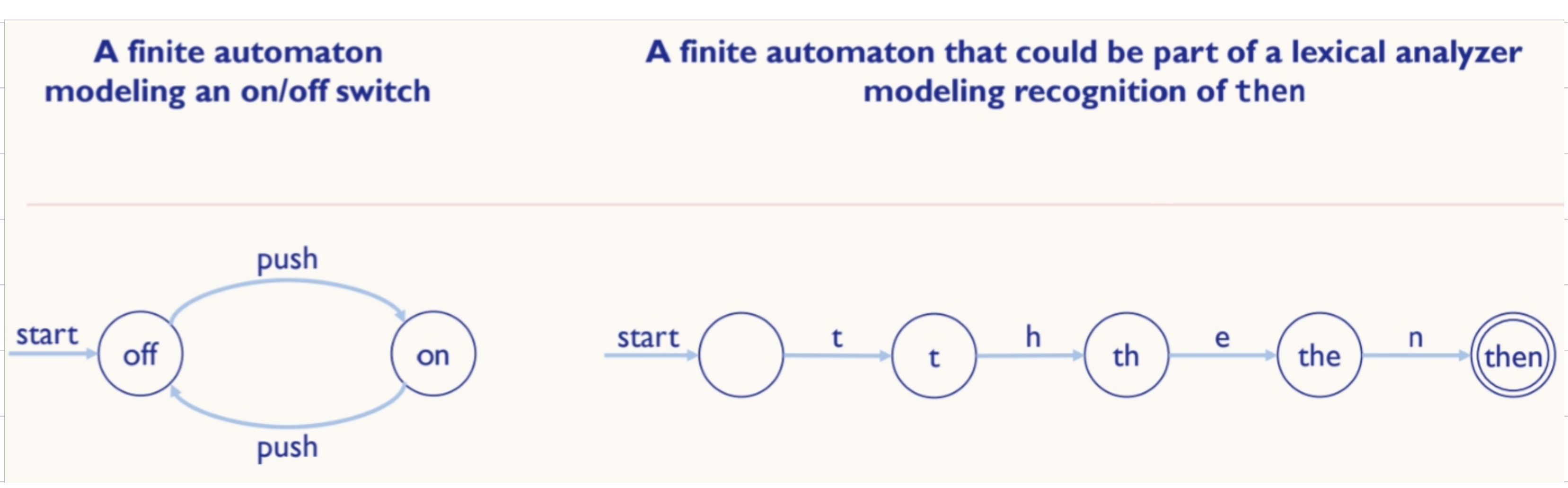
Week 1

- Automata = Belajar abstract computing device / mesin.
  - ↳ Penting untuk study of the limits of computation
  - ↳ Computer bs buat apa aja, bs buat se-efisien apa.

- Hal yang membuat mesin compute mechanically

binary string akhiran "0"   all valid Java codes (Compilers)

- Finite Automata = Useful model for many important kinds of hardware & software.
  - ↳ Software for design & check behavior of digital circuits
  - ↳ The "lexical analyzer" of a typical compiler
    - ↳ The compiler component that breaks input text into logical units, seperti identifiers, keywords, & punctuation
  - ↳ Software for scan byk text, seperti koleksi web pages, to find occurences of word, phrase, or other pattern.
  - ↳ Software for verifying systems of all types that have a finite number of distinct states, such as protokol komunikasi / protokol utk secure exchange of info.

-


Central concepts of automata theory :

- Symbols : 0, 1, 2, x, y, z
  - Valid untuk bahasa {a, b, ...} | Nonvalid untuk bahasa {é, ä, =, +, ...}

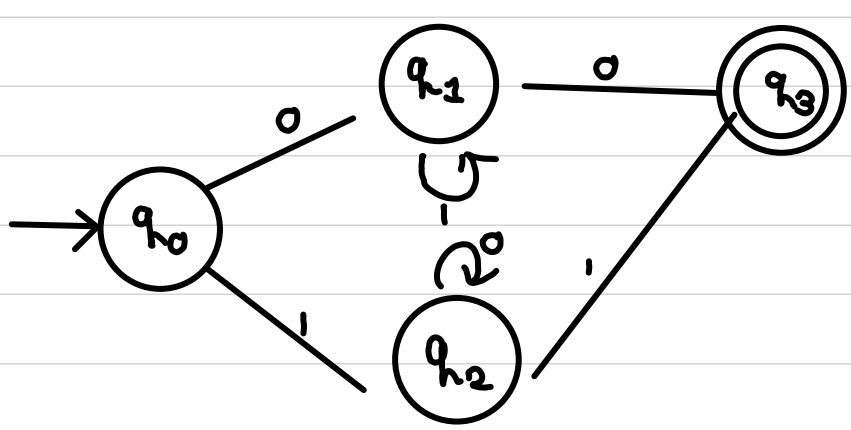
- Alphabets (Σ) : Finite, nonempty set of symbols
  - $\Sigma = \{0, 1\} \rightarrow$ Simbol = 0 dan 1

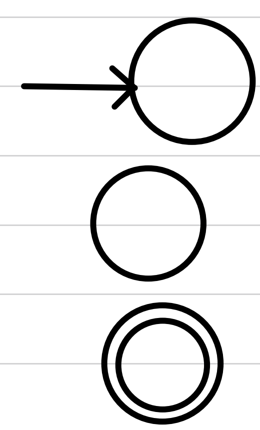
- Strings : Finite sequence of symbols chosen from some alphabet
  - $\Sigma = \{0, 1\} \rightarrow$ String = 001, 1010, ...
  - The empty string (ϵ) = String with 0 occurrence of symbols
  - Length of a string = $|w|$ = Number of positions for symbols in the string
    - $|011| = 3$
    - $|\epsilon| = 0$
  - Powers of $\Sigma = \Sigma^{|w|}$, dengan $|w| = H$ = Length of string
    - $\Sigma^0 = \{\epsilon\} = \{\} = \phi$
    - $\Sigma^1 = \{0, 1\}$
    - $\Sigma^2 = \{00, 11, 01, 10\}$
    - $\Sigma^3 = \{000, 001, 010, 011, 100, 101, 110, 111\}$
  - Cardinality = Numbers of elements in a set
    - $\Sigma^{|w|} \rightarrow$ Cardinality $= 2^{|w|}$
  - $\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup ....$ = Star closure
    - $\Sigma = \{0, 1\} \rightarrow \Sigma^* = \{\epsilon\} \cup \{0, 1\} \cup \{00, 01, 10, 11\} \cup ....$
  - $\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup ....$ = Positive closure
  - $\Sigma^* = \Sigma^+ \cup \{\epsilon\}$
  - Concatenation of strings
    - x is string composed of i symbols $\Rightarrow x = a_1 a_2 ... a_i$
    - y is string composed of j symbols $\Rightarrow y = b_1 b_2 .... b_j$
    - xy is string of length i+j $\Rightarrow xy = a_1 a_2 .... a_i b_1 b_2 .... b_j$
    - $xy \neq yx$
    - $\epsilon w = w\epsilon = w$

Closure = Pengulangan simbol

- Languages = Set of strings all of which are chosen from some Σ^*
  - ↳ $\Sigma = \{0,1\} \rightarrow L_1$ = Set of all strings of length 2 } finite
    - L_2 = Set of all strings of length 3 }
    - L_3 = Set of all strings that begin with 0 } infinite
  - ↳ The empty language = ϕ = Has no strings
  - ↳ Language consisting of only empty string = $\{\epsilon\}$ = Has 1 string
  - ↳ Set formers as a way to define languages.
    - ↳ {w| something about w} = {w| w is a binary integer that is prime}
      $= \{0^n 1^n \mid n \geq 1\}$

Diagram Transisi dari Suatu Mesin Otomata



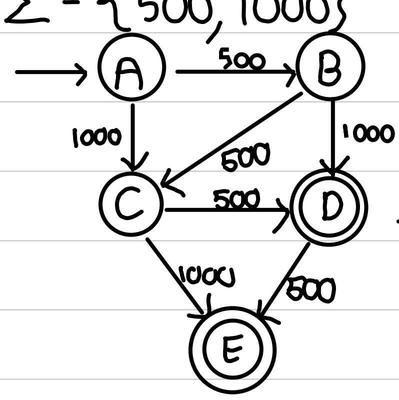


- : State awal
- : State transisi
- : State akhir

- Q = Himpunan state pembentuk otomata

Contoh Soal:

Buat mesin otomata yang mensimulasikan self-vending machine yang menjual pepsi dan fanta seharga 1500 & 2000. Mesin menerima uang 500 & 1000.

- $\Sigma = \{500, 1000\}$
- 

- $q_0 = \{A\}$
- $^*q = F = \{D, E\}$
- $Q = \{A, B, C, D, E\}$

→ DFA (Deterministic Finite Automata)

- $L = \{\underbrace{500.1000}_{w_1}, \underbrace{1000.500.500}_{w_2}, \ldots\}$
- $|w_1| = 2$
- $|w_2| = 3$

- Selain DFA, ada NFA (Nondeterministic Finite Automata)
  PDA (Push Down Automata)

- 2 operasi di TBA :

1) Concatenation : $A \circ B = \{xy \mid x \in A \text{ and } y \in B\}$
   - ↳ $P = \{0, 1\}$, $Q = \{01, 10, 1\}$

     $L = P.Q$ = Himpunan string yang dibentuk dari 1 anggota P disambung dengan 1 anggota Q.

     $P.Q \neq Q.P$

     $P.Q = \{001, 010, 01, 101, 110, 11\}$

2) Union : $A \cup B = \{x \mid x \in A \text{ or } x \in B\}$
   - ↳ $P = \{0, 1\}$, $Q = \{01, 10, 1\}$

     $L = P \cup Q$ = Penggabungan seluruh anggota P dan Q, tapi kalau ada yang sama cukup tulis 1x saja.

