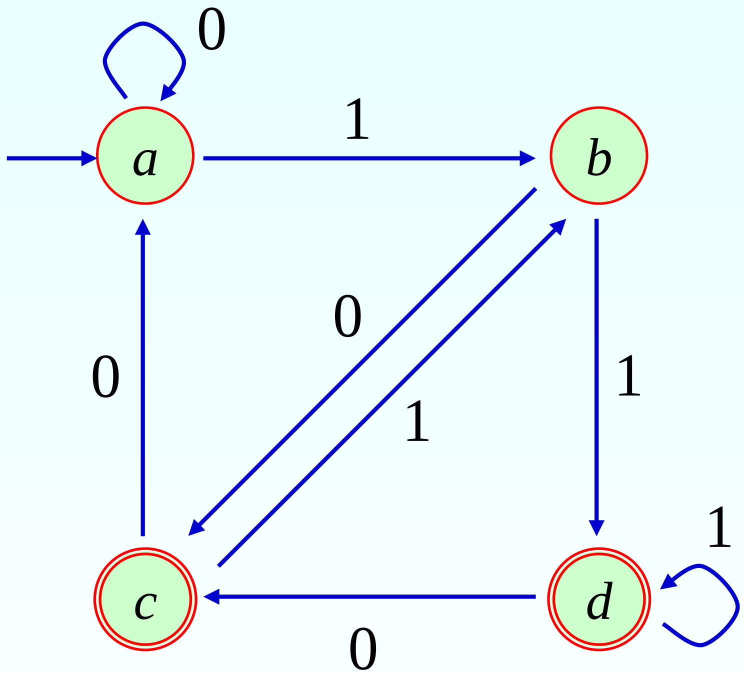
Non-deterministic Finite Automata

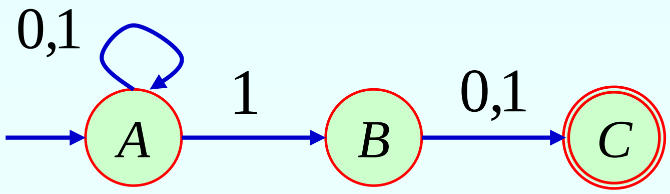
부산대학교
PUSAN NATIONAL UNIVERSITY

Finite Automata

- Deterministic FA (DFA)
  - Every state has one transition on each input character
- Non-deterministic FA (NFA)
  - A state may have more than one transition on an input character
  - The NFA allows λ-transitions
  - The NFA accepts a string if there is at least ONE path from the start state to an accepting state whose edge labels spell out the string

Examples (DFA vs. NFA)





