

12. 오토마타

충북대학교

이재성





학습내용

- 유한 오토마타 정의
- 결정적 유한 오토마타(DFA)
- 비결정적 유한 오토마타(NFA)

- 정규표현을 FA로 변환하는 방법
- FA를 정규문법으로 변환하는 방법



인식기(Recognizer)

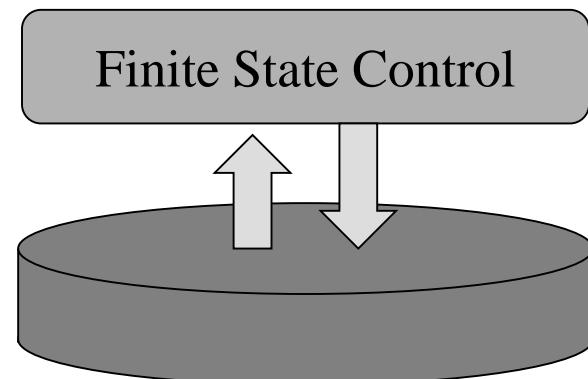
인식기

- 입력으로 스트링을 받아 스트링이 그 언어의 문장이면 “yes”, 아니면 “no”를 출력하는 프로그램

입력

$$a_0a_1a_2 \cdots a_i a_{i+1} a_{i+2} \cdots a_n$$

입력 헤드



12. 오토마타

• 종류

- 튜링 머신
- A에 선형 종속
- 푸쉬다운 오토마타
- 유한 오토마타**



유한 오토마타 (Finite Automata)

정의 : FA(Finite Automata)

알파벳 Σ 에 대한 유한 오토마타 $M = (Q, \Sigma, \delta, q_0, F)$

여기서, Q : 상태(state)들의 유한 집합;

Σ : 입력 알파벳의 유한 집합;

δ : 사상 함수;

$q_0 \in Q$: 시작 상태

$F \subseteq Q$: 종결 상태의 집합

사상함수 $\delta : Q \times \Sigma \rightarrow 2^Q$

i.e. $\delta(q, a) = \{p_1, p_2, \dots, p_n\}$

$G = (V_N, V_T, P, S)$

$re = \emptyset, \epsilon, a, +, \cdot, *$

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



결정적 유한 오토마타 (DFA)

■ 결정적

- $\delta(q, a)$ 가 한 상태만을 갖는 경우
- $\delta(q, a) = \{p\}$ 대신에 " $\delta(q, a) = p$ "로 표기
- $\delta(q, a)$ 가 항상 한 개의 다음 상태를 가질 때, M이 완전히 명시

■ δ 함수의 확장

- $Q \times \Sigma \Rightarrow Q \times \Sigma^*$
- $\delta(q, \varepsilon) = q$
 $\delta(q, ax) = \delta(\delta(q, a), x)$, 여기서 $x \in \Sigma^*$ 이고 $a \in \Sigma$.

■ 한 문장 x의 인식

- 만약 $\delta(q_0, x) = p$ 인 경우, p가 종결 상태에 속할 때 ($p \in F$).
- M에 의해 인식된 언어
 - $L(M) = \{ x \mid \delta(q_0, x) \in F \}$



DFA 예

ex) $M = (\{p, q, r\}, \{0, 1\}, \delta, p, \{r\})$

$$\delta : \delta(p, 0) = q \quad \delta(p, 1) = p$$

$$\delta(q, 0) = r \quad \delta(q, 1) = p$$

$$\delta(r, 0) = r \quad \delta(r, 1) = r$$

- $1001 \in L(M)$?

$$\delta(p, 1001) = \delta(p, 001) = \delta(q, 01) = \delta(r, 1) = r \in F.$$

$$\therefore 1001 \in L(M).$$

- $1010 \in L(M)$?

$$\delta(p, 1010) = \delta(p, 010) = \delta(q, 10) = \delta(p, 0) = q \notin F.$$

$$\therefore 1010 \notin L(M).$$

- 전이테이블: δ 를 matrix 형태로 표시. ex)

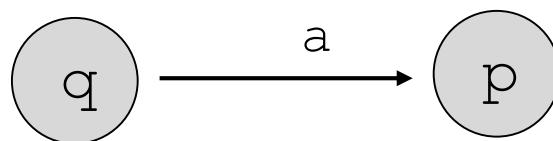
δ	Input symbols	
	0	1
p	q	p
q	r	p
r	r	r



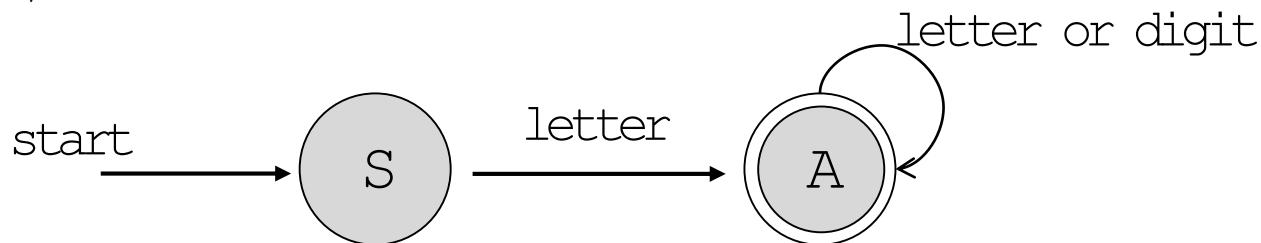
상태 전이도

정의

- 노드: 유한 오토마타의 각 상태를 표시(p, q)
- 전이 상태: 상태 q 에서 상태 p 로 가는 변화를 타나냄
- 레이블: 전이 때의 입력 문자 (a)
- $\delta(q, a) = p$



- 종결 상태는 이중 원, 시작 상태는 start 지시선으로 표시
인식기 예:





DFA 알고리즘

Algorithm : $w \in L(M)$ 결정.