     $P \cup Q = \{0, 1, 01, 10\}$

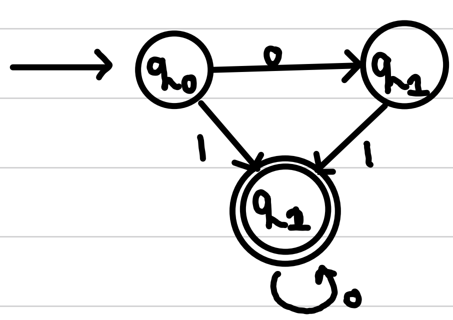
3) Star : $A^* = \{x_1 x_2 x_3 \ldots x_k \mid k \geq 0 \text{ and each } x_i \in A\}$
   - ↳ $P = \{0, 1\}$, $Q = \{01, 10, 1\}$

     $P^* = \{\epsilon, 0, 1, 00, 01, 10, 11, \ldots\}$

- $RE = (0 + 1).1$

  ↳ Tanda OR

  $L(RE) = \{\epsilon.1, 01, 11, 001, 011, 101, 111\}$

- →  → Bahasa automata yang terdiri dari himpunan string berakhiran 0 atau 1.

Week 2

Finite State Automata

- Simplest model of computation
- Has a very limited memory
- e.g.: - Text editor
  - Elevator / Lift control mechanism
  - The classic crossing river puzzle

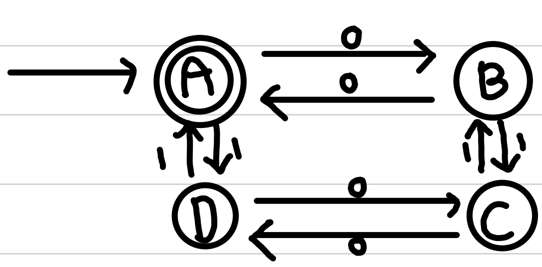
Deterministic Finite Automata

Each input there is one and only one state to which the automation can transition from its current state

- Five-tuple notation: $M = (Q, \Sigma, \delta, q_0, F)$
  - ↳ Q = Set of states
  - ↳ Σ = Set of input symbols / Alphabet
  - ↳ δ = Transiton function from $Q \times \Sigma \rightarrow Q$
  - ↳ q_0 = Start state
  - ↳ F = Final / Accepting state

- **Contoh Soal:**

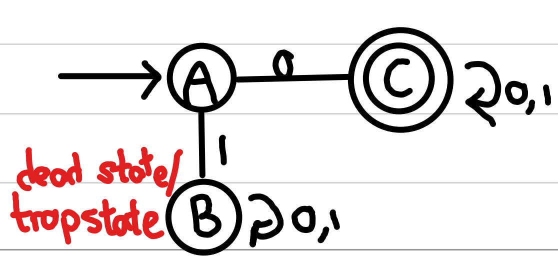
Design a DFA to accept the language $L = \{ w \mid w$ has both an even number of 0's and an even number of 1's$\}$



$M = (\{A, B, C, D\}, \{0, 1\}, \delta, A, \{A\})$

- **Contoh Soal:**

Construct a DFA that accepts any strings over $\{0, 1\}$ that start with '0'

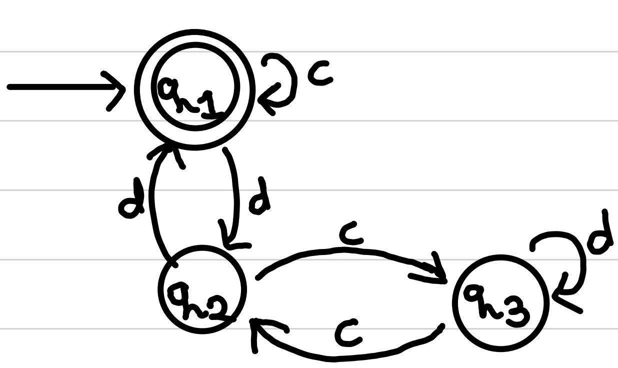


1100 } accepted/ rejected
01011

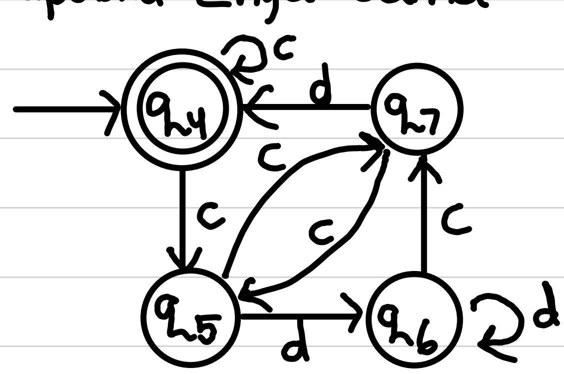
- $1100 \to \delta(A, 1100) = \delta(B, 100) = \delta(B, 00) = \delta(B, 0) = B$ (rejected)
- $01011 \to \delta(A, 01011) = \delta(C, 1011) = \delta(C, 011) = \delta(C, 11) = \delta(C, 1) = C$ (accepted)

Language of DFA

- The language of DFA $M = (Q, \Sigma, \delta, q_0, F)$ denoted $L(M)$
- $L(M) = \{w \mid \delta(q_0, w) \text{ is in } F\}$
  - ↳ The language M is set of strings w that takes start at q_0 to one of F.
- If L is a L(M) for some DFA M, then L is regular languages.
- Language adalah regular language jika dan hanya jika some FSM recognizes it.
- Language NOT Regular
  - Not recognized by any FSM
  - Require memory
    - ↳ Memory FSM limited
    - ↳ Gbs $\underbrace{\text{store}}_{\text{abababa}}$ / $\underbrace{\text{count}}_{a^n b^n,\ aabb}$ strings

- 2 DFA dikatakan sama apabila Lnya sama
- 

A



B

States	c	d
(q_1, q_4)	(q_1, q_4) F F	(q_2, q_5) I I
(q_2, q_5)	(q_3, q_7) I I	(q_1, q_6) F I

beda

A & B not equivalent

$$
\begin{tikzcd}
RE \arrow[r, dashed] \arrow[dr] & NFA \arrow[r] & \underline{\underline{DFA}} \arrow[ll, bend right] \arrow[r] & DFA \text{ minimal} \\
& \varepsilon\text{-NFA} \arrow[u] & &
\end{tikzcd}
$$

- **Finite State Automata / FSA :**

1) **Deterministic Finite Automata / DFA**

   Merupakan sebuah FSA yang setiap simpul pembentuknya menerima input dan mengantarkannya ke hanya 1 simpul tujuan.

2) **Non-deterministic Finite Automata / NFA**

   Merupakan sebuah FSA yang memiliki setidaknya 1 simpul menerima input tapi bisa mengantarkannya ke simpul tujuan yang berbeda.

- Ada 3 tipe penelusuran string apakah bahasanya dari suatu DFA yaitu :

1) **String 00**

   ↳ $\delta(q_0, 00) = \delta(\delta'(q_0, 0), 0)$
   $= \delta(q_1, 0)$
   $= q_1 \rightarrow$ 00 bukan bahasanya karena tidak berakhir di final state.

2) **String 01**

   ↳ $\delta(q_0, 01) = \delta(\delta'(q_0, 0), 1)$
   $= \delta(q_1, 1)$
   $= q_3 \rightarrow$ 01 adlh bahasanya karena string habis terbaca DAN berakhir di final state

3) String 100

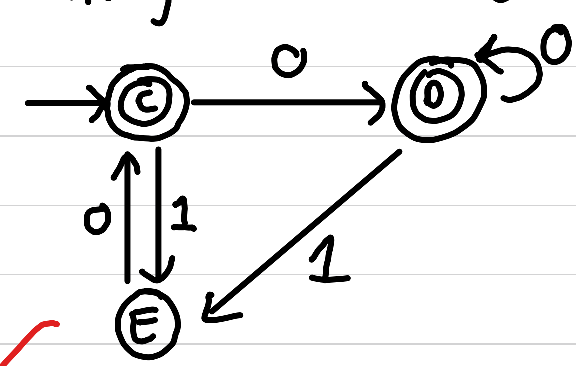
↳ $\delta(q_0, 100) = \delta(\delta'(q_0,1),00)$

$= \delta(q_2,00)$

$= \delta(\delta'(q_2,0),0)$

$= \delta(q_3,0) \rightarrow$ 100 bukan bahasanya karena string tidak habis terbaca

o) Meminimalisir / Menyederhanakan DFA yang statenya banyak menjadi lebih sedikit.

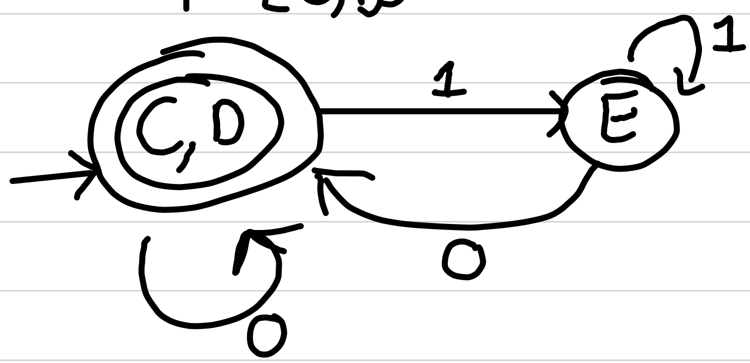


Harus sejenis

↳ First state dgn first state

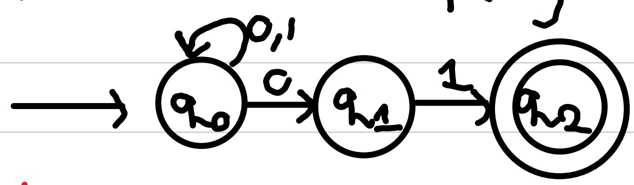
↳ Final state dgn final state

nfs	fs	
E	C,D	
$\delta(E,0) = C$	$\delta(C,0) = D$,	$\delta(C,1) = E$
$\delta(E,1) = E$	$\delta(D,0) = D$,	$\delta(D,1) = E$
	C & D sama	
	$\{C,D\}$	