Input String **11010**

$a \rightarrow b \rightarrow d \rightarrow c \rightarrow b \rightarrow c$

Reachable States

$\{A\}$; start state

$\{A, B\}$ $\leftarrow$ **1**

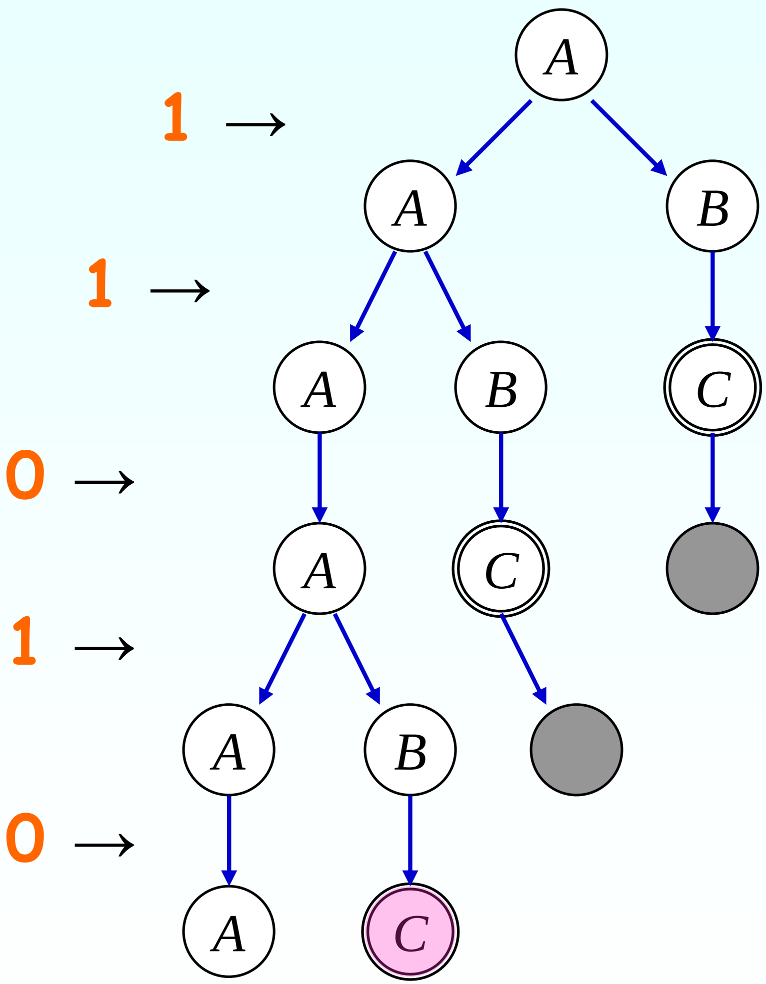
$\{A, B, C\}$ $\leftarrow$ **1**

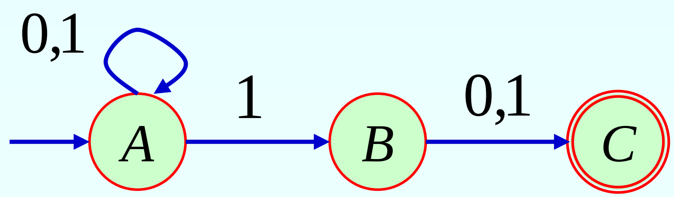
$\{A, C\}$ $\leftarrow$ **0**

$\{A, B\}$ $\leftarrow$ **1**

$\{A, C\}$ $\leftarrow$ **0**

Examples (DFA vs. NFA)





Input String **11010**

$\{A\}$	; start state	
$\{A, B\}$	$\leftarrow$	1
$\{A, B, C\}$	$\leftarrow$	1
$\{A, C\}$	$\leftarrow$	0
$\{A, B\}$	$\leftarrow$	1
$\{A, C\}$	$\leftarrow$	0

NFA

Definition

A non-deterministic finite automata (NFA) N is specified by a quintuple $N = (Q, \Sigma, \Delta, q_0, F)$, where

Q : an alphabet of state symbols ;

Σ : an alphabet of input symbols ;

Δ : a subset of transition relation

$(Q \times (\Sigma \cup \{\lambda\})) \times Q$;

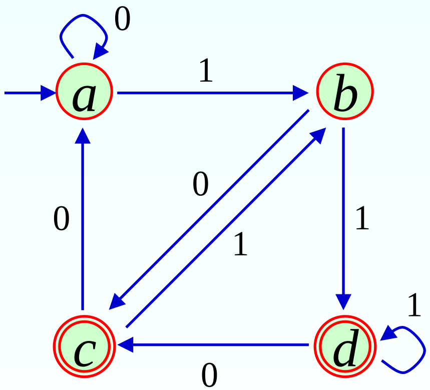
$q_0 \in Q$ is the start state ; and

$F \subseteq Q$ is a set of final states.

(note) δ : transition function in DFA

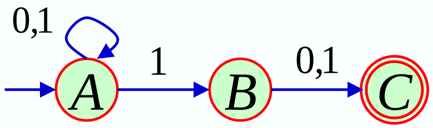
(Note) Relation vs. Function

For nonempty sets A, B, a *function* $f : A \rightarrow B$ is a *relation* from A to B in which *every element of* A *appears exactly once as the first component* of an ordered pair in the relation



Function δ : $Q \times \Sigma \rightarrow Q$

Relation Δ : a subset of $(Q \times \Sigma) \times Q$



$((A,0), A)$,
$((A,1), A)$, $((A,1), B)$,
$((B,0), C)$,
$((B,1), C)$

$((a,0), a)$, $((a,1), b)$,
$((b,0), c)$, $((b,1), d)$, …

An Example

(Ex) (010+01)* string을 accept하는 NFA?