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

begin

 currentstate := q_0 ; (* start state *)

 get(nextsymbol);

 while not eof do

 begin currentstate := $\delta(\text{currentstate}, \text{nextsymbol})$;
 get(nextsymbol)

 end;

 if currentstate in F then write('Valid String')

 else write('Invalid String');

end.

예:

$(p, 001) \rightarrow (q, 01) \rightarrow (r, 1) \rightarrow (r, \$)$

12. 오토마타

δ	Input symbols	
	0	1
p	q	p
q	r	p
r	r	r



DFA의 상태수 최소화

■ 정의

$w \in \Sigma^*$ 에 대해 q_1 에서 w 를 다 본 상태가 q_3 , q_2 에서 w 를 다 본 상태가 q_4 일 때,

q_3, q_4 중 하나만 종결 상태일 때, q_1 과 q_2 는 구별된다고 말함.

■ 동치 관계 구성

1. 전체 상태를 종결 상태와 미종결 상태로 구분
2. 같은 입력에 대해 서로 다른 동치류로 가면, 다른 분할을 하여 새로운 동치류를 만든다.
3. 위의 1, 2 단계를 분할이 더 발생하지 않을 때까지 반복



■ DFA에서 상태수를 줄이는 방법

<step 1> 모든 도달 불가능한 상태는 삭제

<step 2> 동치 관계를 만듦

<step 3> fa $M' = (Q', \Sigma, \delta', q_0', F')$ 를 만듦

(a) Q' : 동치류의 집합

p상태가 포함된 $[p]$ (집합) 형태의 동치류

(b) $q_0' = [q_0]$.

(c) $\delta(p, a) = q$ 이면 $\delta'([p], a) = [q]$

(d) $F' = \{[q] \mid q \in F\}$.



상태수 최소화 예(1)

ex) 유한 오토마トン $M = (\{A,B,C,D,E\}, \{a,b\}, \delta, A, \{C,E\})$ 에 의해 상태화된 언어를 위해 최소 상태 유한 오토마トン을 찾는다.

- δ 가 주어졌을 때

δ	a	b
A	B	D
B	B	C
C	D	E
D	D	E
E	B	C

- \equiv : $NF = \{A, B, D\}$, $F = \{C, E\}$



상태수 최소화 예(2)

ex) 유한 오토마トン $M = (\{A,B,C,D,E\}, \{a,b\}, \delta, A, \{C,E\})$ 에 의해 상태화된 언어를 위해 최소 상태 유한 오토마トン을 찾는다.

- $\delta(A, a) = B(\text{NF}), \delta(A, b) = D(\text{NF})$

- $\delta(B, a) = B(\text{NF}), \delta(B, b) = C(\text{F})$

- $\delta(D, a) = D(\text{NF}), \delta(D, b) = E(\text{F})$

- $\delta(C, a) = D(\text{NF}), \delta(C, b) = E(\text{F})$

- $\delta(E, a) = B(\text{NF}), \delta(E, b) = C(\text{F})$

δ	a	b
A	B	D
B	B	C
C	D	E
D	D	E
E	B	C

$\square \equiv : \{A\}, \{B, D\}, \{C, E\}$

δ'	a	b
[A]=p	q	q
[B,D]=q	q	r
[C,E]=r	q	r



비결정적 유한 오토마타 (NFA)

■ 비결정적

- 상태 q 에서 입력 심벌 a 에 대해 갈 수 있는 상태가 여러 개
- $\delta(q,a) = \{p_1, p_2, \dots, p_n\}$

■ 예

- $M = (\{q_0, q_1, q_2, q_3, q_f\}, \{0,1\}, \delta, q_0, \{q_f\})$
 - if $\delta(q,a) = \emptyset$, then $\delta(q,a)$ is undefined.

δ	0	1
q_0	$\{q_1, q_2\}$	$\{q_1, q_3\}$
q_1	$\{q_1, q_2\}$	$\{q_1, q_3\}$
q_2	$\{q_f\}$	\emptyset
q_3	\emptyset	$\{q_f\}$
q_4	$\{q_f\}$	$\{q_f\}$



-
- 정의 : $\delta(q_0, x)$ 의 상태 p 중 F의 상태를 포함하는 경우, 문장 x는 M에 의해 인식된다.

ex) $1011 \in L(M)$?

$$\begin{aligned}\delta(\{q_0\}, 1011) &= \delta(\{q_1, q_3\}, 011) = \delta(\{q_1, q_2\}, 11) \\ &= \delta(\{q_1, q_3\}, 1) = \{q_1, q_3, q_f\} \\ \therefore 1011 \in L(M) \quad (\because \{q_1, q_3, q_f\} \cap \{q_f\} \neq \emptyset)\end{aligned}$$

ex) $0100 \in L(M)$?