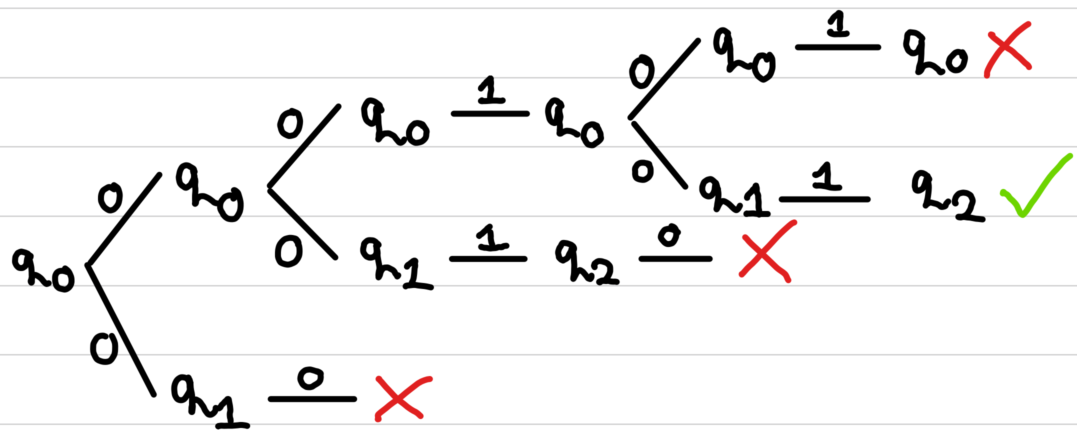


Week 3

- An NFA accepting all strings that end in 01

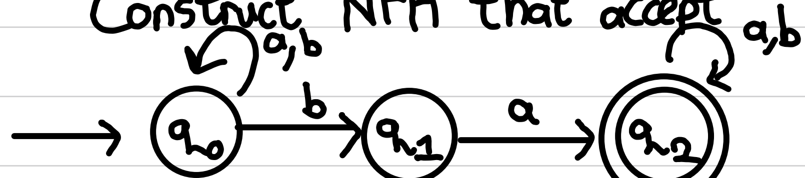


↳ The states NFA during input process of 00101



- Contoh Soal :

Construct NFA that accept sets of all strings over {a,b} that contain 'ba'



↳ Penelusuran string 'ba' : 'ba' $= \delta(q_0, ba)$

$= \delta(\{q_0, q_1\}, a)$

$= \delta(q_0, a) \cup \delta(q_1, a)$

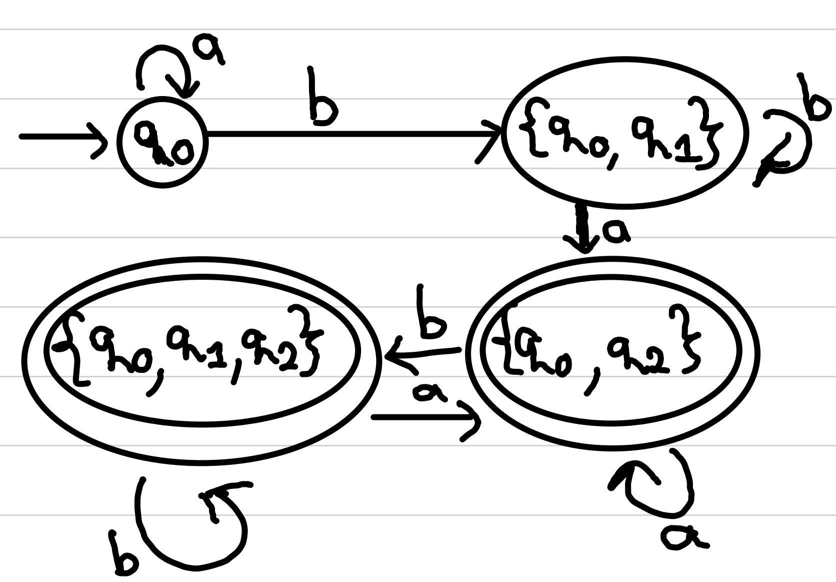
$= q_0 \cup q_2$

$= \{q_0, q_2\}$ hrs muncul di DFA ekivalen

Change to DFA

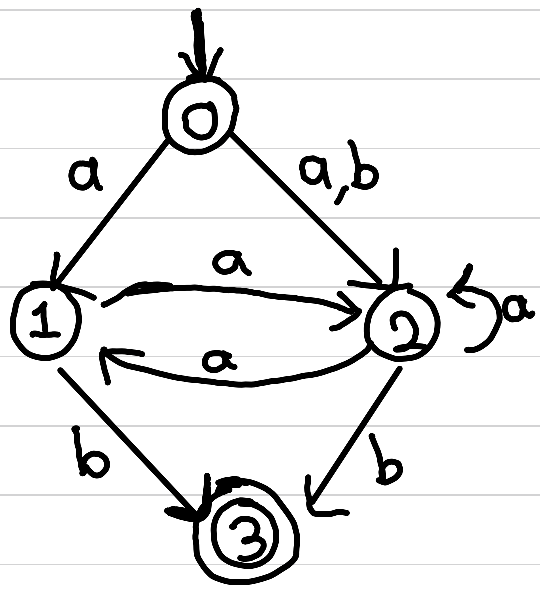
State	a	b
$\rightarrow q_0$	q_0	$\{q_0, q_1\}$
$\{q_0, q_1\}$	$\{q_0, q_2\}$	$\{q_0, q_1\}$
$\{q_0, q_2\}$	$\{q_0, q_2\}$	$\{q_0, q_1, q_2\}$
$*\{q_0, q_1, q_2\}$	$\{q_0, q_2\}$	$\{q_0, q_1, q_2\}$

→

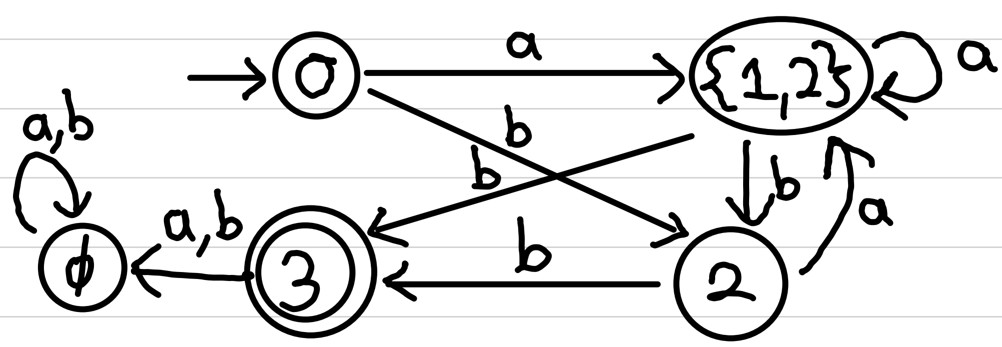


Contoh Soal :

Convert to DFA ekivalen from this NFA.

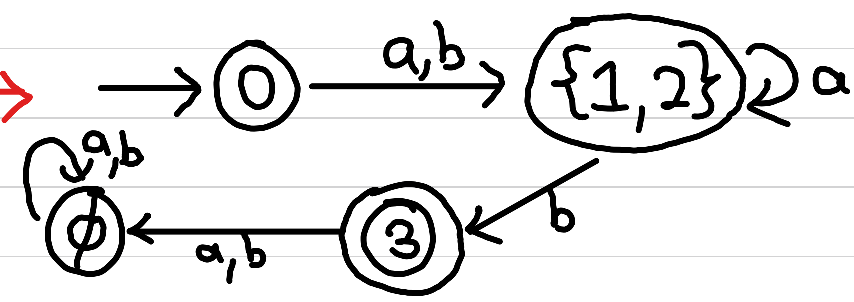


States	a	b
$\rightarrow 0$	$\{1,2\}$	$\{2\}$
$\{1,2\}$	$\{1,2\}$	$\{3\}$
$\{2\}$	$\{1,2\}$	$\{3\}$
$*\{3\}$	ϕ	ϕ



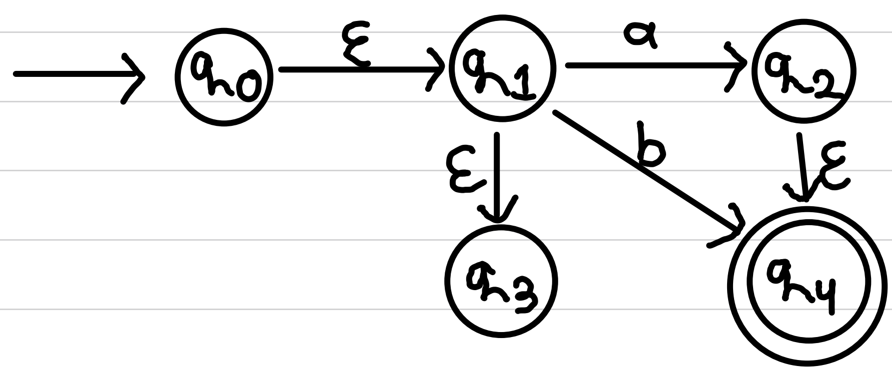
Untuk cari penyederhanaan

nfs	fs
$0, \{1,2\}, 2$	3
$\delta(0,a)=\{1,2\}$ $\delta(\{1,2\},a)=\{1,2\}$ $\delta(2,a)=\{1,2\}$ $\delta(0,b)=\{2\}$ $\delta(\{1,2\},b)=3$ $\delta(2,b)=2$	$\delta(3,a)=\phi$ $\delta(3,b)=\phi$



$\underbrace{\{1,2\} \;\&\; 2}_{\text{Dibuat dalam 1 simpul}}$ adlh sama / Indistinguishable

Week 4



- ε = The empty string
  ↳ An NFA is allowed to make a transition spontaneously, without receiving an input symbol

- δ = A transition from $Q \times \Sigma \cup \{\varepsilon\} \to 2^Q$
- Every state on ε goes to itself

- Epsilon closure(q_i) = Himpunan state-state yang bisa dicapai dari q_i ke state-state lain dengan input ε tanpa terputus, dan q_i termasuk dalam himpunan tersebut juga.

