Chapter 3

2. 유한 오토마타 : Part II

목차

NFA versus DFA

- **For every NFA, there *is* a DFA that accepts the same set of strings**
  - NFA may have transitions labeled by ε (*spontaneous* transitions)
  - All transitions labels in a DFA belongs to Σ
  - For some string *x*, there may be ***many*** accepting paths in an NFA
  - For all strings *x*, there is ***one unique*** accepting path in a DFA

  - Usually, an input string can be recognized ***faster*** with a DFA
  - NFAs are typically ***smaller*** than the corresponding DFAs

From Regular Expressions to DFAs

Regular Expression to NFA

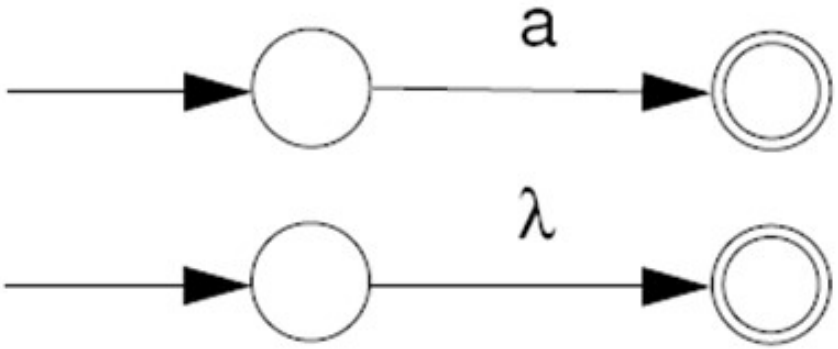
- *Thompson' Construction*

  개별 정규표현에 대한 NFA를 구성

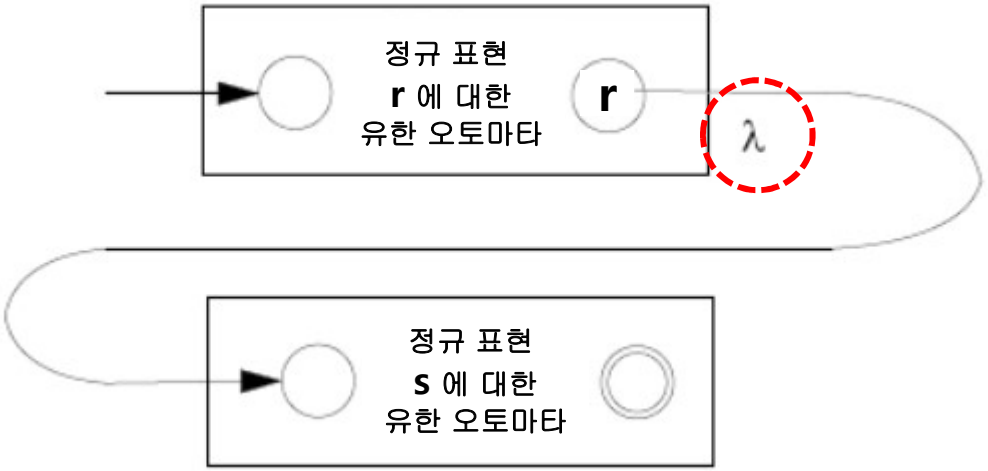
  → λ- (ε-) transition 을 사용하여 이들을 연결

  → 완전한 NFA를 구성

For *basic* regular expressions, $a \in \Sigma$



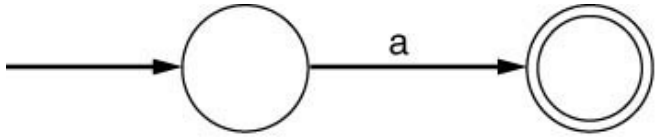
Concatenation : *rs*



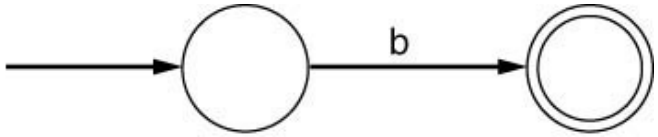
Uses λ-transition to "*glue together*" the NFAs of each piece of regular expression

RE ➔ NFA변환 : Example 1(a)