δ	0	1
q_0	$\{q_1, q_2\}$	$\{q_1, q_3\}$
q_1	$\{q_1, q_2\}$	$\{q_1, q_3\}$
q_2	$\{q_f\}$	\emptyset
q_3	\emptyset	$\{q_f\}$
q_4	$\{q_f\}$	$\{q_f\}$

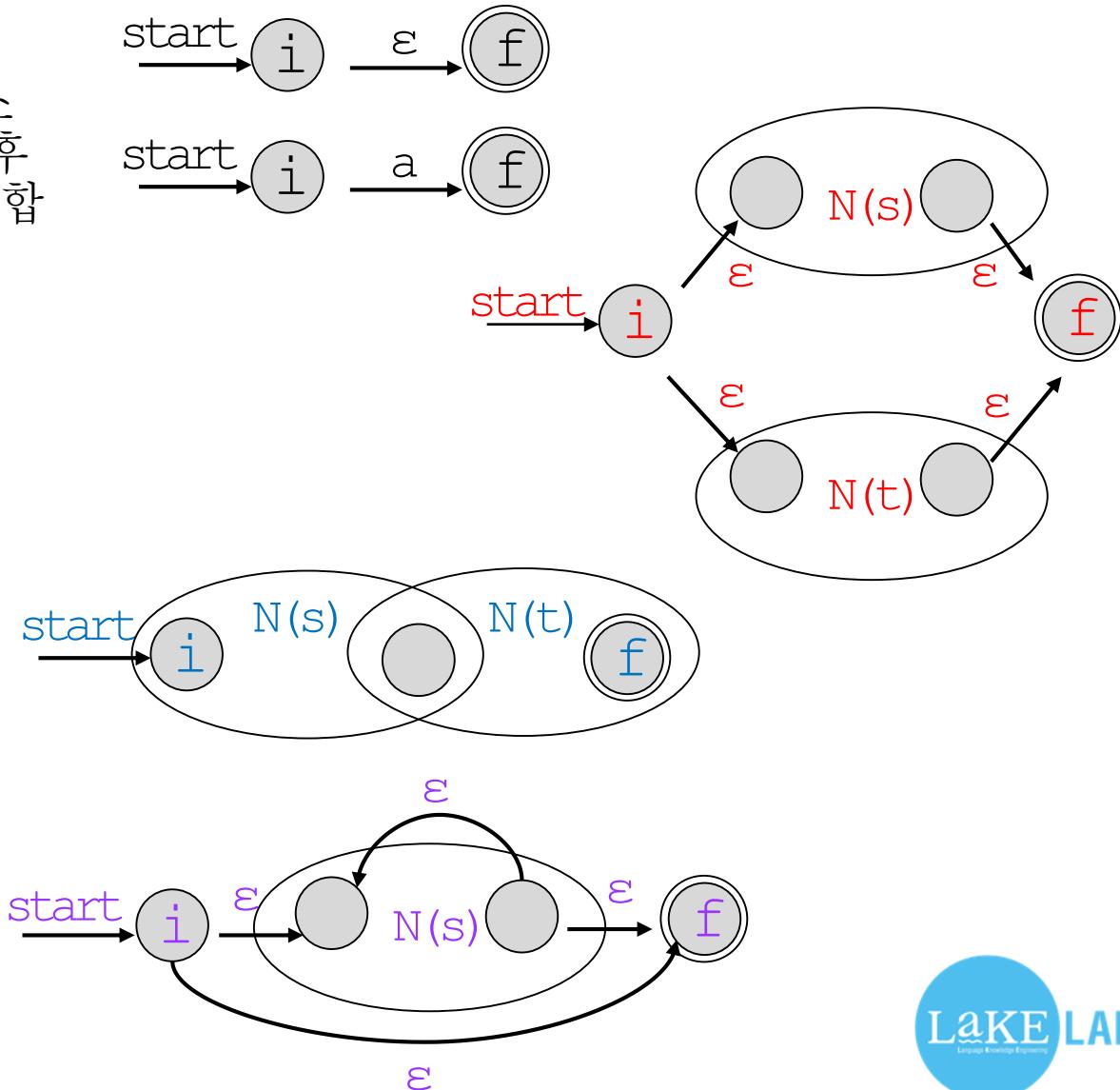


정규표현의 NFA로의 변환

구성요소 분해

- 정규표현을 구성요소 부분식으로 분리한 후 다시 규칙에 따라 조합

- ϵ
- a
- $s \mid t \Rightarrow N(s \mid t)$
- $st \Rightarrow N(st)$
- $s^* \Rightarrow N(s^*)$
- $(s) \Rightarrow N(s)$