$\varepsilon\text{-}cl(q0) = \{q_0, q_1, q_3\}$
$\varepsilon\text{-}cl(q1) = \{q_1, q_3\}$
$\varepsilon\text{-}cl(q2) = \{q_2, q_4\}$
$\varepsilon\text{-}cl(q3) = \{q_3\}$
$\varepsilon\text{-}cl(q4) = \{q_4\}$

- Conversion of ε-NFA to NFA :

- $\delta'(\text{state}, \text{input}) = \varepsilon\text{-}cl(\delta(\varepsilon\text{-}cl(\text{state}), \text{input}))$

- $F' = F \cup \{q \mid (\varepsilon\text{-}cl(q) \cap F) \neq \phi\}$

-

δ	a	b	ε
$\rightarrow q_0$	ϕ	ϕ	q_1
q_1	q_2	q_4	q_3
q_2	ϕ	ϕ	q_4
q_3	ϕ	ϕ	ϕ
$*q_4$	ϕ	ϕ	ϕ

↓

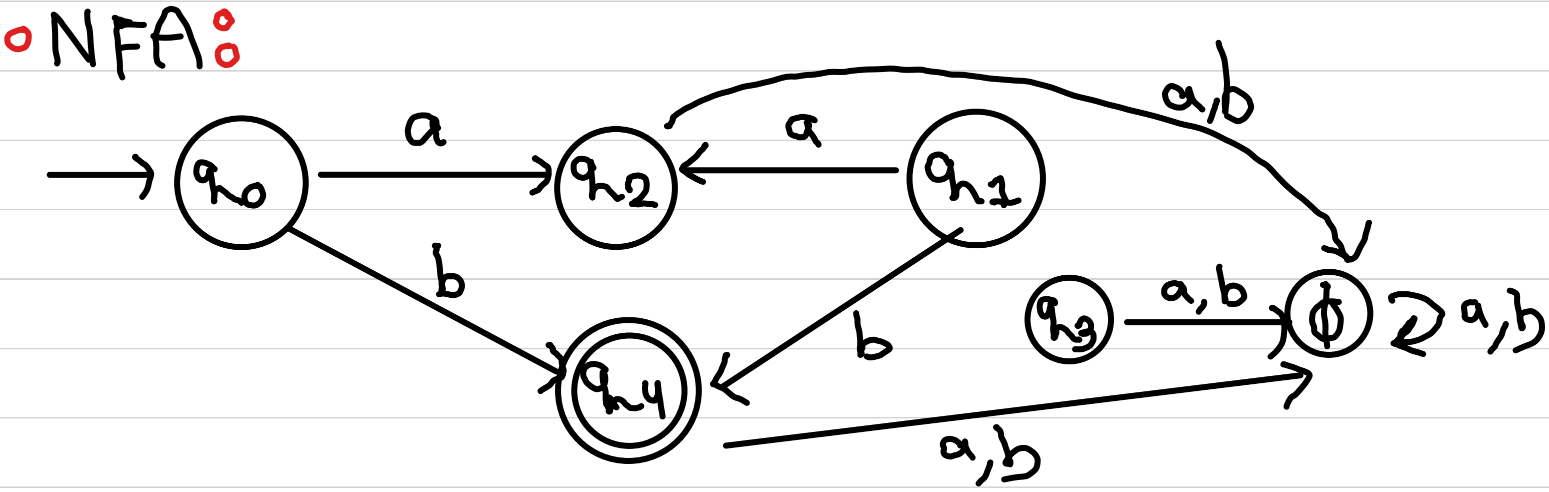
∘ $\varepsilon(\varepsilon\text{-}cl(\varepsilon\text{-}cl(q_i), a))$

δ	$\varepsilon\text{-}cl(q_i)$	δ, input a	$\varepsilon\text{-}cl(\delta)$
$\rightarrow q_0$	$\{q_0,$ $q_1,$ $q_3\}$	ϕ q_2 ϕ	ϕ $\{q_2, q_4\}$ ϕ
q_1	$\{q_1,$ $q_2\}$	q_2 ϕ	$\{q_2, q_4\}$ ϕ
q_2	$\{q_2,$ $q_4\}$	ϕ ϕ	ϕ ϕ
q_3	q_3	ϕ	ϕ
$*q_4$	q_4	ϕ	ϕ

δ	$\varepsilon\text{-}cl(q_i)$	δ, input b	$\varepsilon\text{-}cl(\delta)$
$\rightarrow q_0$	$\{q_0,$ $q_1,$ $q_3\}$	ϕ q_4 ϕ	ϕ q_4 ϕ
q_1	$\{q_1,$ $q_2\}$	q_4 ϕ	q_4 ϕ
q_2	$\{q_2,$ $q_4\}$	ϕ ϕ	ϕ ϕ
q_3	q_3	ϕ	ϕ
$*q_4$	q_4	ϕ	ϕ

Perhatikan δ & $\varepsilon\text{-}cl(\delta)$

∘ NFA:



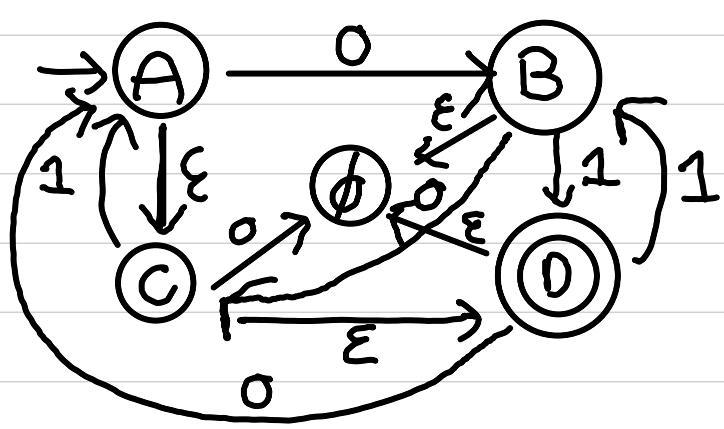
o) Contoh Soal :

Diberikan Tabel Fungsi Transisi ε-NFA

δ	0	1	ε
→A	B	∅	C
B	C	D	∅
C	∅	A	D
*D	A	B	∅

a. Gambar ε-NFA
b. ε-cl
c. NFA

a.