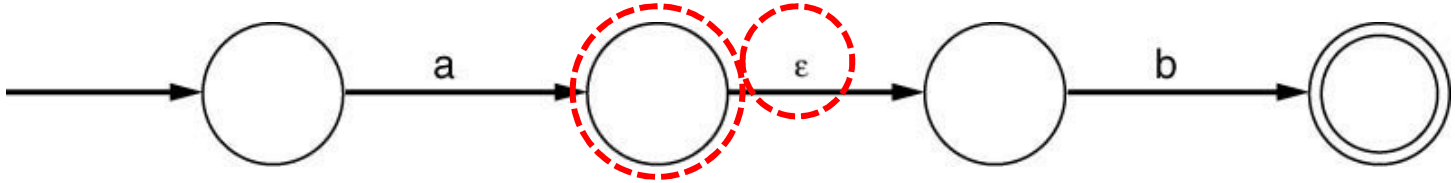
a



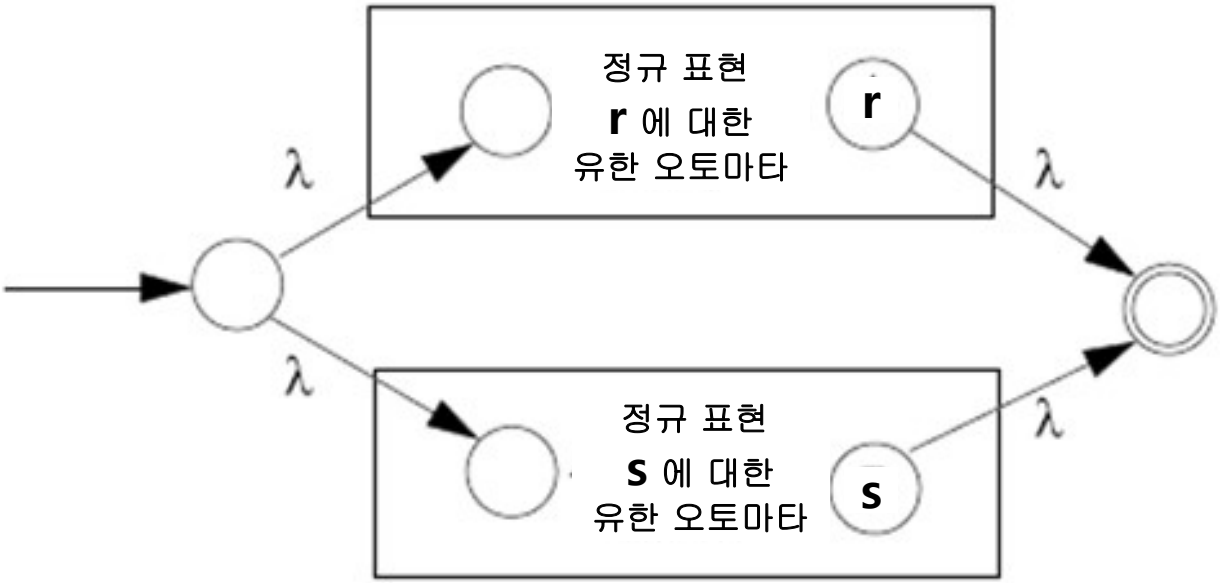
b



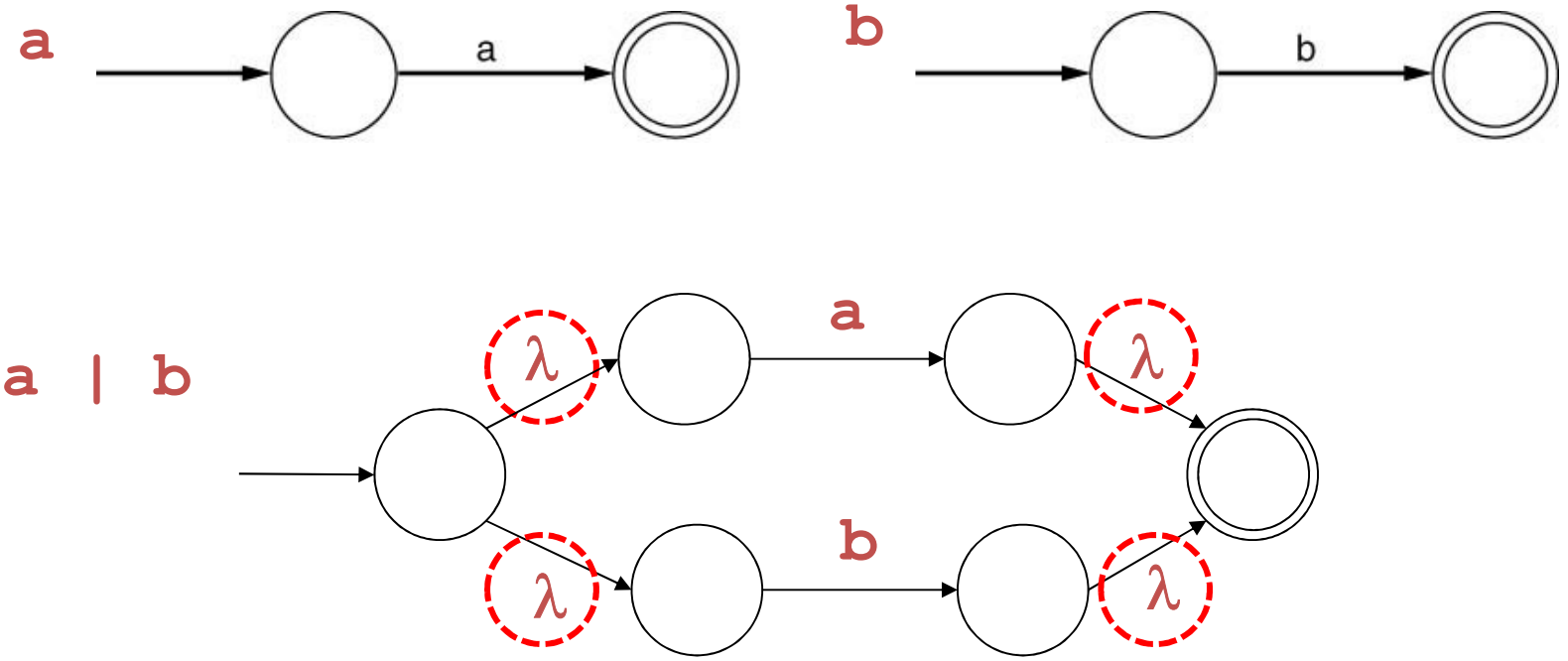
ab



Choice Among Alternatives: *r*|*s*

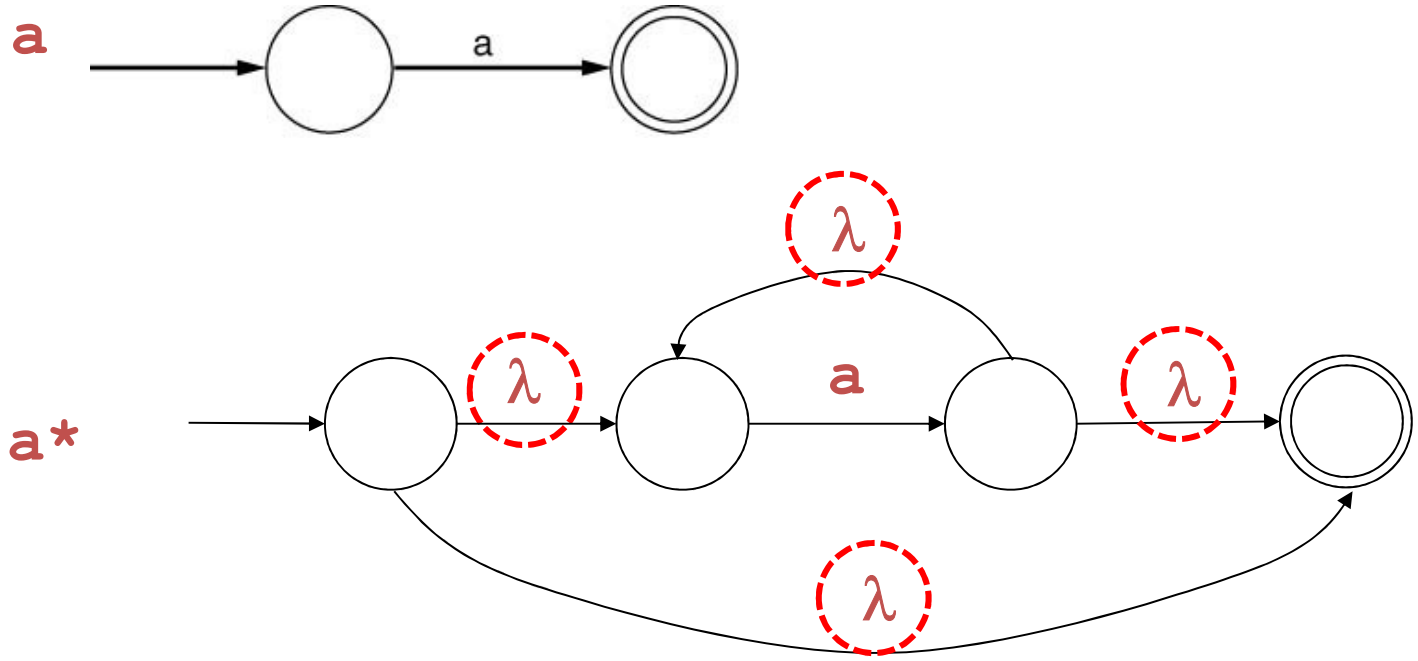


RE ➔ NFA변환 : Example 1(b)



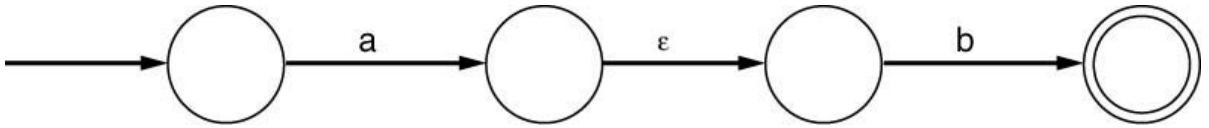
Repetition : r*

RE ➔ NFA변환 : Example 1(c)

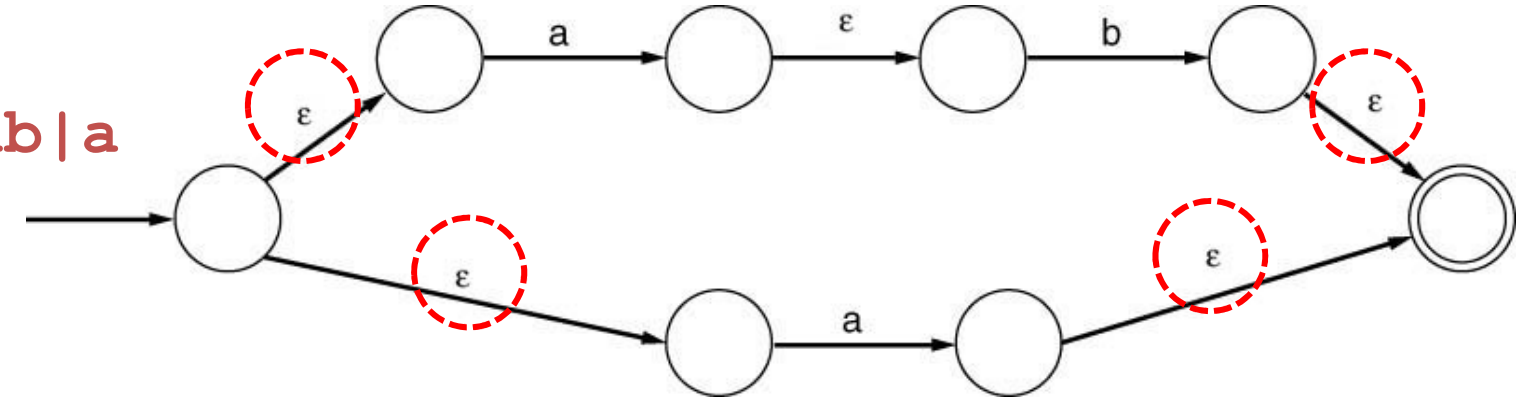


RE ➔ NFA변환 : Example 1(d)

ab

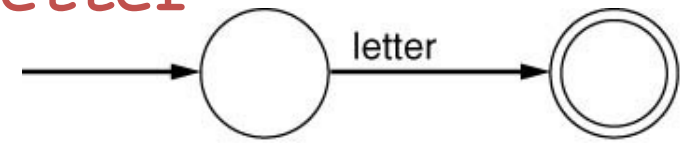


ab | a

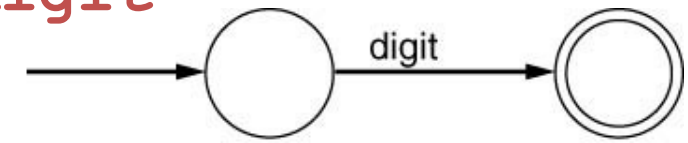


RE ➔ NFA변환 : Example 2 (1/3)