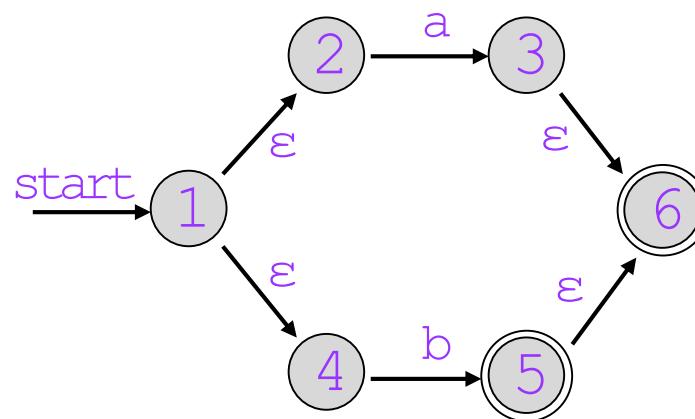
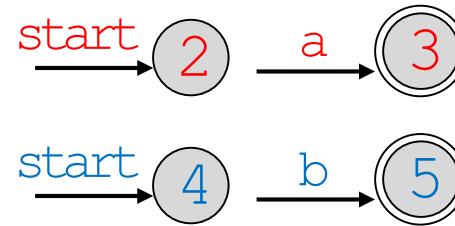




정규표현의 NFA로의 변환 예(1)

- 정규표현 $r=(a|b)^*abb$ 의 변환순서

1. $r1=a$
2. $r2=b$
3. $r3= r1 | r2$
4. $r4= (r3)$
5. $r5= (r3) ^*$
6. $r6=a$
7. $r7=r5 \ r6$
8. :

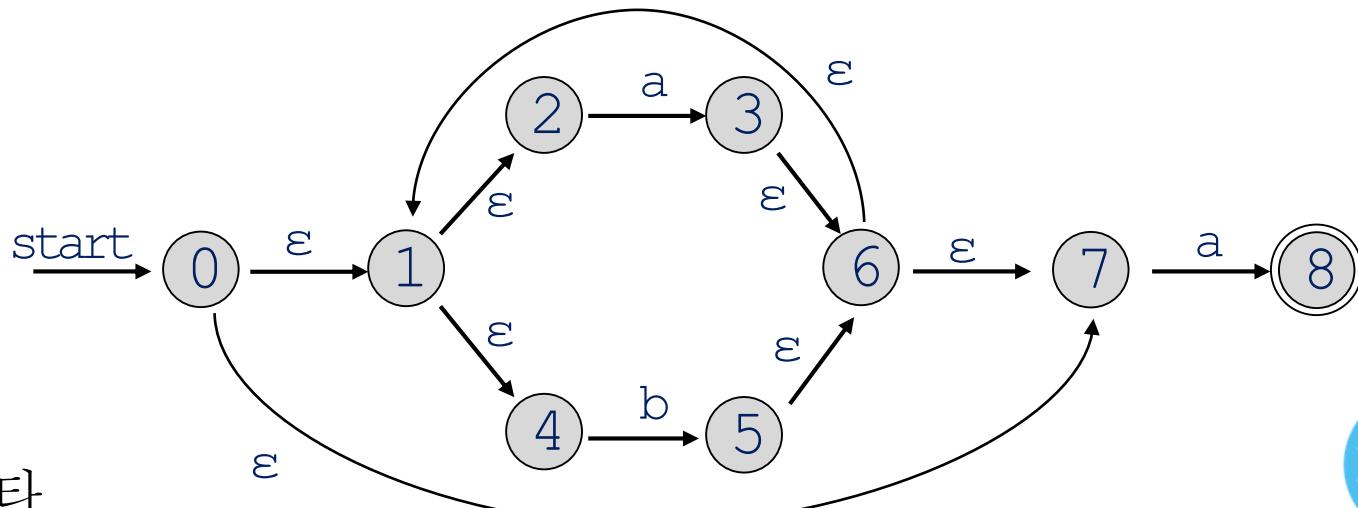
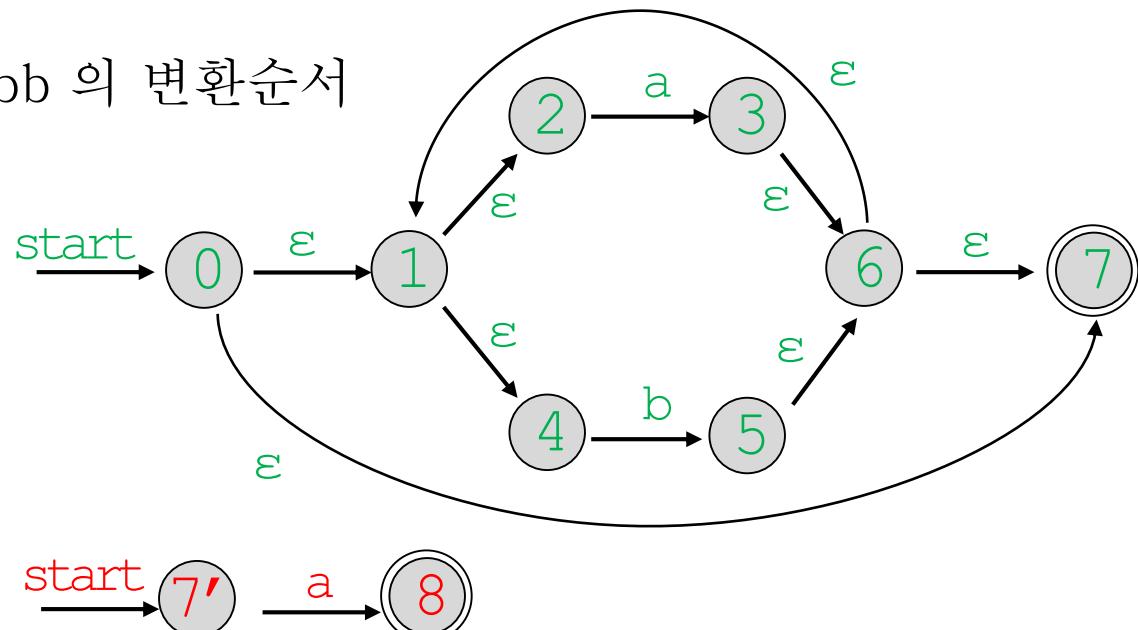