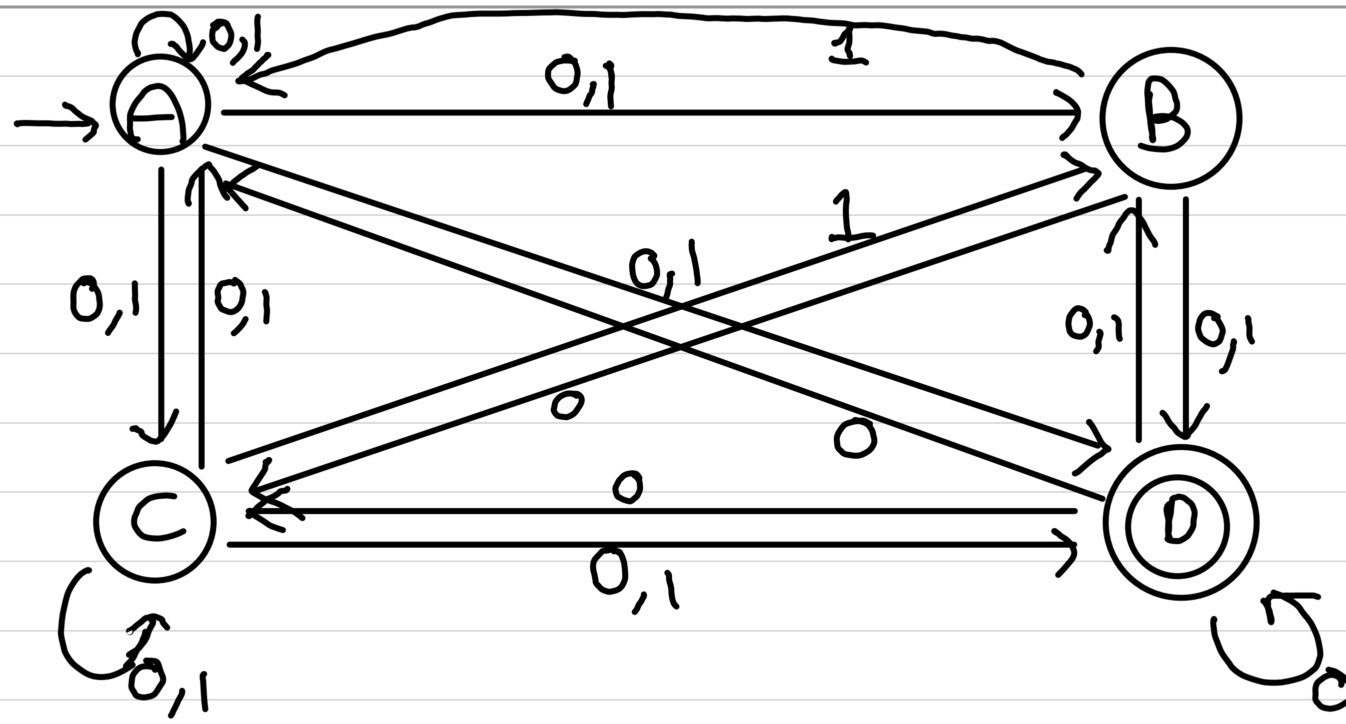


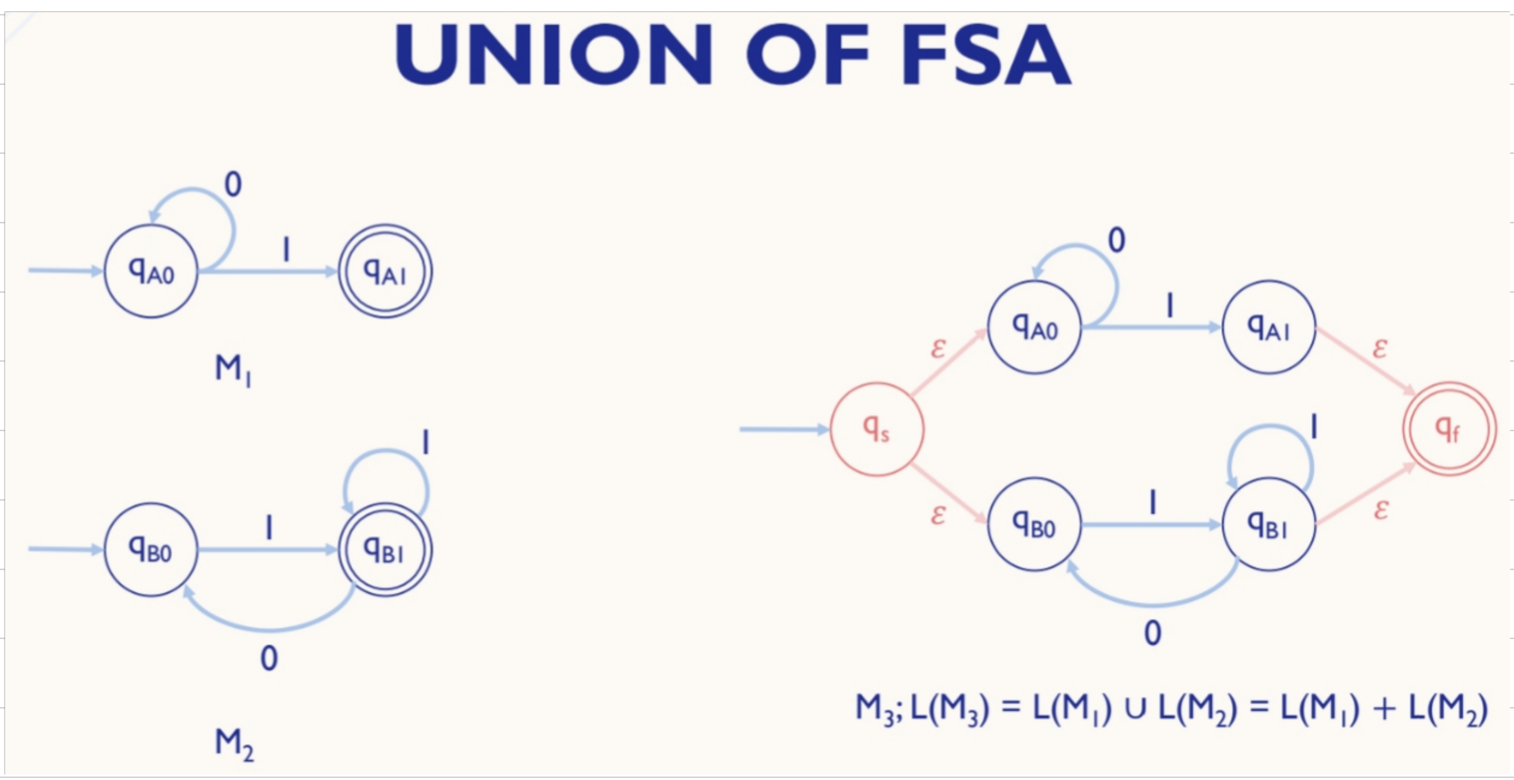
b. ε-cl(A) = {A, C, D}
ε-cl(B) = {B}
ε-cl(C) = {C, D}
ε-cl(D) = {A, D}

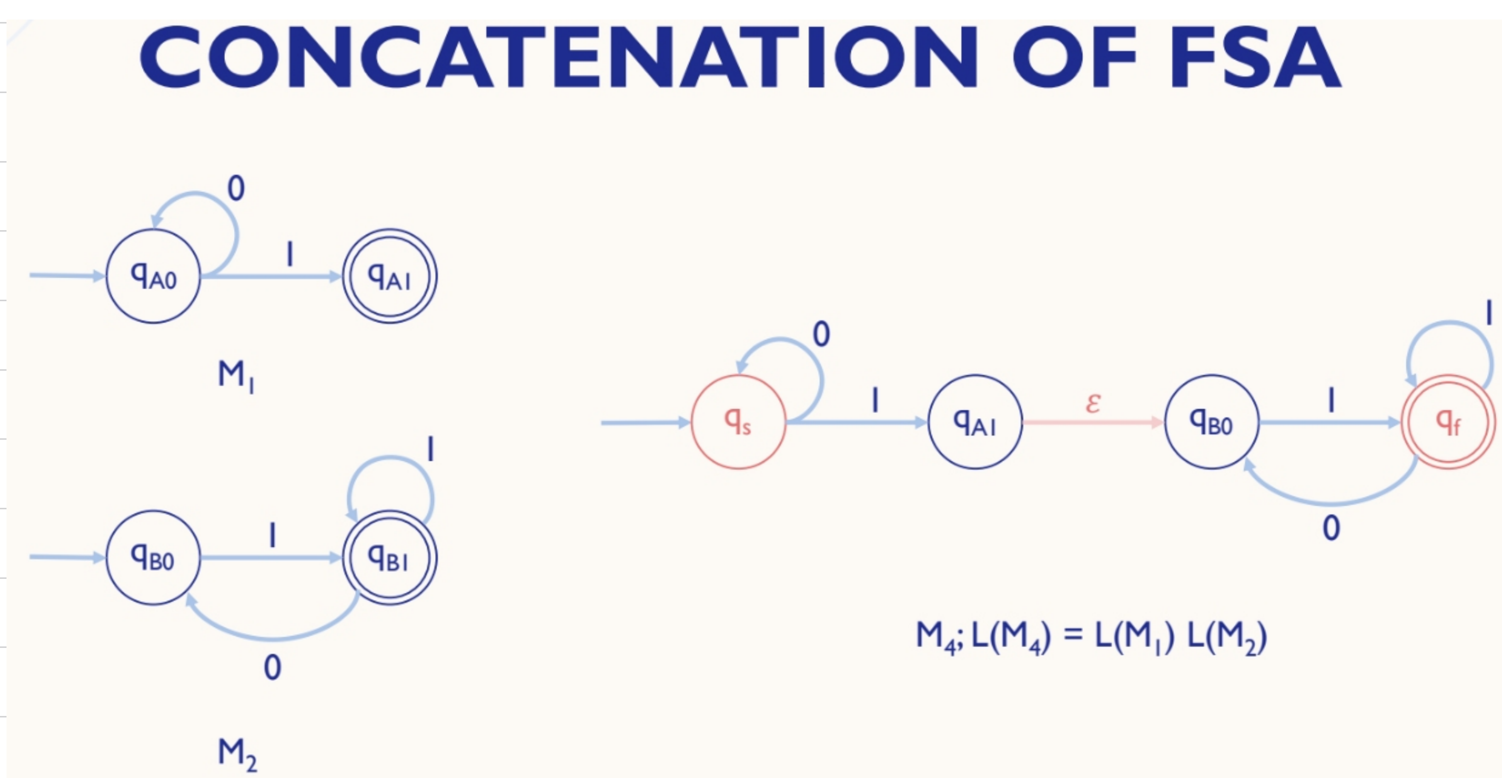
c.

δ	ε-cl(q_i)	δ, input 0	ε-cl(S)
→A	{A, C, D}	B ∅ A	B ∅ {A, C, D}
B	B	C	{C, D}
C	{C, D}	∅ A	∅ {A, C, D}
*D	{A, D}	B A	B {A, C, D}

δ	ε-cl(q_i)	δ, input 1	ε-cl(S)
→A	{A, C, D}	∅ A B	∅ {A, C, D} B
B	B	D	{A, D}
C	{C, D}	A B	{A, C, D} B
*D	{A, D}	∅ B	∅ B


A
B
C
D
0,1
1
0


UNION OF FSA
M₁
M₂
ε
M_3; $L(M_3) = L(M_1) \cup L(M_2) = L(M_1) + L(M_2)$


CONCATENATION OF FSA
M₁
M₂
ε
M_4; $L(M_4) = L(M_1)\, L(M_2)$

Week 5

- 2 states p and q are said to be EQUIVALENT or INDISTINGUISHABLE if :
  - $\delta(p, w) \in F$ and $\delta(q, w) \in F$
  - $\delta(p, w) \notin F$ and $\delta(q, w) \notin F$

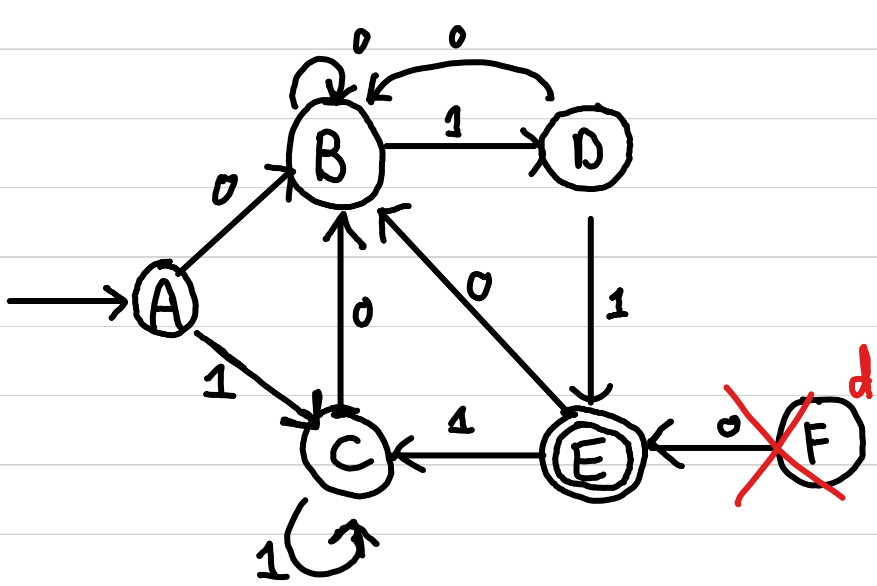
  - If $|w| = 0$, then p and q are said to be 0 equivalent
  - If $|w| = 1$, then p and q are said to be 1 equivalent
  - If $|w| = 2$, then p and q are said to be 2 equivalent
  - If $|w| = n$, then p and q are said to be n equivalent

  - Kalau p & q equivalent, q & r equivalent, maka p & r juga equivalent.