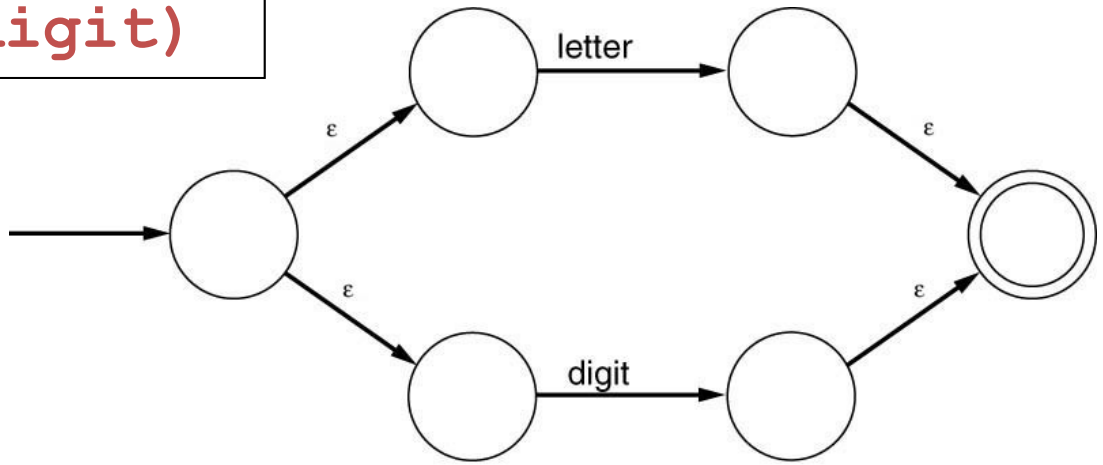
letter



digit

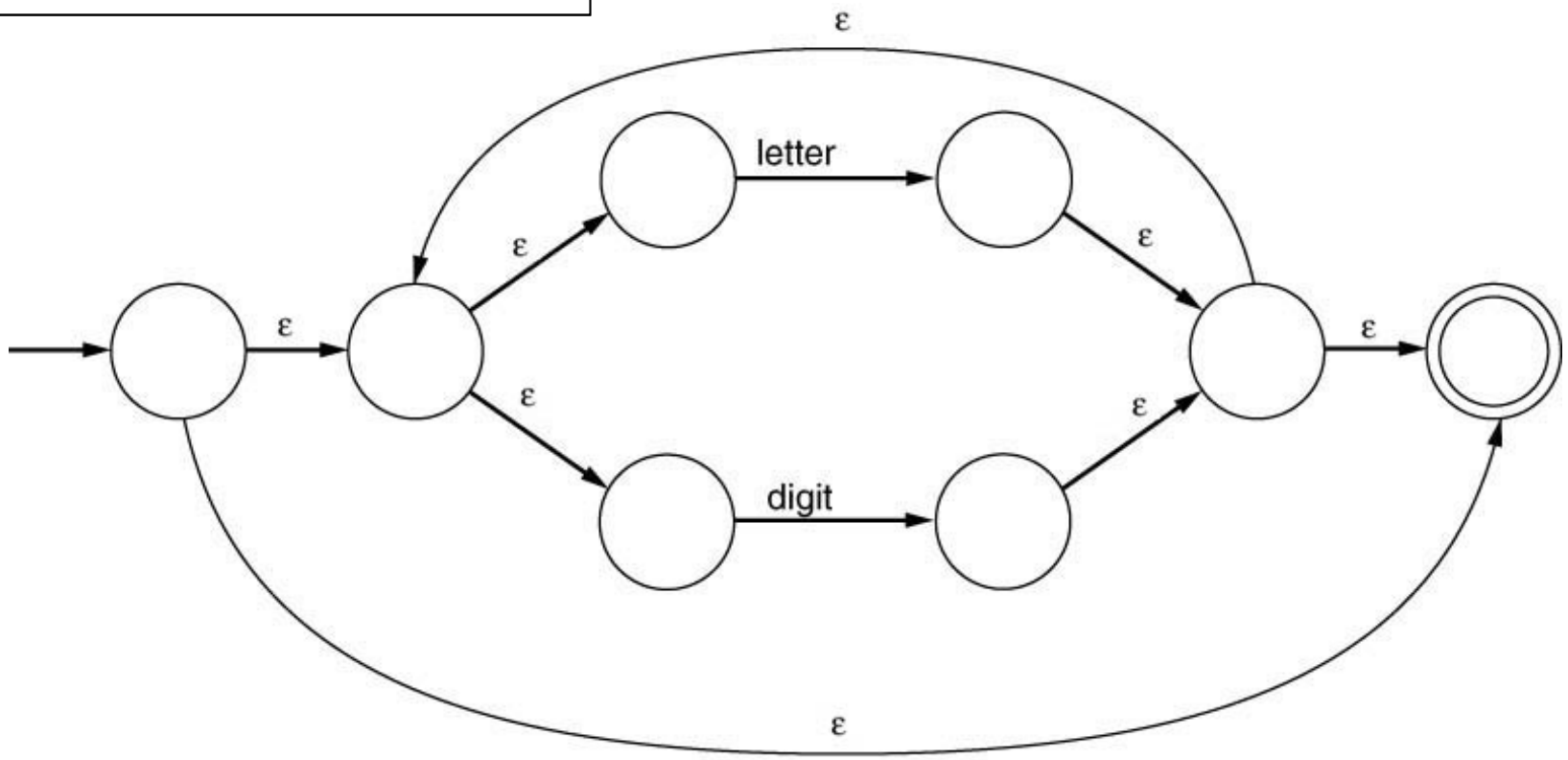


(letter|digit)



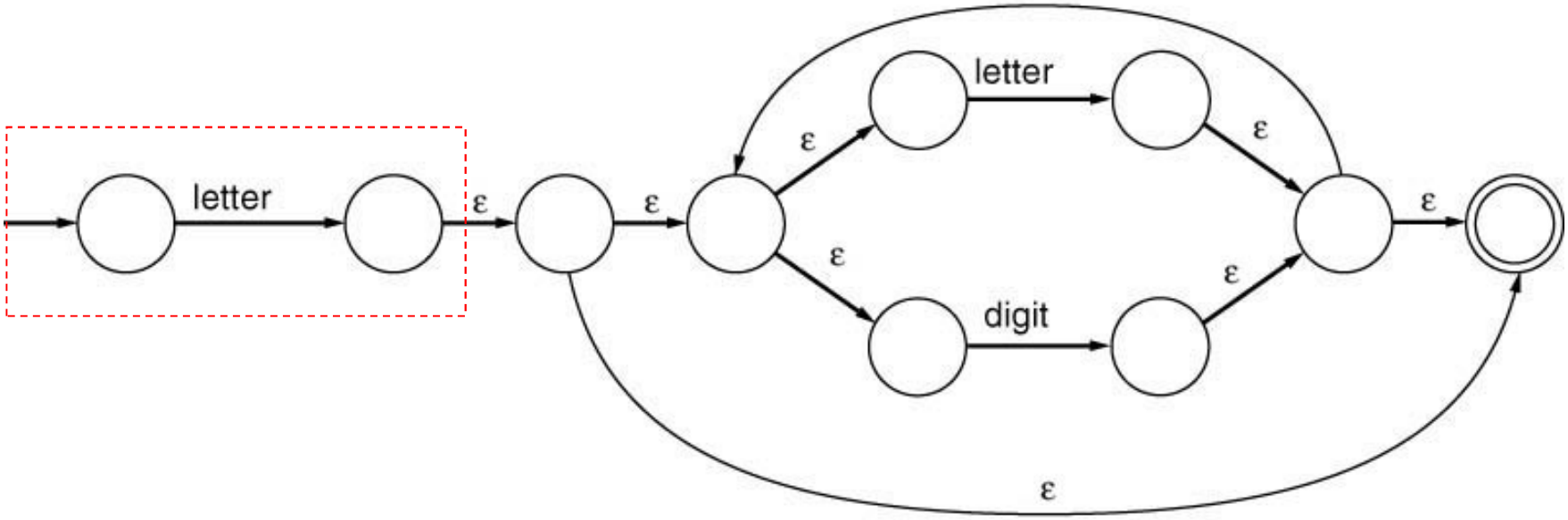
RE ➔ NFA변환 : Example 2 (2/3)

```
(letter|digit)*
```



RE ➔ NFA변환 : Example 2 (3/3)

```
letter(letter|digit)*
```



Quiz #1

아래 정규 표현을 NFA로 변환하시오.

`(a|b)*a(a|b)`

From an NFA to DFA

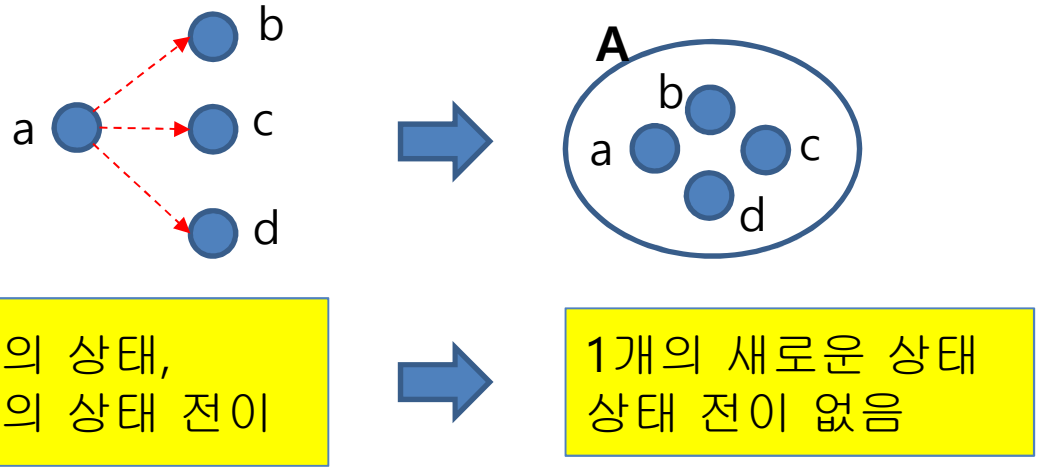
■ *Subset construction* (부분집합 구성)

- Eliminate *ε-transition*
  - ε-transition으로 도달할 수 있는 상태(***ε- closure***)는 독립적인 상태가 아니라 하나의 상태로 취급
    - ε- *closure* 를 구함 → $\{q_0, q_1, q_2, \ldots, q_N\}$ → Q
  - 없앤다(*eliminate*)는 …
    - **NFA의 여러 개의 상태들이 DFA에서는 하나의 상태로 묶임**

- Eliminate *multiple transition* on a single input character
  - 같은 입력 문자로 도달할 수 있는 상태들 역시 독립 상태가 아니라 이들을 묶어 하나의 상태로 취급

From an NFA to DFA

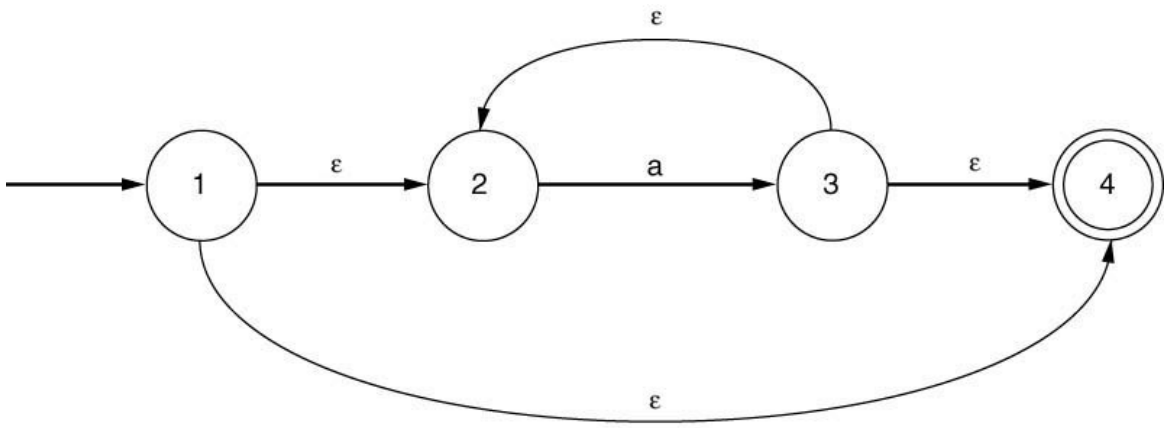
■ ***Subset construction*** **(부분집합 구성)**