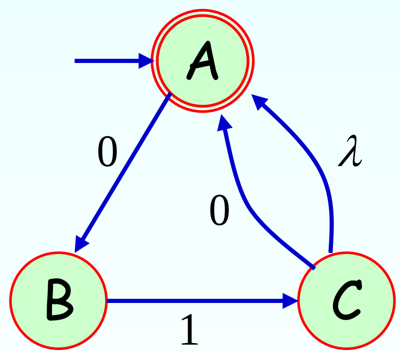
N = ({A,B,C}, {0,1}, Δ, A, {A})

(A,0) Δ B

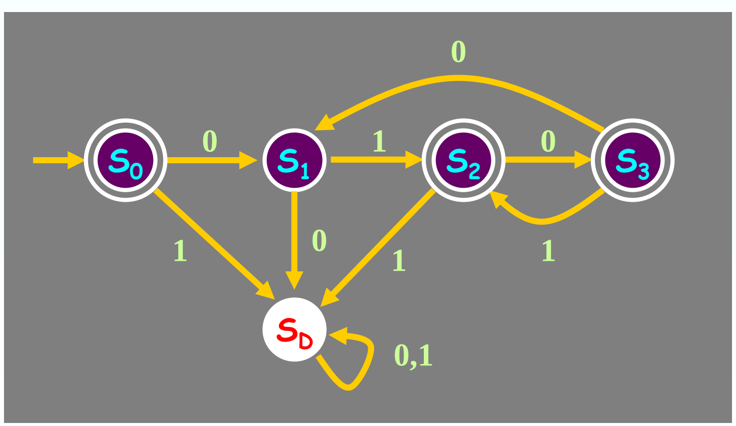
(B,1) Δ C

(C,0) Δ A

(C,λ) Δ A



$\{A\}$	; start state
$\{B\}$	$\leftarrow$ 0
$\{C\} \cup \{A\}$	$\leftarrow$ 1
$\{A\} \cup \{B\}$	$\leftarrow$ 0
$\{B\}$	$\leftarrow$ 0
$\{C\} \cup \{A\}$	$\leftarrow$ 1



01001

Δ_f : Transition function for NFAs

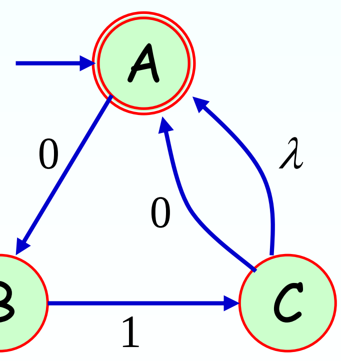
- **Remarks**

  It is sometimes convenient to consider the transition relation Δ as a function

$$\Delta_f : Q \times (\Sigma \cup \{\lambda\}) \rightarrow 2^Q$$

where $2^Q \equiv \wp(Q)$ is the power set of Q.

(Ex) (010+01)* NFA?



Δ_f	Input		
	0	1	λ
A	$\{B\}$	ϕ	ϕ
B	ϕ	$\{C\}$	ϕ
C	$\{A\}$	ϕ	$\{A\}$

Example

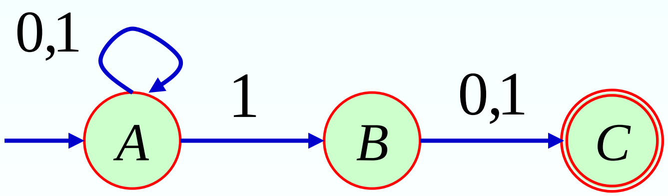
(Ex) (0+1)*1(0+1) string을 accept하는 NFA?

N = ({A,B,C}, {0,1}, Δ_f , A, {C})

Δ_f (A,0) = {A}, Δ_f (A,1) = {A,B}

Δ_f (B,0) = {C}, Δ_f (B,1) = {C}

Δ_f (*,λ) = {*}, The others Δ_f (*,*) = ϕ



Δ_f	Input		
	0	1	λ
A	$\{A\}$	$\{A,B\}$	ϕ
B	$\{C\}$	$\{C\}$	ϕ
C	ϕ	ϕ	ϕ

Δ_S : Extended transition func. to a set

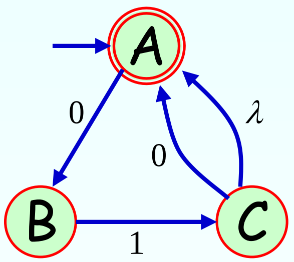
- Definition of Δ_s

$$\Delta_s : 2^Q \times (\Sigma \cup \{\lambda\}) \rightarrow 2^Q$$

$$\Delta_s(P, a) = \bigcup_{q \in P} \Delta_f(q, a)$$

where $P \subseteq Q$ and $a \in (\Sigma \cup \{\lambda\})$.