정규표현의 NFA로의 변환 예(2)

- 정규표현 $r=(a|b)^*abb$ 의 변환순서

1. $r_1 = a$
2. $r_2 = b$
3. $r_3 = r_1 | r_2$
4. $r_4 = (r_3)$
5. $r_5 = (r_3)^*$
6. $r_6 = a$
7. $r_7 = r_5 \ r_6$
8. :





NFA를 DFA로 전환

필요성

- NFA에서 ϵ 전이에 대해 컴퓨터 처리가 어려움

부분집합 구성 알고리즘 용어

- ϵ -closure(s):

- NFA상태 s에서 ϵ 전이만으로 도달 가능한 상태집합

- ϵ -closure(T):

- T에 포함된 어떤 상태 s에서 ϵ 전이만으로 도달가능한 상태 집합

- move(T,a):

- T에 포함되는 어떤 상태s에서 입력기호 a에 의해 전이되는 상태집합

- Dtran:

- D에 대한 전이 테이블

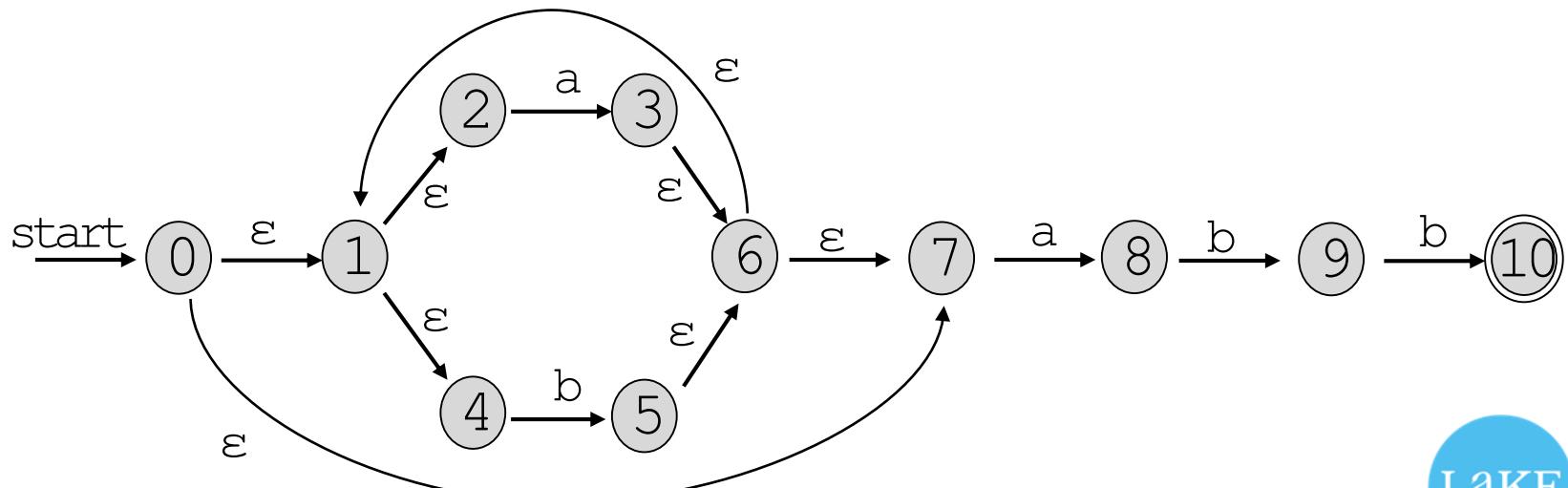
- Dstates:

- D의 상태들의 집합



ϵ -closure 및 move 예

- ϵ -closure(0)= $\{0,1,2,4,7\}$
- ϵ -closure($\{3,8\}$) = $\{1,2,3,4,6,7\} \cup \{8\} = \{1,2,3,4,6,7,8\}$
- move($1, a$)= { 3 }
- move($\{3,8\}, b$) = $\{5\} \cup \{9\} = \{5, 9\}$