From NFA to DFA : *ε-transition* 제거

- **ε-transition 제거 : *ε-closure*를 구하면 됨**
  - ε-closure(***s*** : state)
    - the set of states reachable by a series of *zero or more ε-transitions from s* (**자신을 포함**)

From NFA to DFA : *ε-transition* 제거 예(1)



ε-closure(1) = {1,2,4}

ε-closure(2) = {2}

ε-closure(3) = {2,3,4}

ε-closure(4) = {4}

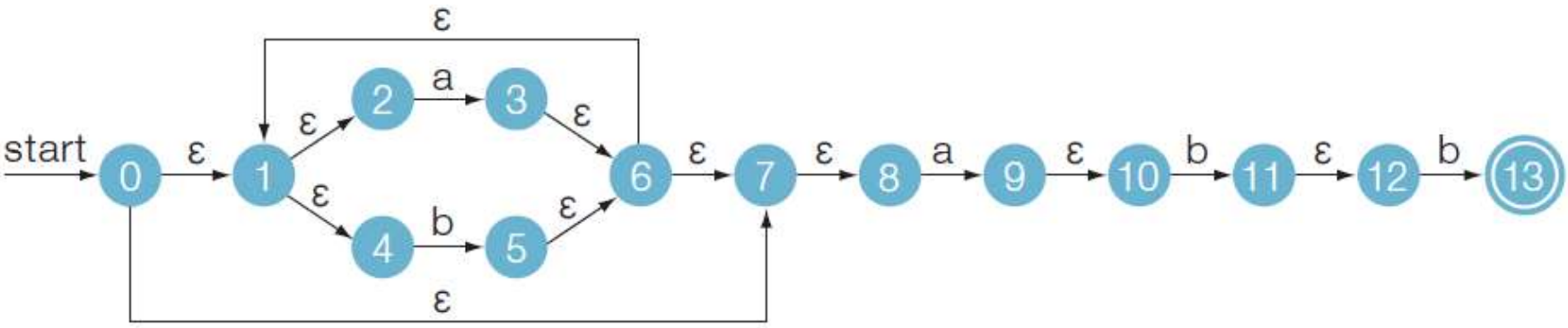
$$\varepsilon\text{-closure}(S) = \bigcup_{s \text{ in } S} \varepsilon\text{-closure}(s)$$

ε-closure({1,3})

= ε-closure(1) ∪ ε-closure(3)

= {1,2,3,4}

From NFA to DFA : *ε-transition* 제거 예(2)



ε-closure(1) = {1, 2, 4}

ε-closure(3) = {1, 2, 3, 4, 6, 7, 8}

ε-closure(5) = {1, 2, 4, 5, 6, 7, 8}

ε-closure(9) = {9, 10}

ε-closure(11) = {11, 12}

ε-closure(0, 9) = ε-closure(0) ∪ ε-closure(9) = {0, 1, 2, 4, 7, 8, 9, 10}

ε-closure(5, 11) = ε-closure(5) ∪ ε-closure(11) = {1, 2, 4, 5, 6, 7, 8, 11, 12}

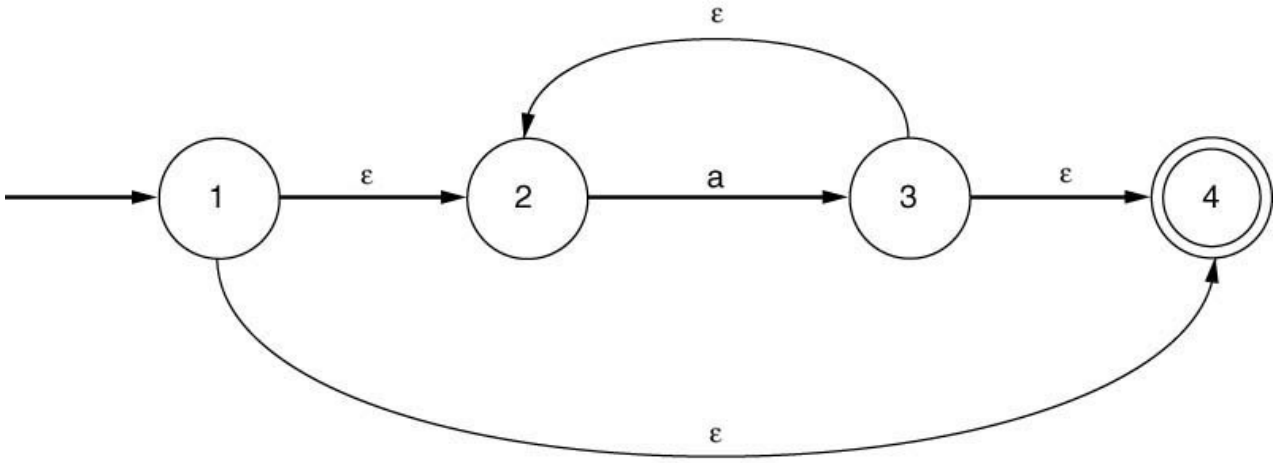
From NFA to DFA : *The Subset Construction*

- **The *start state* of DFA = *ε-closure* (*the start state of NFA*)**

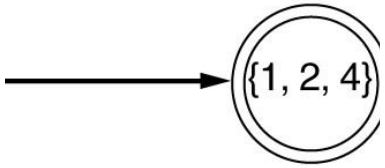
- **새로운 상태가 만들어지지 않거나 *transition*이 나타나지 않을 때까지 아래 과정을 반복 실행**
  - S' = ***transition*** $(S, a) = \{ t \mid$ for some s in S
    there is a transition from s to t on a, where $a \in \Sigma \}$
  - ***ε-closure*** (S') = S''
    - S'' 은 DFA의 새로운 상태.
  - 상태 천이 함수 $T(S, a) = S''$ 추가.

- **NFA의 accepting state를 하나라도 포함하고 있는 DFA의 상태는 모두 accepting state 가 됨.**

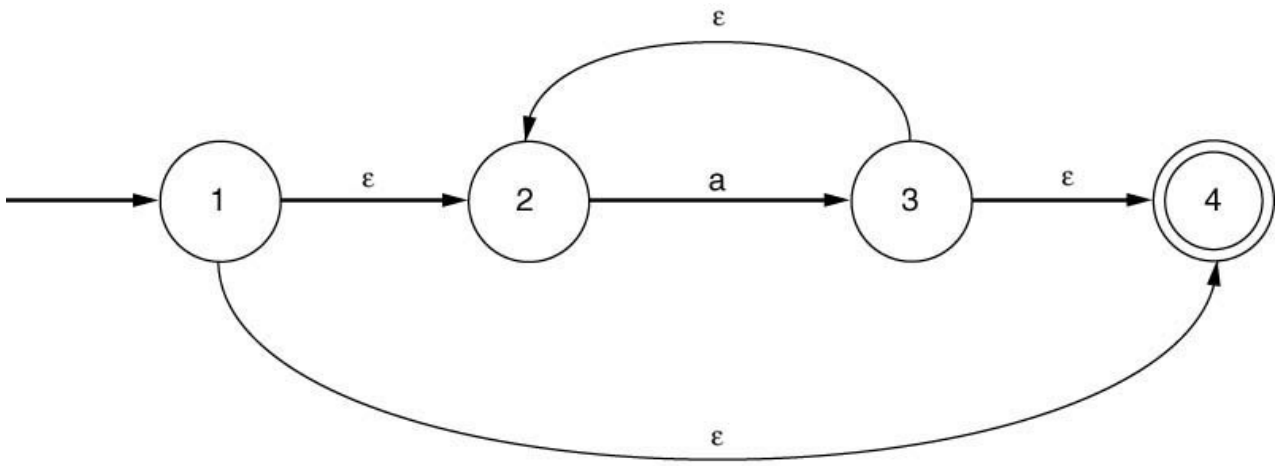
NFA ➔ DFA변환: Example 3 (1/2)



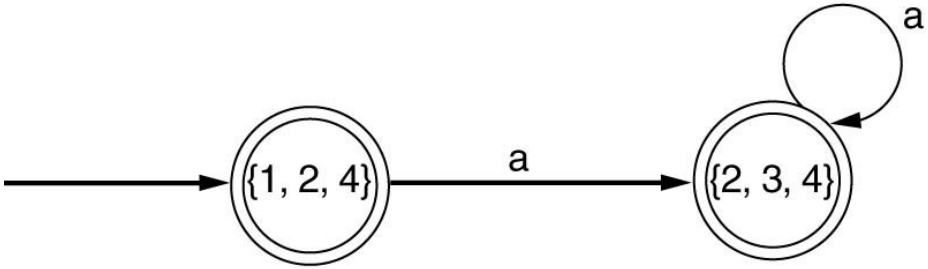
ε-closure(1) = {1,2,4}
transition({1,2,4}, **a**) = {3}



NFA ➔ DFA변환: Example 3 (2/2)



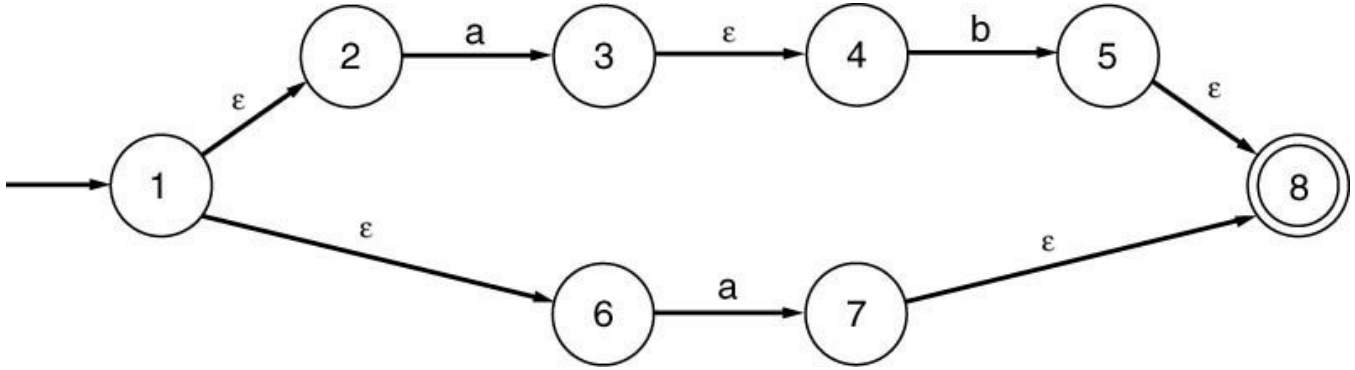
- ε-closure(1) = {1,2,4}
- *transition*({1,2,4}, **a**) = {3}



