,b)=2]{} \overset{\uparrow}{2}$$

end of the string

lands on state 2

$2 \notin F$

baabb is rejected (No)



input string aaab

aaab $\underset{\delta(0,a)=1}{\longrightarrow}$ aab $\underset{\delta(1,a)}{\longrightarrow}$ ab $\underset{\delta(2,a)=3}{\longrightarrow}$ b

$\uparrow$ 0 $\uparrow$ 1 $\uparrow$ 2 $\uparrow$ 3

$\underset{\delta(3.b)=3}{\longrightarrow}$ $\uparrow$ 3

end of the string
lands on state 3

$3 \in F$

aaab is accepted (Yes)

Formally, we define a function $\hat{\delta} : Q \times \Sigma^* \to Q$ to specify whether an input string is accepted or rejected by the DFA.

- $\hat{\delta}$ maps a state q and a string x to a new state $\hat{\delta}(q, x)$
- Intuitively $\hat{\delta}$ is the multistep version of δ

$$\hat{\delta}(q, \varepsilon) = q$$

$\leftarrow$ stay put on input ε.

$$\hat{\delta}(q, ya) = \delta(\hat{\delta}(q, y), a)$$

the state M ends up in when started in state q and consuming all symbols in y according to the transition function δ

Def. A string x is accepted by the DFA M if $\hat{\delta}(s, x) \in F$
and is rejected - - - - - - - - if $\hat{\delta}(s, x) \notin F$

s is the start state of M, F is the set of final states of M

Def. The language accepted (recognized) by M is the set of all strings accepted by M, denoted as $L(M)$

$$L(M) = \{ x \in \Sigma^* \mid \hat{\delta}(s, x) \in F \}$$

Def. A language $A \subseteq \Sigma^*$ is a regular language if $A = L(M)$ for some DFA M.

Meaning there exists a DFA M that satisfies the two conditions.

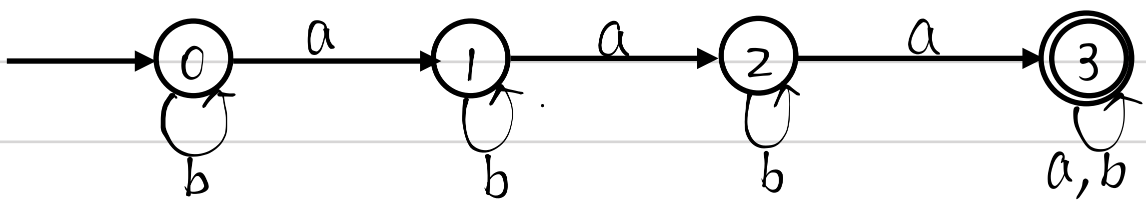
① $\forall x \in A$, M accepts x and

② $\forall x \notin A$, M rejects x.

Example

$A = \{ x \in \{a, b\}^* \mid x \text{ contains at least three } a\text{'s} \}$

M:



A is regular because there exists a DFA M that accepts it.

$A = L(M)$

Example:

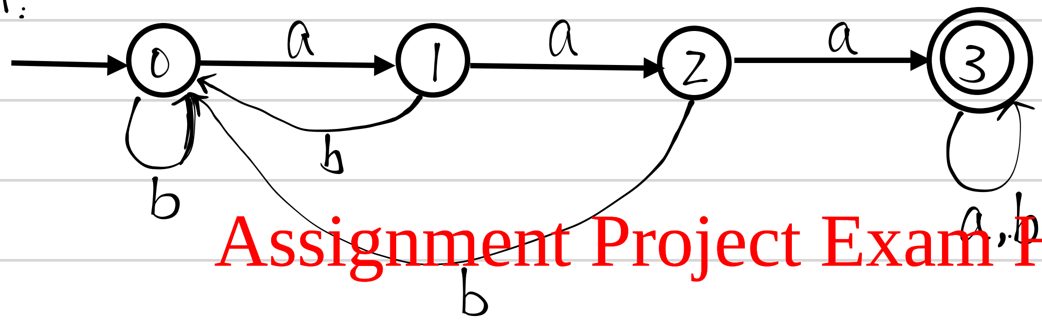
$A = \{ xaaay \mid x, y \in \{a,b\}^* \}$

$= \{ z \in \{a,b\}^* \mid z$ contains a substring of thre consecutive a's $\}$

baab $\notin A$, babaaaab $\in A$

A is regular because there exists a DFA M) that accepts it.

M:



the idea behind the design of M is

use the states to count the number of consecutive a's the machine have read (consumed) so far.

If M hasn't seen three a's in a row and sees a b, M goes back to the start state (recounting)

If M has seen three a's in a row, M stays in 3 no matter what symbol it sees (consumes) thereafter.