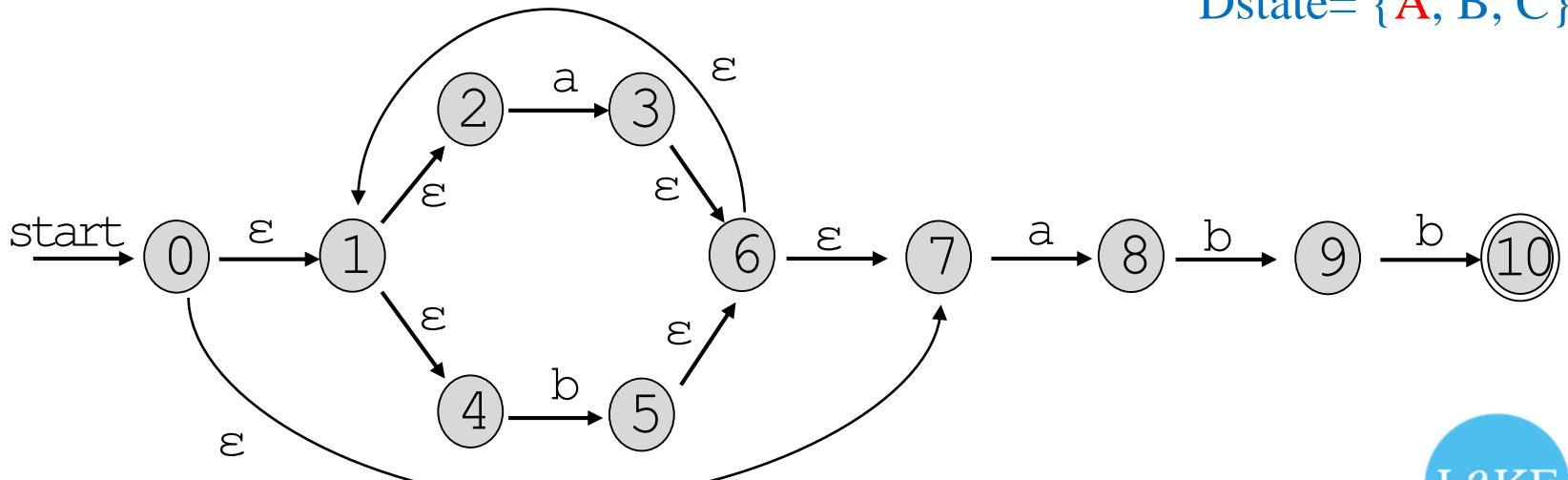


12. 오토마타



부분집합 구성 알고리즘 예(1)

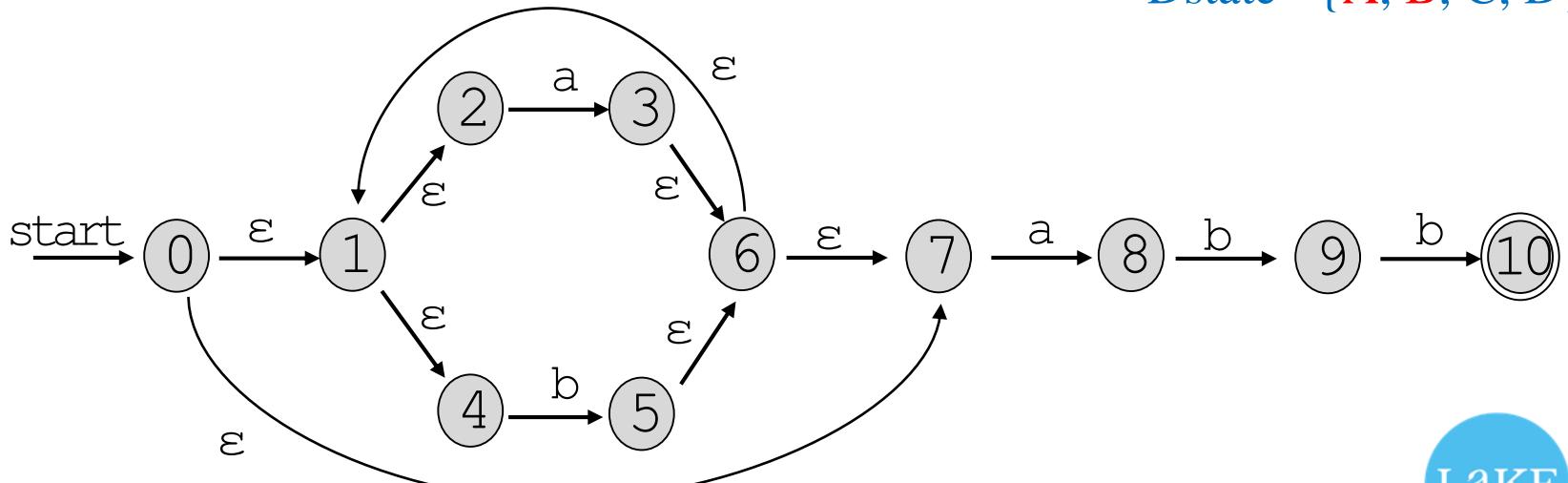
- 시작상태 ϵ -closure(0) $A = \{0, 1, 2, 4, 7\}$ $Dstate = \{A\}$
 - 입력 알파벳 {a, b}
- ϵ -closure(move(A, a)) = ϵ -closure(move({0,1,2,4,7}, a))
= ϵ -closure({3,8}) = {1,2,3,4,6,7,8} $\rightarrow B$
 - $Dtran[A, a] = B$ $Dstate = \{A, B\}$
- ϵ -closure(move(A, b)) = ϵ -closure(move({0,1,2,4,7}, b))
= ϵ -closure({5}) = {1,2,4,5,6,7} $\rightarrow C$
 - $Dtran[A, b] = C$ $Dstate = \{A, B, C\}$





부분집합 구성 알고리즘 예(2)