- ε-closure({3}) = {2,3,4}
- *transition*({2,3,4}, **a**) = {3}

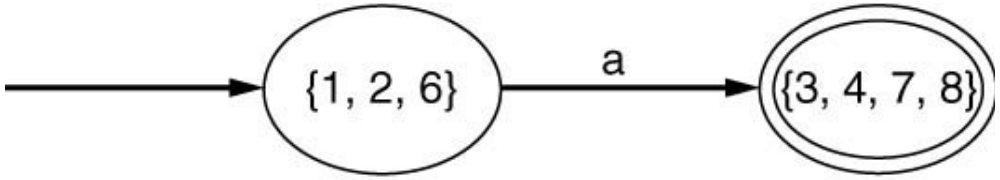
NFA ➔ DFA변환: Example 4 (1/2)

ab|a



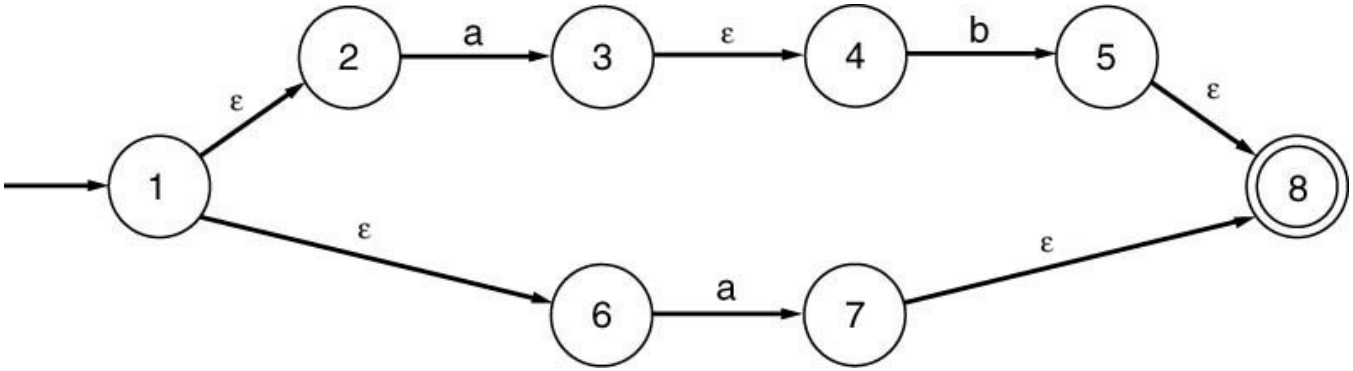
ε-*closure*(1) = {1,2,6}
transition({1,2,6}, **a**) = {3,7}

ε-*closure*({3,7}) = {3,4,7,8}



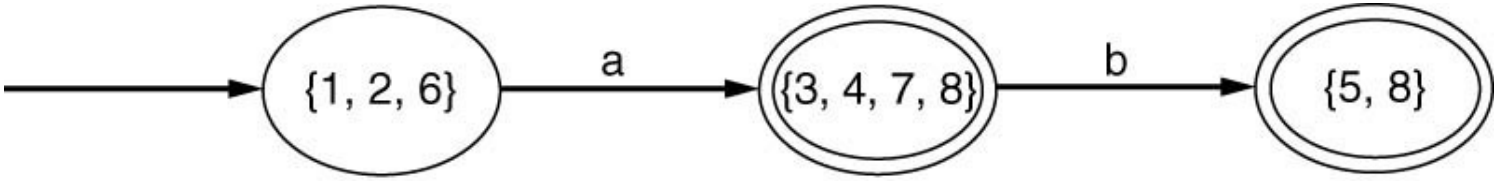
NFA ➔ DFA변환: Example 4 (2/2)

ab|a



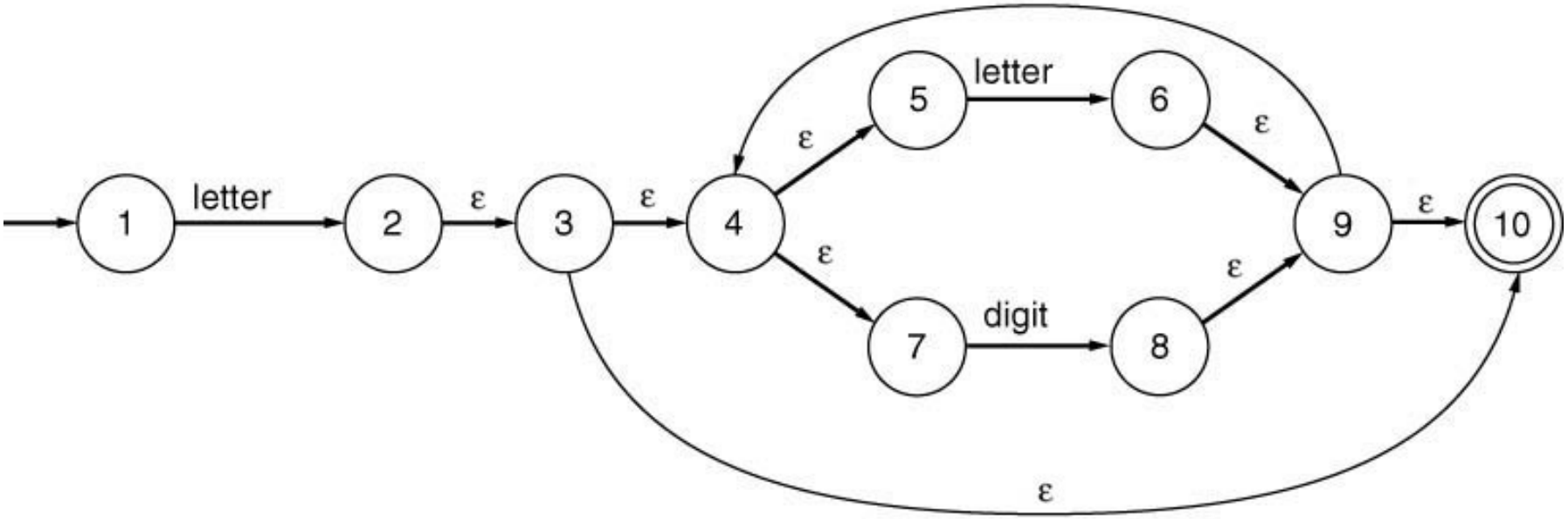
transition({3,4,7,8}, **b**) = {5}

ε-*closure*({5}) = {5,8}

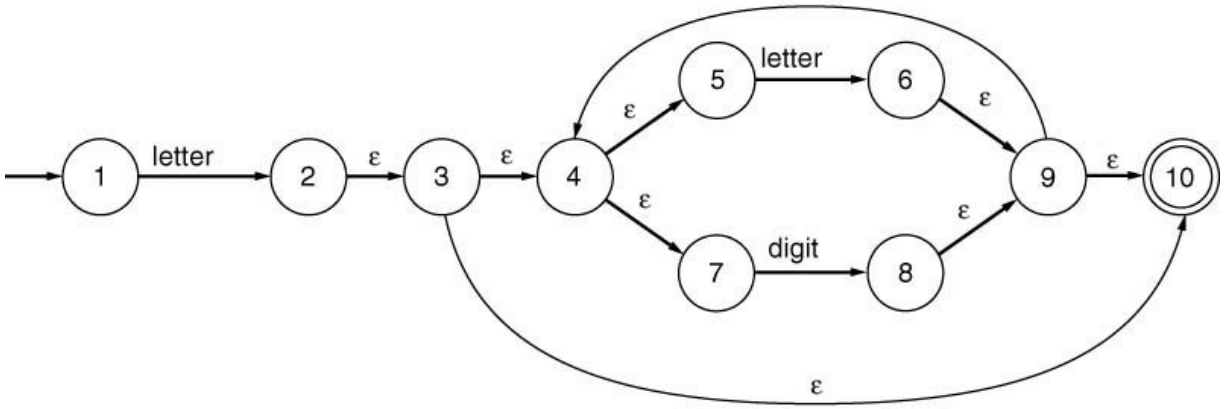


NFA ➔ DFA변환: Example 5 (1/4)

```
letter(letter|digit)*
```



NFA ➔ DFA변환: Example 5 (2/4)