$(P \in 2^Q)$



Δ_f	Input		
	0	1	λ
A	$\{B\}$	ϕ	ϕ
B	ϕ	$\{C\}$	ϕ
C	$\{A\}$	ϕ	$\{A\}$

$\Delta_s(\{A\},0) = \{B\}$ $\quad\Delta_s(\{B\},1) = \{C\}$

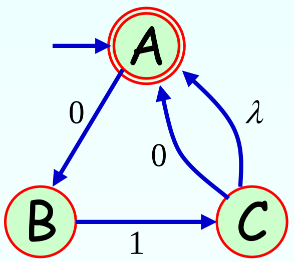
$\Delta_s(\{C\},0) = \{A\}$ $\quad\Delta_s(\{C\},\lambda) = \{C\}$

$\Delta_s(\{A,C\},0) = \{B\} \cup \{A\} = \{A,B\}$

$\Delta_s(\{A,C\},1) = \phi \cup \phi = \phi = \{\ \}$

Δ_s for a Set

Δ_S	Input		
	0	1	λ
ϕ	ϕ	ϕ	ϕ
$\{A\}$	$\{B\}$	ϕ	ϕ
$\{B\}$	ϕ	$\{C\}$	ϕ
$\{C\}$	$\{A\}$	ϕ	$\{A\}$
$\{A,B\}$	$\{B\}$	$\{C\}$	ϕ
$\{B,C\}$	$\{A\}$	$\{C\}$	$\{A\}$
$\{A,C\}$	$\{A,B\}$	ϕ	$\{A\}$
$\{A,B,C\}$	$\{A,B\}$	$\{C\}$	$\{A\}$



Δ_f	Input		
	0	1	λ
A	$\{B\}$	ϕ	ϕ
B	ϕ	$\{C\}$	ϕ
C	$\{A\}$	ϕ	$\{A\}$

E, E_s : λ-Closure

- Definition of $E(q)$ and $E_s(P)$

  $E(q)$: a set of states that can be reachable from q without reading any input symbol (through λ)

  $E_s(P)$: $\bigcup_{q \in P} E(q)$

  where $P \subseteq Q$.

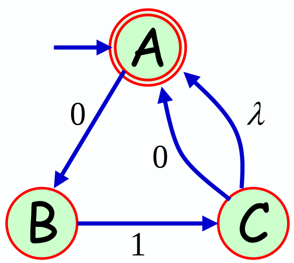
$E(A) = \{A\}$, $E(B) = \{B\}$

$E(C) = \{A, C\}$

$E_s(\{A,B\}) = E(A) \cup E(B) = \{A,B\}$

$E_s(\{B,C\}) = E(B) \cup E(C)$

$= \{A,B,C\}$

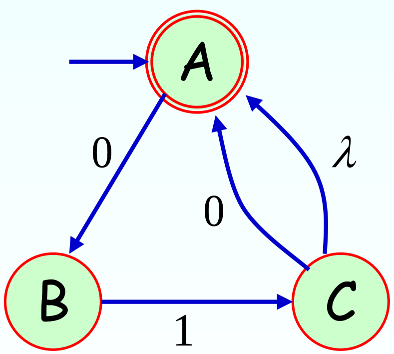


Δ_f	Input		
	0	1	λ
A	$\{B\}$	ϕ	ϕ
B	ϕ	$\{C\}$	ϕ
C	$\{A\}$	ϕ	$\{A\}$

NFA Operation ?

(Ex) (010+01)* string을 accept하는 NFA?