- 현재: $A=\{0,1,2,4,7\}$, $B=\{1,2,3,4,6,7,8\}$, $C=\{1,2,4,5,6,7\}$
- ϵ -closure(move(B,a))= ϵ -closure(move({1,2,3,4,6,7,8}, a)) = ϵ -closure({3,8}) = {1,2,3,4,6,7,8} \rightarrow B
 - Dtran[B,a] = B Dstate= {A, B, C}
- ϵ -closure(move(B,b))= ϵ -closure(move({1,2,3,4,6,7,8}, b)) = ϵ -closure({5,9}) = {1,2,4,5,6,7,9} \rightarrow D
 - Dtran[B,b] = D Dstate= {A, B, C, D}





부분집합 구성 알고리즘 예(3)

■ 최종 구성된 부분집합

- $A = \{0, 1, 2, 4, 7\}$ $B = \{1, 2, 3, 4, 6, 7, 8\}$ $C = \{1, 2, 4, 5, 6, 7\}$
- $D = \{1, 2, 4, 5, 6, 7, 9\}$ $E = \{1, 2, 4, 5, 6, 7, 10\}$

$$D_{\text{tran}}[A, a] = B$$

$$D_{\text{tran}}[A, b] = C$$

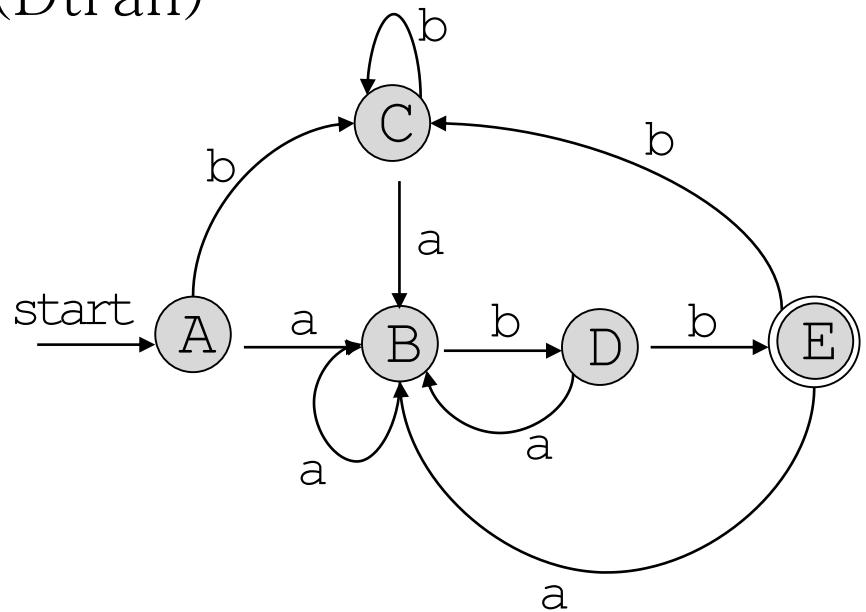
$$D_{\text{tran}}[B, a] = B$$

$$D_{\text{tran}}[B, b] = D$$

:

■ DFA를 위한 전이 테이블(Dtran)

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





ϵ -closure의 계산

T안의 상태를 모두 스택에 푸시

ϵ -closure(T)를 T로 초기 설정

while 스택이 비어있지 않음 do

begin

스택에서 t를 pop

for t에서 ϵ 레이블로 연결 가능한 상태 u do

if u가 ϵ -closure(T)안에 없음 then do

begin

u를 ϵ -closure(T)에 추가

u를 스택에 push

end

end



부분집합 구성 알고리즘

- 입력: NFA N
- 출력: 동일한 언어를 수용하는 하나의 DFA D

- 방법

상태 ϵ -closure(s_0)만을 Dstates에 넣는다.

while Dstates 안에 마크가 안된 상태 T가 있다 do begin

 mark T;

 for 각 입력기호 a do begin

 U := ϵ -closure(move(T,a));