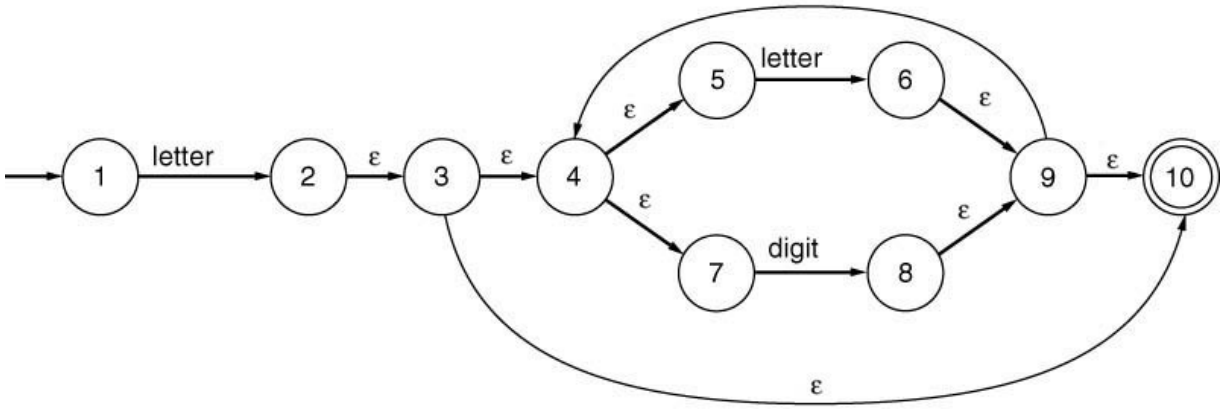
ε-*closure*(1) = {1}

transition({1},letter) = {2}, ε-*closure*({2}) = {2,3,4,5,7,10}

transition({2,3,4,5,7,10}, `letter`) = {6}

ε-*closure*({6}) = {4,5,6,7,9,10}

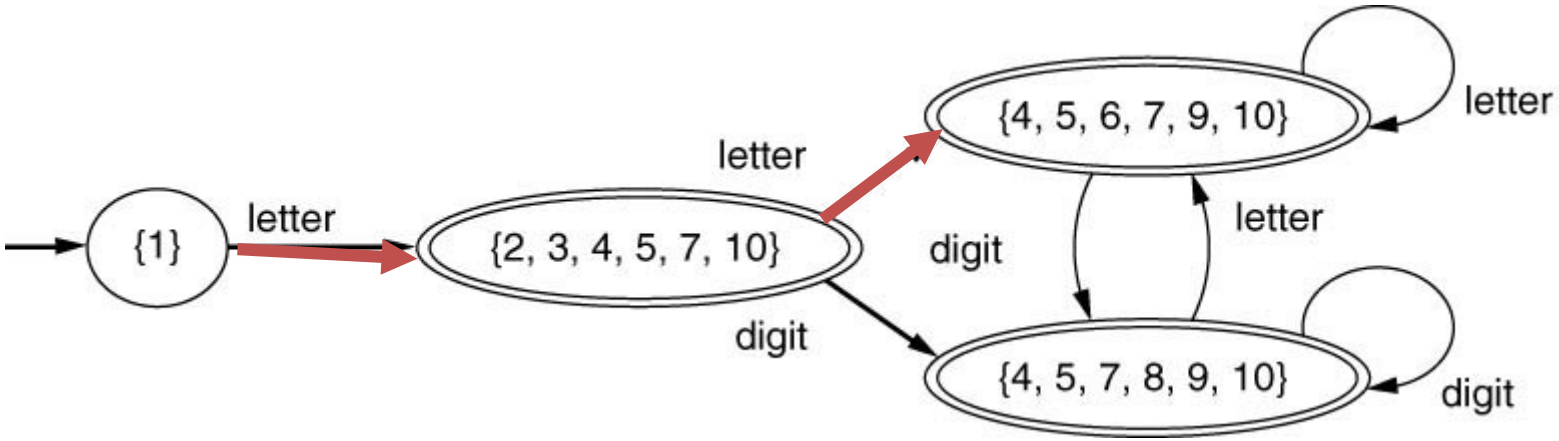
NFA ➔ DFA변환: Example 5 (3/4)



transition({2,3,4,5,7,10}, **digit**) = {8}
ε-*closure*({8}) = {4,5,7,8,9,10}

transition({4,5,6,7,9,10}, **letter**) = {6}
transition({4,5,6,7,9,10}, **digit**) = {8}
transition({4,5,7,8,9,10}, **letter**) = {6}
transition({4,5,7,8,9,10}, **digit**) = {8}

NFA ➔ DFA변환: Example 5 (4/4)



ε-*closure*(1) = {1}

transition({1},letter) = {2}, ε-*closure*({2}) = {2,3,4,5,7,10}

transition({2,3,4,5,7,10}, `letter`) = {6}

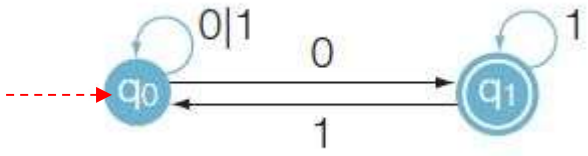
ε-*closure*({6}) = {4,5,6,7,9,10}

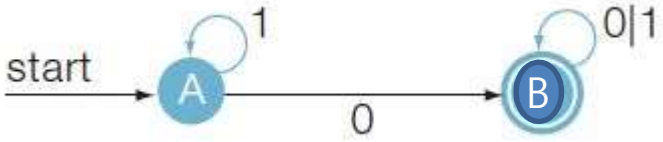
NFA ➔ DFA변환: Example 6

■ 아래 상태 전이 테이블에서 정의하는 NFA를 DFA로 변환하시오.

δ	0	1
q_0	$\{q_0, q_1\}$	$\{q_0\}$
q_1	ϕ	$\{q_0, q_1\}$

NFA $M = (\{q_0, q_1\}, \{0, 1\}, \delta, q_0, \{q_1\})$

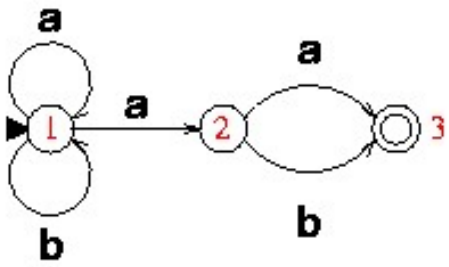




A = $\{q_0\}$
B = $\{q_0, q_1\}$

Quiz #2

아래 NFA를 DFA로 변환하시오.

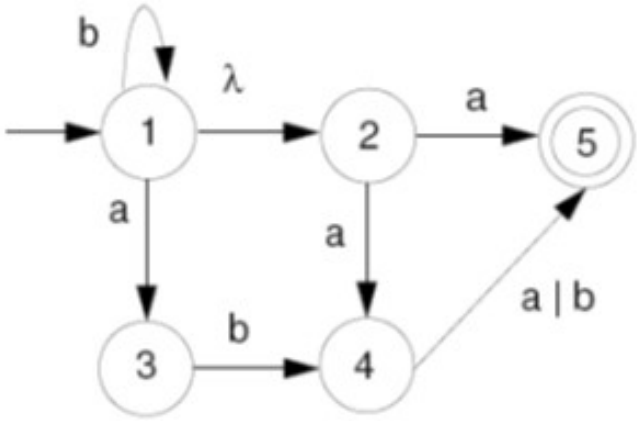


	a	b
1	{1,2}	{1}
2	{3}	{3}
3*	{ }	{ }

$$
\begin{aligned}
\epsilon\text{-closure}(\{1\}) &= \{1\} \\
\text{goto}(\{1\}, \mathrm{a}) &= \{1,2\} \\
\text{goto}(\{1\}, \mathrm{b}) &= \{1\} \\
\text{goto}(\{1,2\}, \mathrm{a}) &= \{1,2,3\} \\
\text{goto}(\{1,2\}, \mathrm{b}) &= \{1,3\} \\
\text{goto}(\{1,2,3\}, \mathrm{a}) &= \{1,2,3\} \\
&\vdots
\end{aligned}
$$

Quiz #3

아래 NFA를 DFA로 변환하시오.



	a	b
1	{3}	{1}
2	{4,5}	{ }
3	{ }	{4}
4	{5}	{5}
5*	{ }	{ }

DFA 상태 최소화(1/2)

- **상태 수가 작으면**
  - 상태 전이 횟수가 줄어들어 인식 속도가 빨라짐
  - 상태 수가 줄어든 만큼 `case` 문이 줄어들어 프로그램 복잡도를 줄이고 실행 속도를 줄일 수 있음

- ***minimum* DFA는 존재하며, 유일하다**

DFA 상태 최소화(2/2)

How?

- *The Big Picture*
  - **동치 상태**(*equivalent states*) 집합을 찾아
    - 동치 상태 집합에 속한 상태들을 하나의 상태로 간주

- 두 개의 상태 p, q 가 동치임을 어떻게 판단하는가?
  - 모든 $\alpha \in \Sigma$ 에 대한 상태 천이 결과가 같을 때 p, q 는 동치

    $\delta(p, \alpha) = \delta(q, \alpha) = r$

Key Idea: Splitting S around α

처음 시작할 때 상태 집합 S를 2개로 나눌 수 있다.