N = ({A,B,C}, {0,1}, Δ, A, {A})



Δ_f	Input		
	0	1	λ
A	{*B*}	ϕ	ϕ
B	ϕ	{*C*}	ϕ
C	{*A*}	ϕ	{*A*}

E(*A*) = {*A*} ; E(start state)

$E_s(\Delta_s(\{A\},0))$

$= E_s(\{B\}) = \{B\}$

$E_s(\Delta_s(\{B\},1))$

$= E_s(\{C\}) = \{A,C\}$

$E_s(\Delta_s(\{A,C\},0))$

$= E_s(\{A,B\}) = \{A,B\}$

$E_s(\Delta_s(\{A,B\},0))$

$= E_s(\{B\}) = \{B\}$

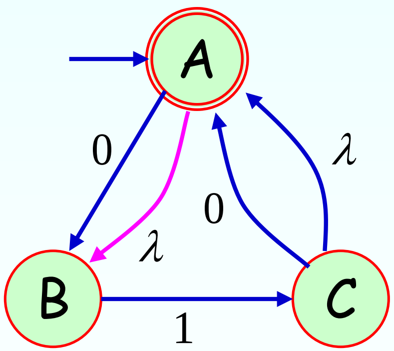
$E_s(\Delta_s(\{B\},1))$

$= E_s(\{C\}) = \{A,C\}$

NFA Operation ?

(Ex) ? string을 accept하는 NFA

N = ({A,B,C}, {0,1}, Δ, A, {A})



Δ_f	Input		
	0	1	λ
A	$\{B\}$	ϕ	$\{B\}$
B	ϕ	$\{C\}$	ϕ
C	$\{A\}$	ϕ	$\{A\}$

E(A) = {A,B}, E(B) = {B}

E(C) = {A,B,C}

E_s({B,C}) = E(B) ∪ E(C) = {A,B,C}

$E(A) = \{A,B\}$; E(start state)

$E_s(\Delta_s(\{A,B\},0))$

$= E_s(\{B\}) = \{B\}$

$E_s(\Delta_s(\{B\},1))$

$= E_s(\{C\}) = \{A,B,C\}$

$E_s(\Delta_s(\{A,B,C\},0))$

$= E_s(\{A,B\}) = \{A,B\}$

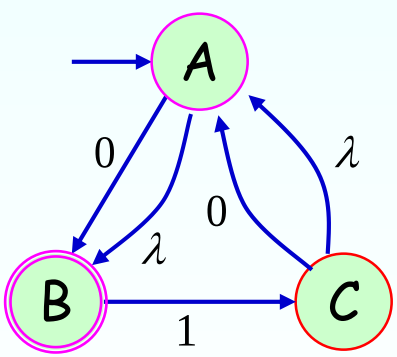
$E_s(\Delta_s(\{A,B\},0))$

$= E_s(\{B\}) = \{B\}$

NFA Operation ?

(Ex) ? string을 accept하는 NFA

N = ({A,B,C}, {0,1}, Δ, A, {A})



Δ_f	Input		
	0	1	λ
A	$\{B\}$	ϕ	$\{B\}$
B	ϕ	$\{C\}$	ϕ
C	$\{A\}$	ϕ	$\{A\}$

$E(A) = \{A,B\}$; E(start state)

$E_s(\Delta_s(\{A,B\},0))$

$= E_s(\{B\}) = \{B\}$

$E_s(\Delta_s(\{B\},1))$

$= E_s(\{C\}) = \{A,B,C\}$

$E_s(\Delta_s(\{A,B,C\},0))$

$= E_s(\{A,B\}) = \{A,B\}$

$E_s(\Delta_s(\{A,B\},0))$

$= E_s(\{B\}) = \{B\}$

Δ^* : Reachable State Function

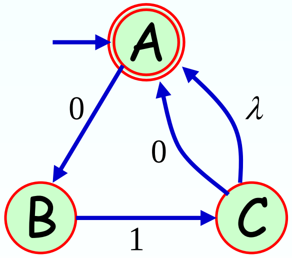
Definition of Δ^*

$$\Delta^* : Q \times \Sigma^* \rightarrow 2^Q$$

(i) $\Delta^*(q,\lambda) = E(q)$

(ii) $\Delta^*(q,wa) = E_s(\Delta_s(\Delta^*(q,w),a))$, $w \in \Sigma^*$, $a \in \Sigma$