- 2 states p and q are said to be DISTINGUISHABLE if :
  - $\delta(p, w) \notin F$ and $\delta(q, w) \in F$

- Contoh Soal :

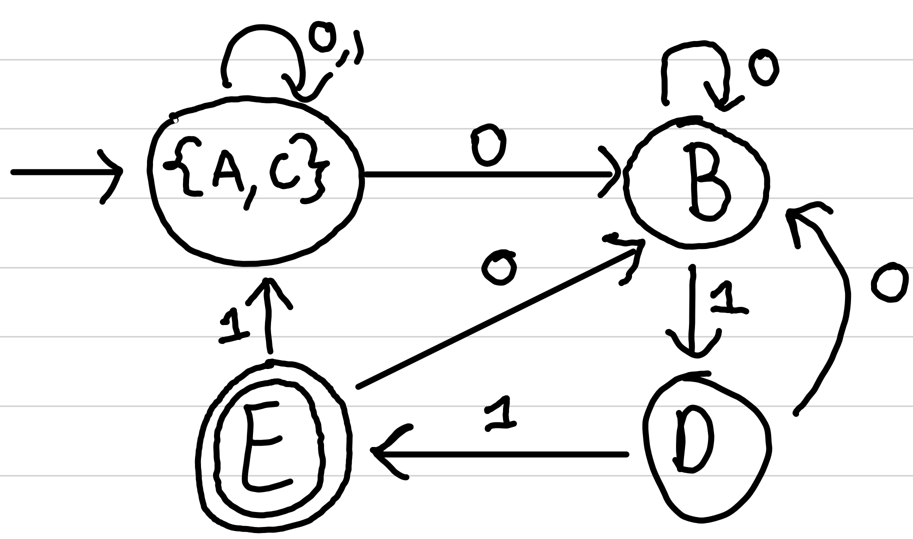
Minimize the following DFA, using partitioning method



dibuang krn tdk mulai dr start state dan lgsg ke final state

δ	0	1
→A	B	C
B	B	D
C	B	C
D	B	E
*E	B	C

	nfs	fs
0 equivalence	{A, B, C, D}	{E}
1 equivalence	{A, B, C} {D}	{E}
2 equivalence	{A, C} {B} {D}	{E}



- Always eliminate any state that cannot be reached from the start state.

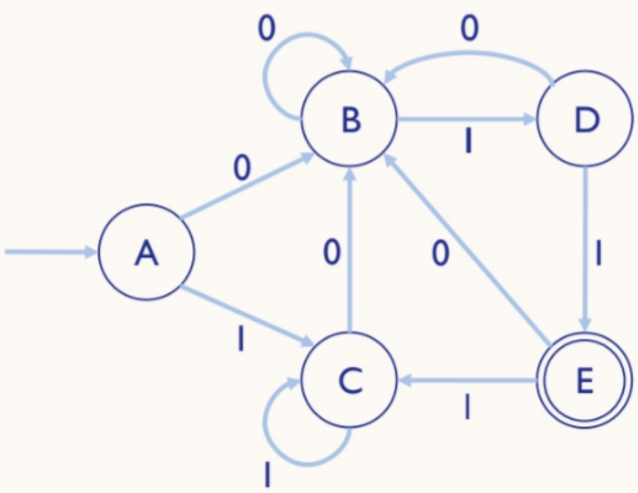
Table Filling Method / Myhill Nerode Theorem

1) Draw a table for all pairs of states (p, q)
2) Mark all pairs where $p \in F$ and $q \notin F$
3) If there are any unmarked pairs (p, q) such that $[\delta(p, w), \delta(q, w)]$ is marked, then mark $[p, q]$; where w is an input symbol
   ↳ REPEAT this until no more markings can be made
4) Combine all the unmarked pairs and make them a single state in the minimized DFA.

TABLE FILLING METHOD

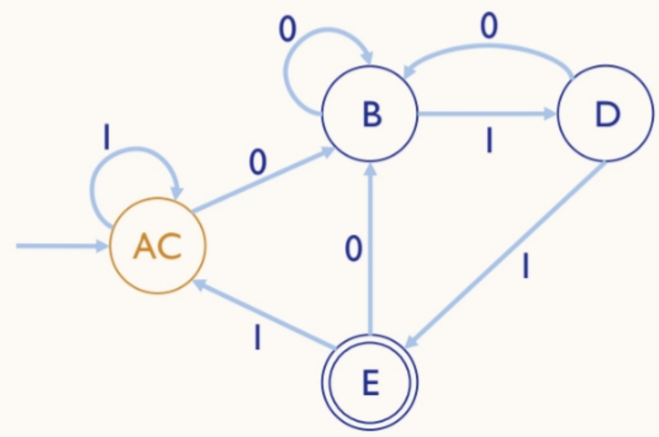
(MYHILL NERODE THEOREM)

Minimize the following DFA.

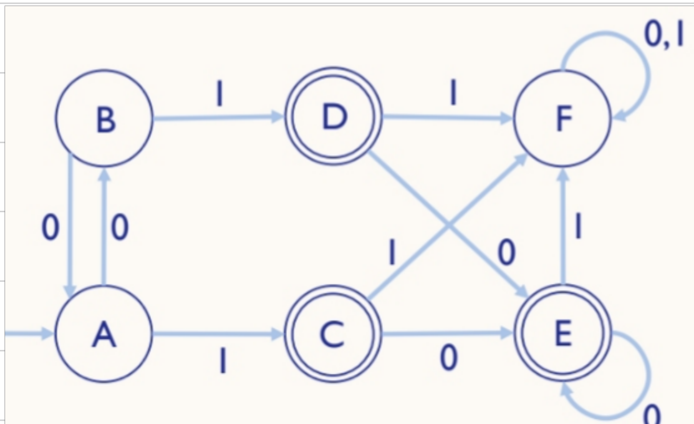


States	0	1
(B, A)	(B, B)	(D, C)
(C, A)	(B, B)	(C, C)
(C, B)	(B, B)	(C, D)
(D, A)	(B, B)	**(E, C)**
(D, B)	(B, B)	**(E, D)**
(D, C)	(B, B)	**(E, C)**
(B, A)	(B, B)	**(D, C)**
(C, A)	(B, B)	(C, C)
(C, B)	(B, B)	**(C, D)**
(C, A)	(B, B)	(C, C)

	A	B	C	D	E
A					
B	✓				
C		✓			
D	✓	✓	✓		
E	✓	✓	✓	✓	



Unmarked:
(C, A)



	A	B	C	D	E	F
A						
B						
C	✓	✓				
D	✓	✓				
E	✓	✓				
F	✓	✓	✓	✓	✓	

States	0	1
(B, A)	(A, B)	(D, C)
(D, C)	(E, E)	(F, F)
(E, C)	(E, E)	(F, F)
(E, D)	(E, E)	(F, F)
(F, A)	(F, B)	**(F, C)**
(F, B)	**(F, A)**	

Unmarked:
(B, A) (D, C) (E, C) (E, D)

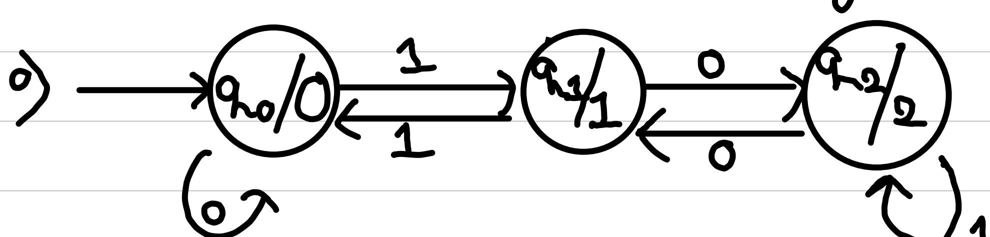
Week 6

- Def-formal (Six tuple notation) : $M=(Q,\Sigma,\delta,q_0,\Delta,\lambda)$
  - Q : Set of states
  - Σ : Set of input symbols
  - δ : Transition function = $Q \times \Sigma \to Q$
  - q_0 : Start state
  - Δ : Set of output
  - λ : Output function = $\Sigma \times Q \to \Delta$

- FSA dengan output :

1) Moore Machine : Output ada di simpul dengan nama state

- n inputs → (n+1) outputs
- Contoh Soal : Mesin moore untuk fungsi modulo 3



- $0 \bmod 3 \equiv 0$
- 0001 $1 \bmod 3 \equiv 1$
- 0010 $2 \bmod 3 \equiv 2$
- 0011 $3 \bmod 3 \equiv 0$
- 0100 $4 \bmod 3 \equiv 1$
- 0101 $5 \bmod 3 \equiv 2$
- 0110 $6 \bmod 3 \equiv 0$
- 0111 $7 \bmod 3 \equiv 1$
- 1000 $8 \bmod 3 \equiv 2$
- 1001 $9 \bmod 3 \equiv 0$

- $Q = \{q_0, q_1, q_2\}$
- $\Sigma = \{0, 1\}$
- $q_0 = \{q_0\}$
- $\Delta = \{0, 1, 2\}$
-