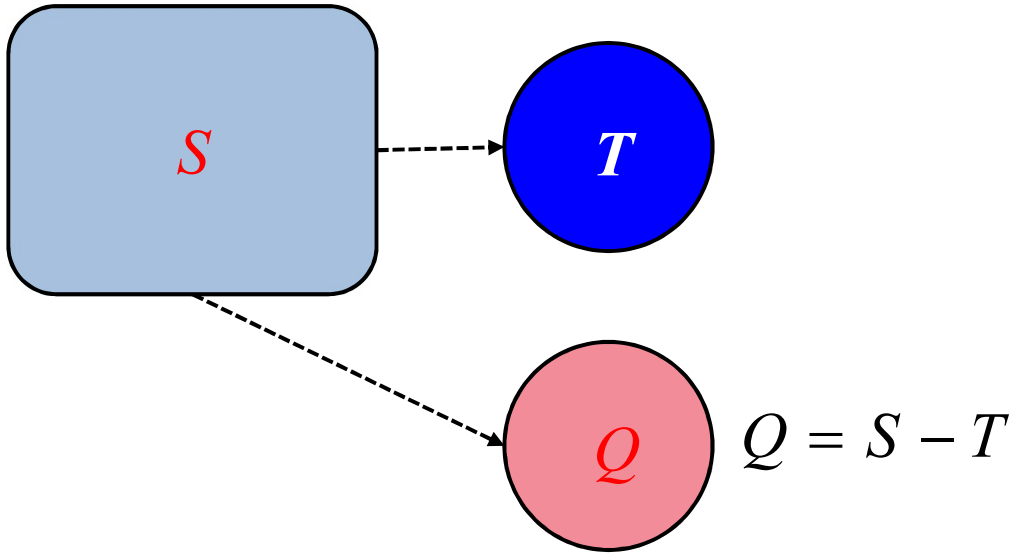
T = {종결 상태}

$Q = S - T$ ={종결 상태 집합에 속하지 않는 상태}

S : 상태 집합

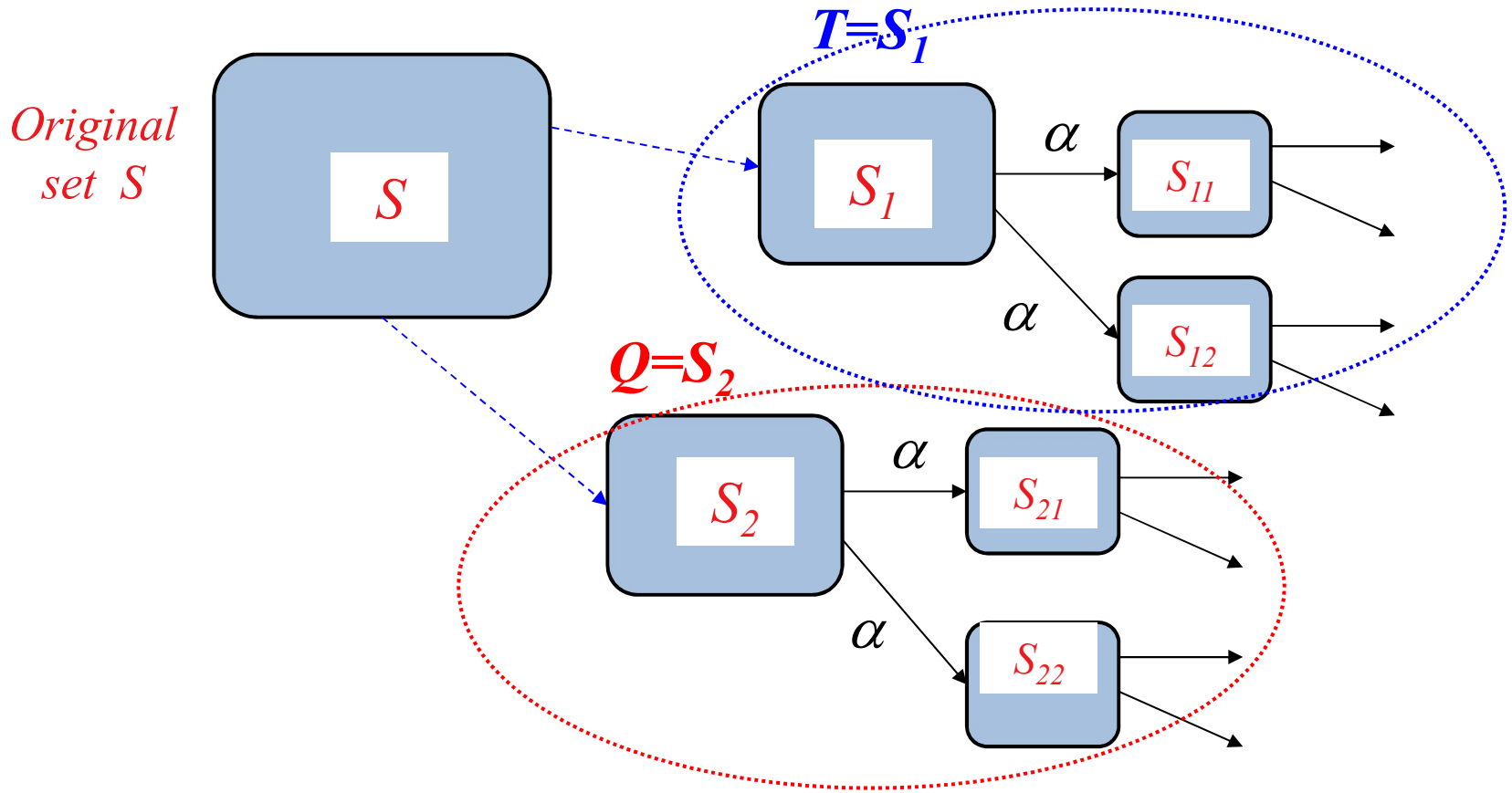
$Q, T \subseteq S$

$\alpha \in \Sigma$



Key Idea: Splitting *S* around α

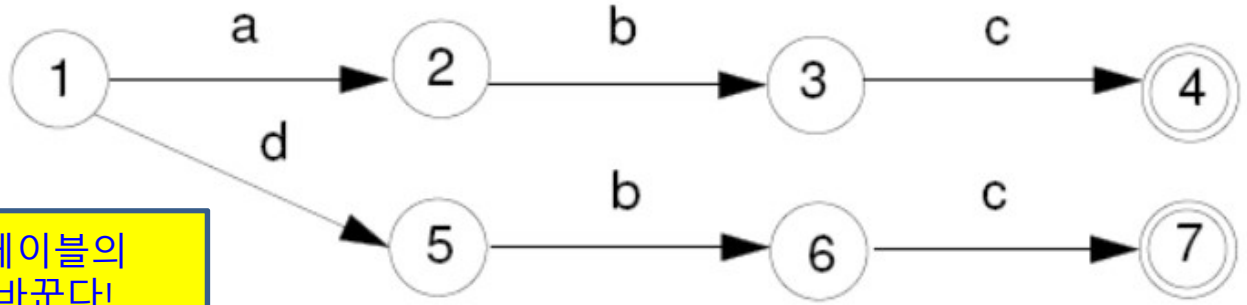
기호 α 에 대한 상태 천이 결과에 따라
상태 집합 T와 Q를 쪼갤 수 있다.



DFA Minimization : *Hopcroft* algorithm

- 상태 집합을 초기에 최대 크기를 갖는 2개의 집합으로 나눈다
  - Initial partition, P_0, has two sets: $\{F\}$ & $\{Q\text{-}F\}$
    - $D = (Q, \Sigma, \delta, q_0, F)$
- 반복해서 이 집합을 분할한다.
- 더 이상 분할되지 않을 경우, 남아있는 상태는 하나의 상태로 묶인다.

상태 최소화 : Example 7 (1/2)



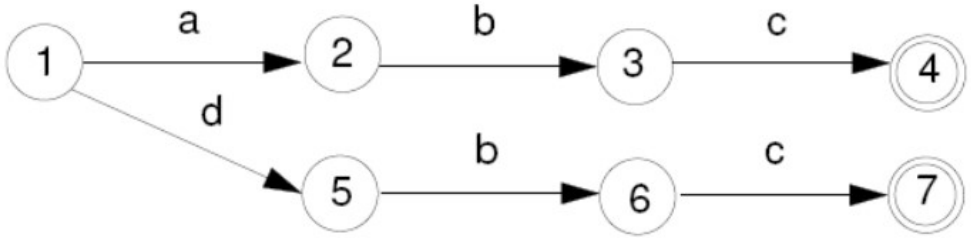
상태 전이 테이블의
행과 열을 바꾼다!

입력	{Q-F}:1					{F}:2	
	1	2	3	5	6	4	7
a	1						
b		1		1			
c			2		2		
d	1						

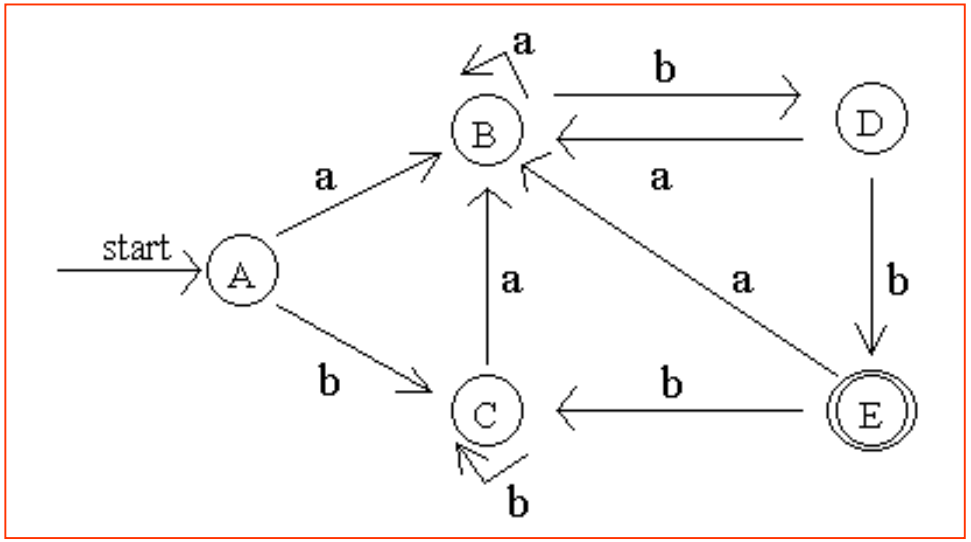
입력	1			3		2	
	1	2	5	3	6	4	7
a	1						
b		3	3				
c				2	2		
d	3						

상태 최소화 : Example 7 (2/2)

입력	1	4	3	2
	1	2 5	3 6	4 7
a	4			
b		3 3		
c			2 2	
d	4			



DFA 상태 최소화 : Example 8 (1/4)



입력 \ 상태	A	B	C	D	E
a	B	B	B	B	B
b	C	D	C	E	C

final states = {*E*},
non-final states = {*A, B, C, D*}

입력	{Q-F}:1				{F}:2
	A	B	C	D	E
a	1	1	1	1	1
b	1	1	1	2	1

DFA 상태 최소화 : Example 8 (2/4)