 if U가 Dstates안에 없다 then

 U를 마크하지 않고 Dstates에 추가

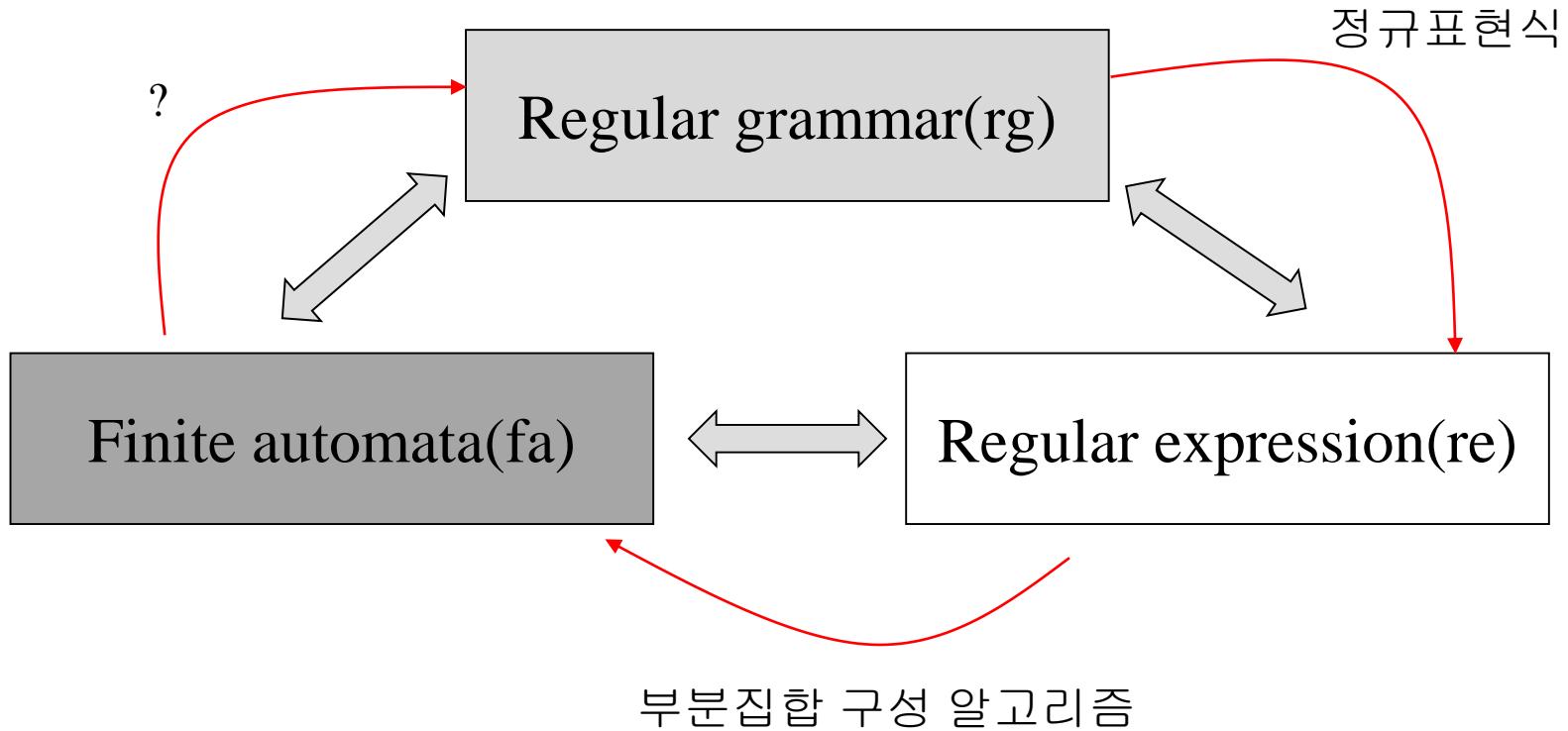
 Dtran[T, a] := U

 end

end



정규 언어의 변환 관계





유한 오토마타와 정규 표현

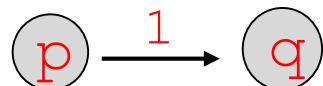
■ 2단계 접근

- 유한 오토마타에서 정규 문법 구하기
- 정규 문법에서 정규 표현 구하기 (정규 표현식 이용)

■ 유한 오토마타에서 정규 문법 만들기

- 오토마타의 상태에 입력 문자를 추가한 형태의 문법 기호로 표기

- $p \rightarrow 1q$



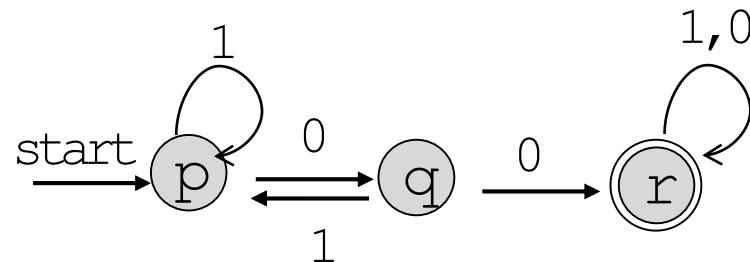
- $p \rightarrow \epsilon$





DFA에서 RG로 변환 예

DFA



대응 정규 문법

- 모든 상태: $Q = \{p, q, r\}$
- 모든 입력 문자: $\Sigma = \{0, 1\}$
- 시작 기호: p
- $G = (V_N, V_T, P, S)$
- $V_N = \{p, q, r\}, V_T = \{0, 1\}, S = p$
- $P:$
 $p \rightarrow 0q, p \rightarrow 1p$
 $q \rightarrow 0r, q \rightarrow 1p$
 $r \rightarrow 0r, r \rightarrow 1r, r \rightarrow \epsilon$



RG에서 RE로 변환 예

■ RG

- P: $p \rightarrow 0q, p \rightarrow 1p$
 $q \rightarrow 0r, q \rightarrow 1p$
 $r \rightarrow 0r, r \rightarrow 1r, r \rightarrow \epsilon$
- $p = 0q + 1p$
- $q = 0r + 1p$
- $r = 0r + 1r + \epsilon$

$$r = (0+1)r + \epsilon \Rightarrow r = (0+1)^*$$

$$q = 0r + 1p = 0(0+1)^* + 1p$$

$$\begin{aligned} p &= 0q + 1p = 0(0(0+1)^* + 1p) + 1p \\ &= 00(0+1)^* + 01p + 1p \end{aligned}$$

$$= (01+1)p + 00(0+1)^*$$

$$L(M) = (01+1)^*00(0+1)^*$$



참고 문헌

- [1] Alfred V. Aho, Ravi Sethi, Jeffrey D. Ullman, “Compilers – Principles, Techniques, and Tools,” Bell Telephone Laboratories, Incorporated, 1986.
- [2] 오세만, “컴파일러 입문”, 정의사, 2004.