δ	0	1
$\to q_0$	q_0	q_1
q_1	q_2	q_0
q_2	q_1	q_2

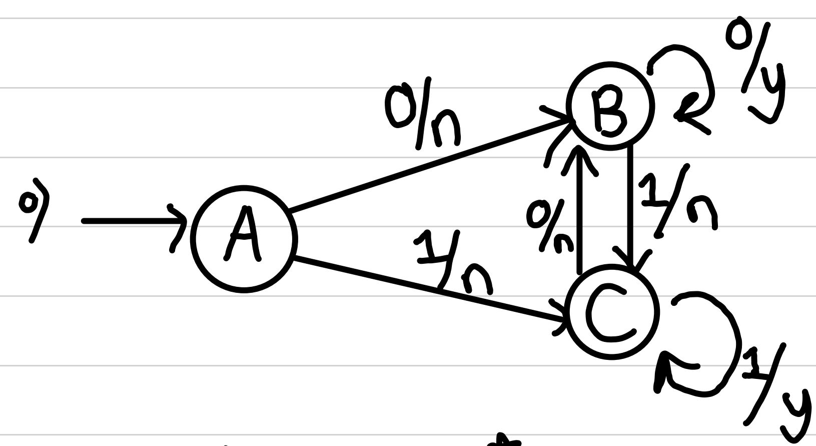
-

λ	0	1
$\to q_0$	0	1
q_1	2	0
q_2	1	2

- So $M = (\{q_0, q_1, q_2\}, \{0,1\}, \delta, \{q_0\}, \{0,1,2\}, \lambda)$

2) Mealy Machine : Output ada di tanda panah dengan input

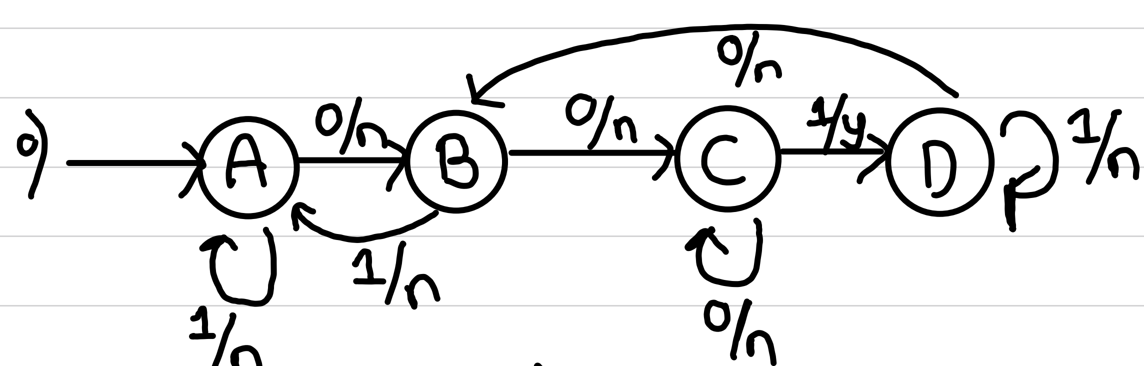
- Contoh Soal : Mesin mealy yang menerima bahasa yang mengandung string dari Σ^*, dimana $\Sigma = \{0,1\}$ dan stringnya harus berakhiran "00" atau "11" (Cetak 'y', kalau tidak 'n')



○) Contoh cek Σ^*

input	0	1	1	0	0	0	1
output	n	n	y	n	y	y	n

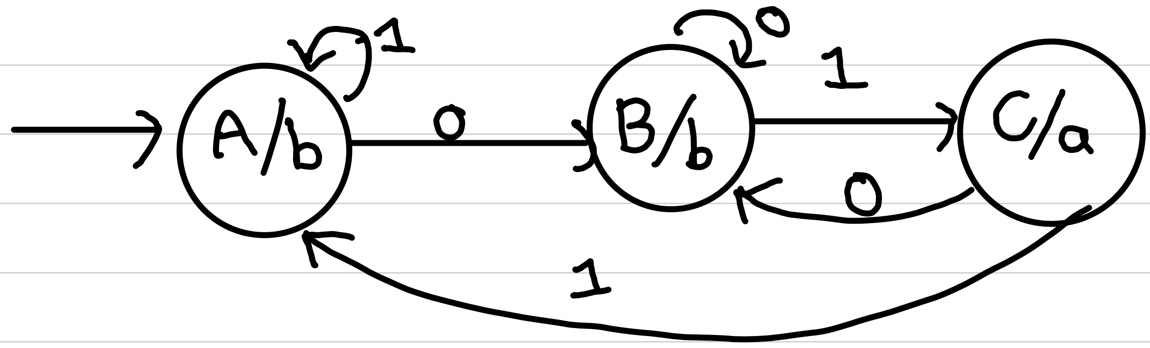
- Contoh Soal : Buat Mealy yang menerima input "001" cetak 'y', selain itu cetak 'n'



○) Contoh cek Σ^*

input	1	0	0	1	0	1	0	0	0	1
output	n	n	n	y	n	n	n	n	n	y

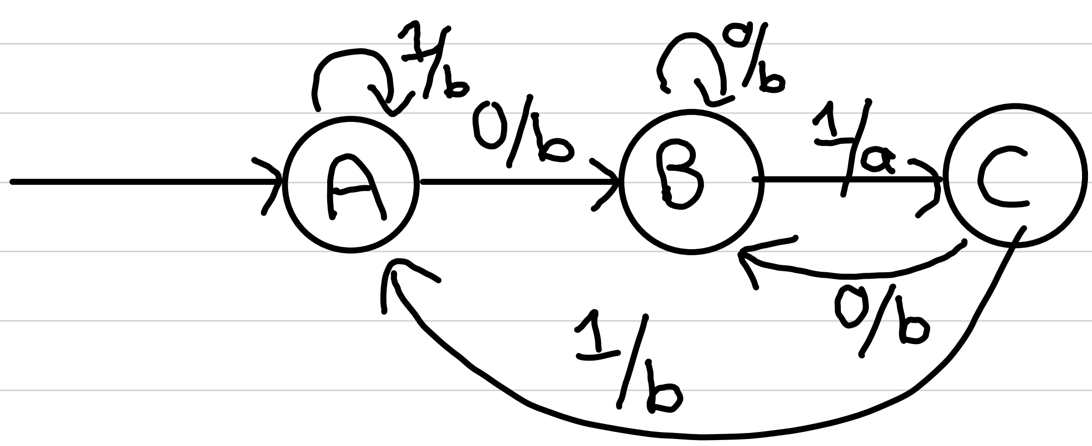
Konversi Moore ke Mealy



State	0	1	Output 0	Output 1
→A	B	A	b	b
B	B	C	b	a
C	B	A	b	b

menjadi

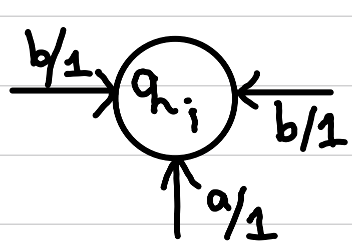
State	0	1
→A	B/b	A/b
B	B/b	C/a
C	B/b	A/b



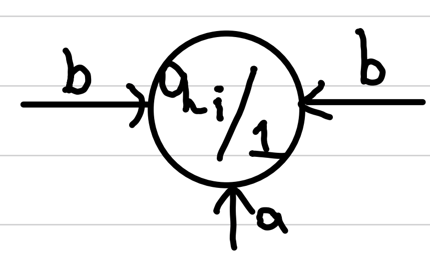
Konversi Mealy ke Moore

- Aturan konversinya yaitu :

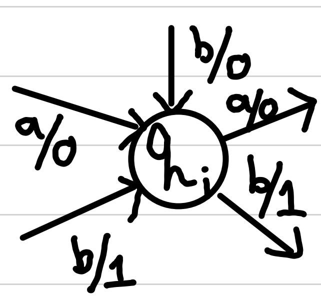
1)



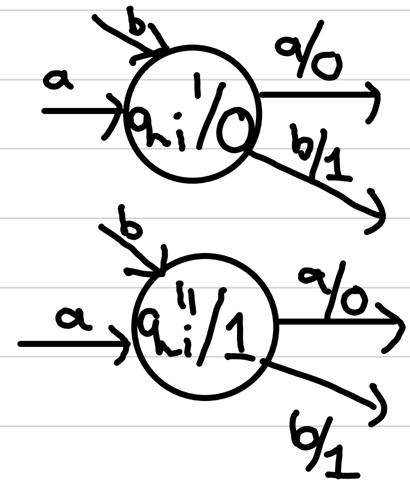
menjadi



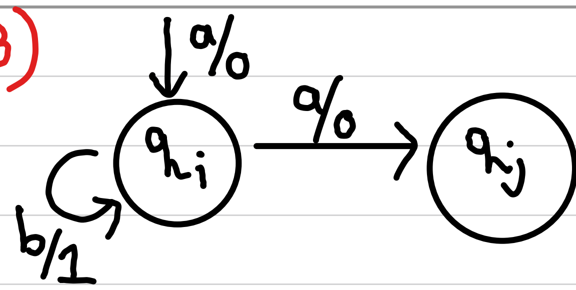
2)



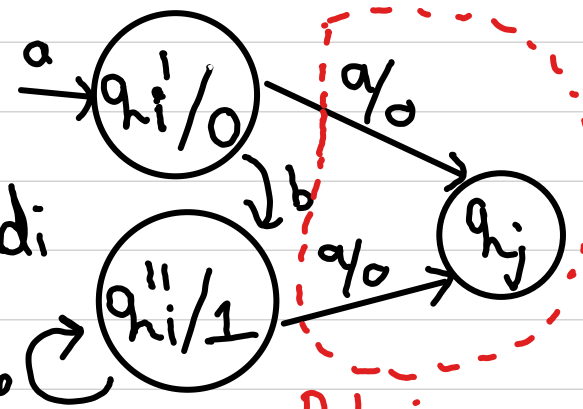
menjadi



3)