상태 / 입력	A	B	C	D	E
a	B	B	B	B	B
b	C	D	C	E	C

입력	1				2
	A	B	C	D	E
a	1	1	1	1	1
b	1	1	1	2	1

입력	1			3	2
	A	B	C	D	E
a	1	1	1	1	1
b	1	3	1	2	1

DFA 상태 최소화 : Example 8 (3/4)

입력 \ 상태	A	B	C	D	E
a	B	B	B	B	B
b	C	D	C	E	C

입력	1			3	2
	A	B	C	D	E
a	1	1	1	1	1
b	1	3	1	2	1

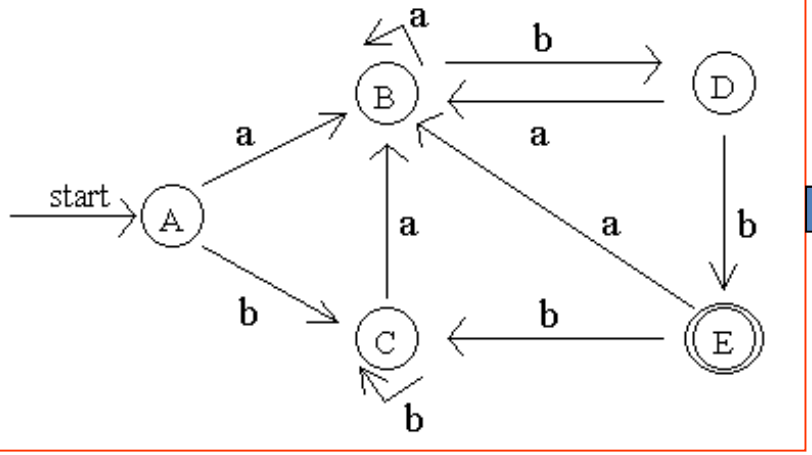
입력	1		4	3	2
	A	C	B	D	E
a	4	4	4	4	4
b	1	1	3	2	1

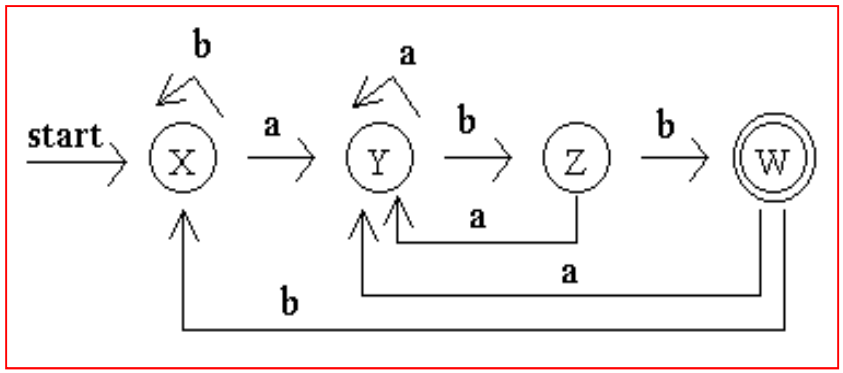
DFA 상태 최소화 : Example 8 (4/4)

입력	1	4	3	2
	A C	B	D	E
a	4 4	4	4	4
b	1 1	3	2	1

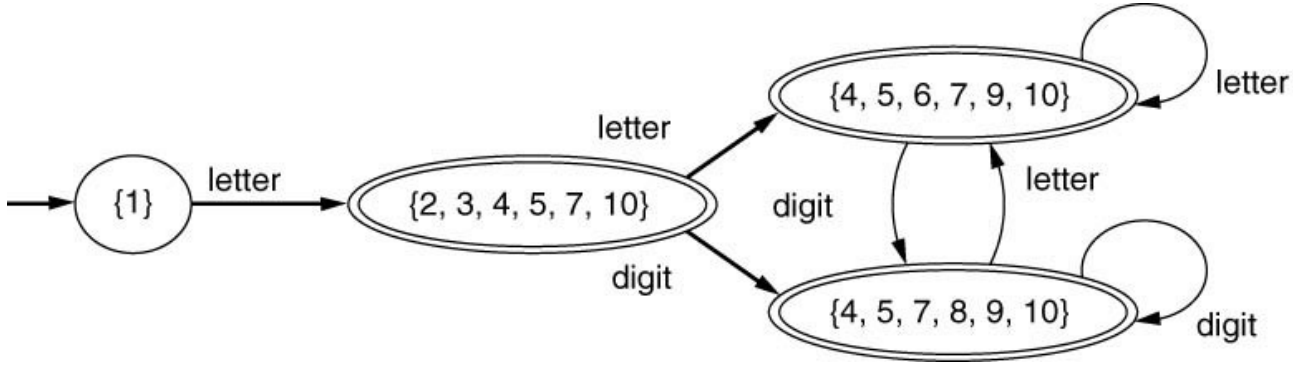
{A,C} → X
{B} → Y
{D} → Z
{E} → W

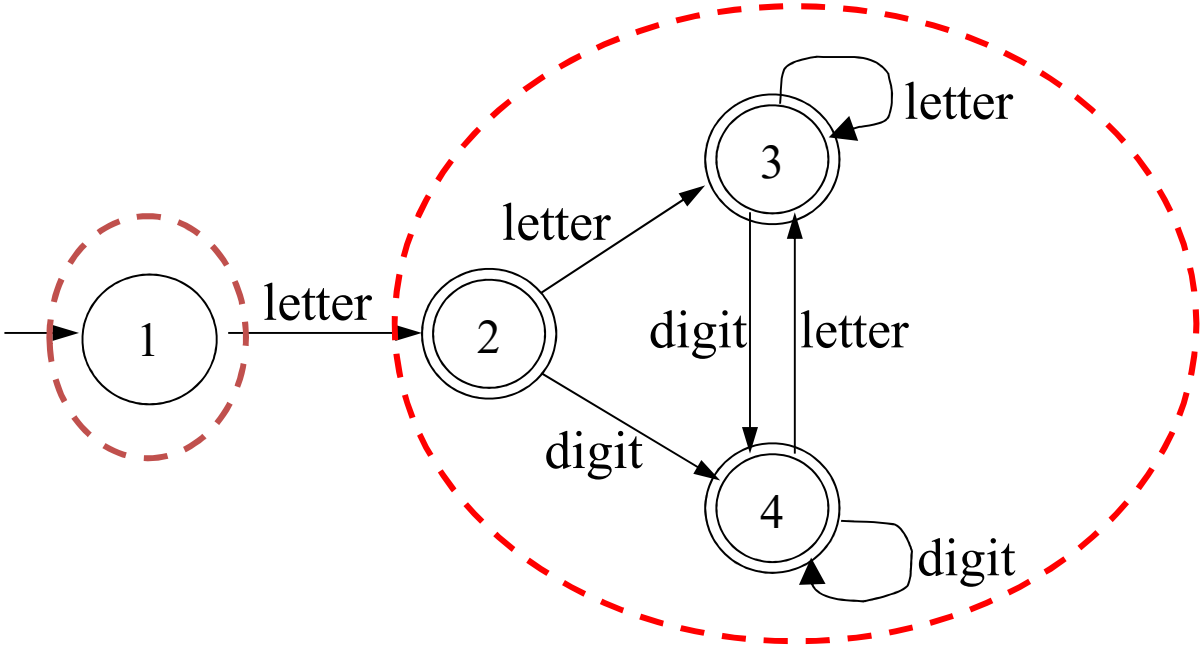
δ	a b
X	Y X
Y	Y Z
Z	Y W
W	Y X





상태 최소화 : Example 9 (1/2)

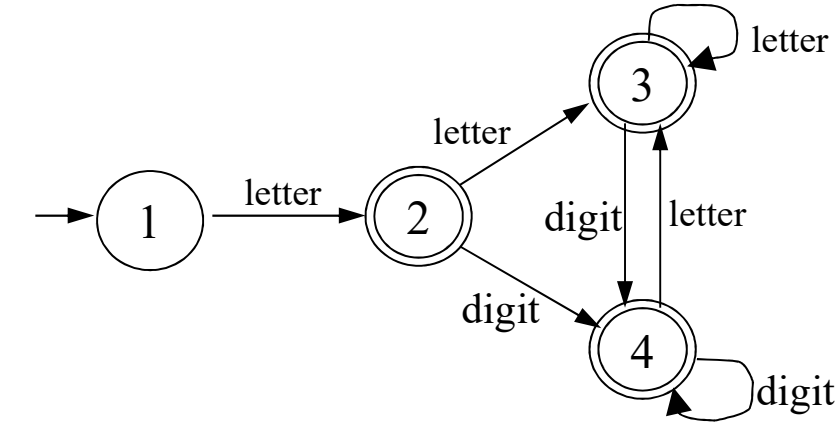




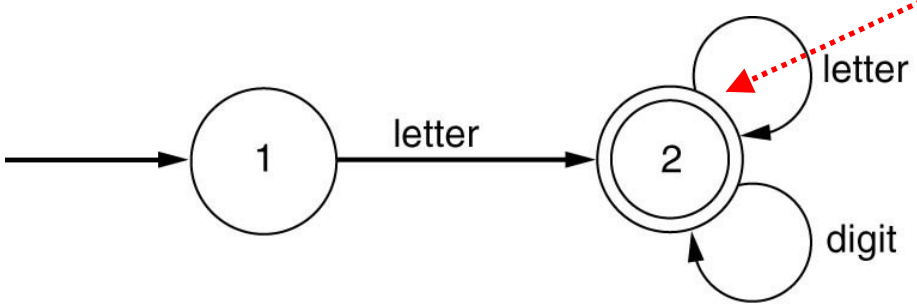
	letter	digit
1	2	
2	3	4
3	3	4
4	3	4

상태 최소화 : Example 9 (2/2)

The accepting states {2,3,4} cannot be distinguished.
→ *combine* these into one



	letter	digit
1	2	
2	3	4
3	3	4
4	3	4



Quiz #4

(a|b)* abb

(a | b)* abb 에 대한 NFA를 DFA로 변환하고, 상태 수를 최소화하시오.