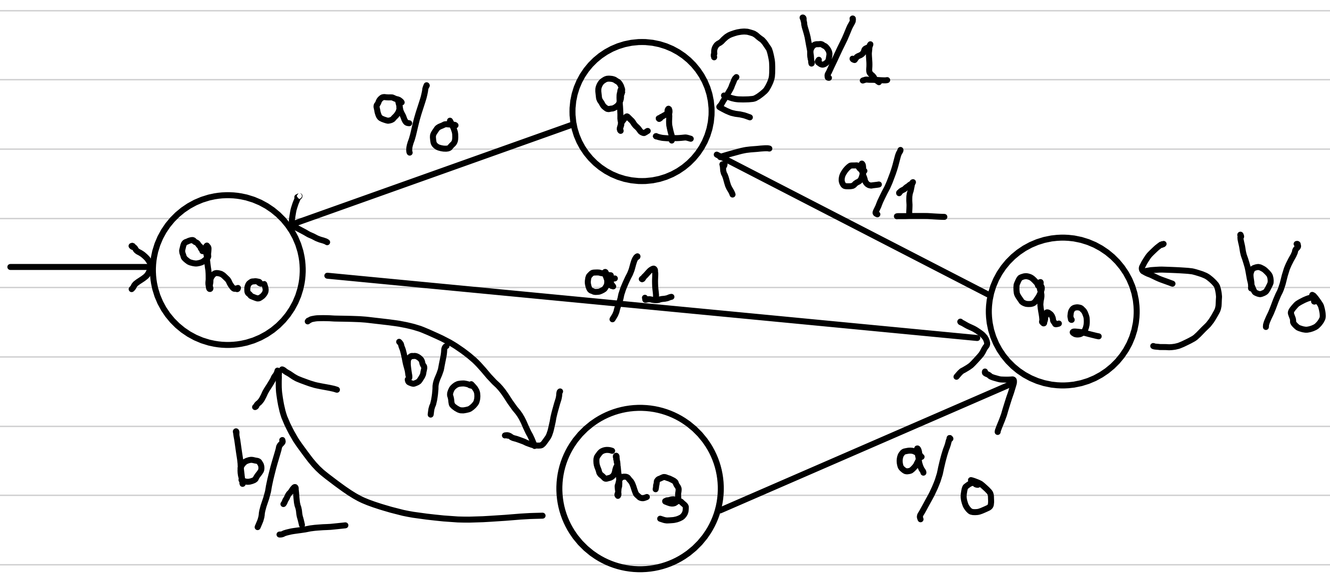


menjadi

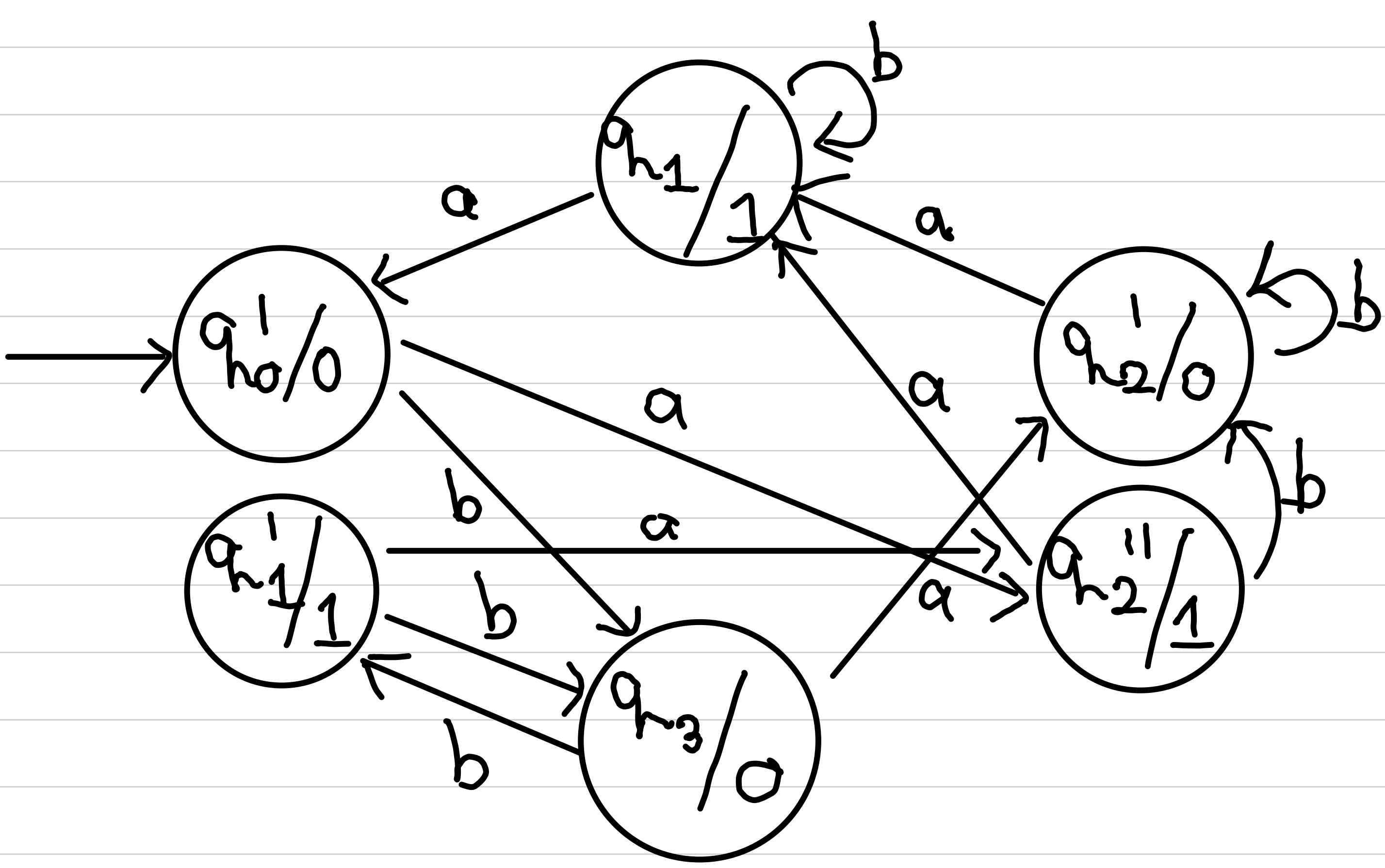


Pakai aturan 1

o



menjadi



- Moore → Mealy : Number of states are same
- Mealy → Moore : Number of states increase
  - Mealy : x states & y outputs
  - Moore : max $(x * y)$ number of states

Week 7

- Regular expression (RE) = Representing certain sets of strings in an algebraic fashion.
  - ↳ Any terminal symbols (symbols $\in \Sigma$, including ϕ) = RE
  - ↳ Union of 2 RE is also a RE : $R_1, R_2 \rightarrow R_1 + R_2$
  - ↳ Concatenation of 2 RE is also a RE : $R_1, R_2 \rightarrow R_1 \cdot R_2$
  - ↳ Iteration / Closure of a RE is also a RE : $R \rightarrow R^*$
  - ↳ RE over Σ are precisely those obtained recursively by the application of the above rules once or several times

Contoh Soal :

- $\{0, 1, 2\}$: $R = 0 + 1 + 2$
- $\{abb, a, b, bba\}$: $R = abb + a + b + bba$
- $\{\epsilon, 0, 00, 000, \ldots\}$: $R = 0^*$
- $\{1, 11, 111, 1111, \ldots\}$: $R = 1^+$
- $R = (0+1)^* 00 (0+1)^*$: $\{00, 10010, 010011, \ldots\}$
- $R = (1 + 10)^*$: $\{\epsilon, 1, 110\}$
- $R = (aa + ab + ba + bb)^+ a$: $\{aaa, aba, abbaa, \ldots\}$
- $R = ab^* a$: $\{aa, aba, abba, abbba, \ldots\}$
- Design RE for language accepting strings of length exactly 2 over $\{a, b\}$: $L = \{aa, ab, ba, bb\}$

$$R = aa + ab + ba + bb$$
$$= a(a+b) + b(a+b)$$
$$= (a+b)(a+b)$$

- Design RE for language accepting strings of length at least 2 over $\{a, b\}$: $L = \{aa, bb, ab, ba, aaa, \ldots\}$

$$R = (a+b)(a+b)(a+b)^*$$

- Design RE for language accepting strings of length at most 2 over $\{a, b\}$: $L = \{\epsilon, a, b, aa, ab, ba, bb\}$

$$R = (\epsilon + a + b)(\epsilon + a + b)$$

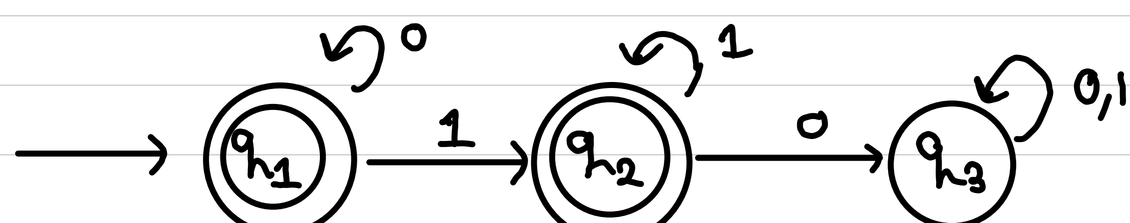
Identities of RE :

1) $\phi + R = R$
2) $\phi R = R\phi = \phi$
3) $\epsilon R = R\epsilon = R$
4) $\epsilon^* = \epsilon$ and $\phi^* = \epsilon$
5) $R + R = R$
6) $R^* R^* = R^*$
7) $RR^* = R^* R = R^+$
8) $(R^*)^* = R^*$
9) $\epsilon + RR^* = \epsilon + R^* R = R^*$
10) $(PQ)^* P = P(QP)^*$
11) $(P+Q)^* = (P^* Q^*)^* = (P^* + Q^*)^*$
12) $(P+Q)R = PR + QR$ and $R(P+Q) = RP + RQ$

Arden's Theorem =

Let P and Q be 2 RE over Σ. If P doesn't contain ϵ, then the following equation in R, $R = Q + RP$ has a unique solution given by $R = QP^*$

Konversi FSA ke RE



- $q_1 = \varepsilon + q_1 0$
- $q_2 = q_1 \cdot 1 + q_2 \cdot 1$
- $q_3 = q_2 \cdot 0 + q_3 0 + q_3 \cdot 1$

- $\underbrace{q_1}_{R} = \underbrace{\varepsilon}_{Q} + \underbrace{q_1}_{R} \cdot \underbrace{0}_{P}$ ← Arden's Theorem

  $q_1 = \varepsilon 0^* = 0^*$

- $\underbrace{q_2}_{R} = \underbrace{q_1 \cdot 1}_{Q} + \underbrace{q_2}_{R} \cdot \underbrace{1}_{P}$ ← Arden's Theorem

  $q_2 = q_1 \cdot 1 \cdot 1^* = 0^* 1 1^*$

- Union of final states q_1 and $q_2 \Rightarrow R = q_1 + q_2$

$$= 0^* + 0^* 1 1^*$$
$$= 0^* (\varepsilon + 1 1^*)$$
$$= 0^* 1^*$$