(Ex.) NFA accepting $(010+01)^*$



$\Delta^*(A,0) = E_s(\Delta_s(\Delta^*(A,\lambda),0)) = E_s(\Delta_s(E(A),0))$
$= E_s(\Delta_s(\{A\},0)) = E_s(\{B\}) = \{B\}$

$\Delta^*(A,01) = E_s(\Delta_s(\Delta^*(A,0),1)) = E_s(\Delta_s(\{B\},1))$
$= E_s(\{A,C\}) = \{A,C\}$

$\Delta_s(P,a) = \cup_{q\in P} \Delta_f(q,a)$

NFA Language

- Definition of NFA Languages

  A string w is said to be **accepted** by a NFA N if $\Delta^*(q_0, w)$ contains one or more final states.

  A set of all the strings accepted by N is called the **language** of N, $L(N)$.

$$L(N) = \{ w \in \Sigma^* \mid \Delta^*(q_0, w) \cap F \neq \phi \}$$

q_0 : Start state

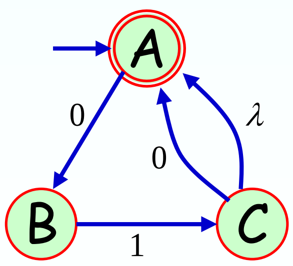
$\Delta^*(A,0) = \{B\}$

$\Delta^*(A,01) = \{A,C\}$

$\Delta^*(A,010) = \{A,B\}$

$\Delta^*(A,0100) = \{B\}$

$\Delta^*(A,01001) = \{A,C\}$



$\Delta^*(A,1)$

$= E_s(\Delta_s(\Delta^*(A,\lambda),1))$

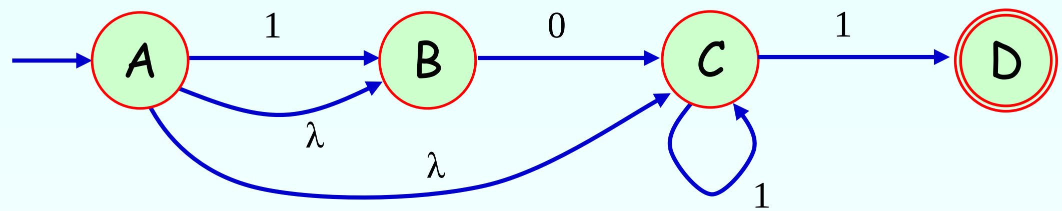
$= E_s(\Delta_s(\{A\},1))$

$= E_s(\phi) = \phi$

Another Example of NFA

?

string을 accept 하는 NFA



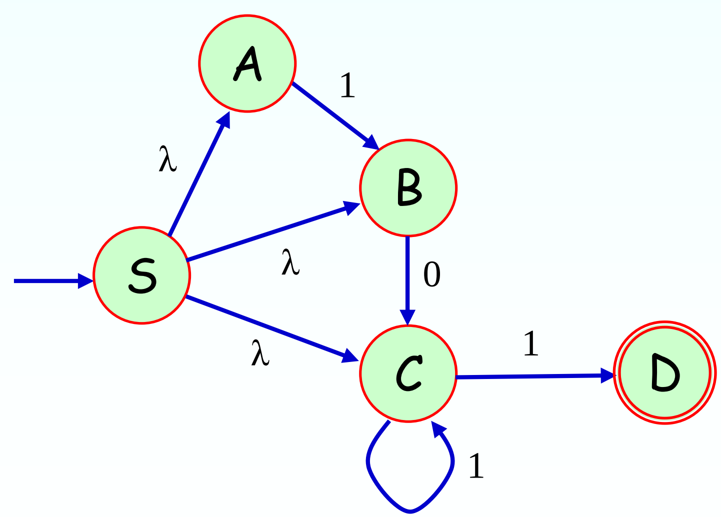
$\Delta^*(A,\lambda) = \{A,B,C\}$

$\Delta^*(A,0) = \{C\}$

$\Delta^*(A,01) = \{C,D\}$

$\Delta^*(A,011) = \{C,D\}$

$\Delta^*(A,0110) = \{\ \}$



$\Delta^*(S,\lambda) = \{S,A,B,C\}$

$\Delta^*(S,1) = \{B,C,D\}$

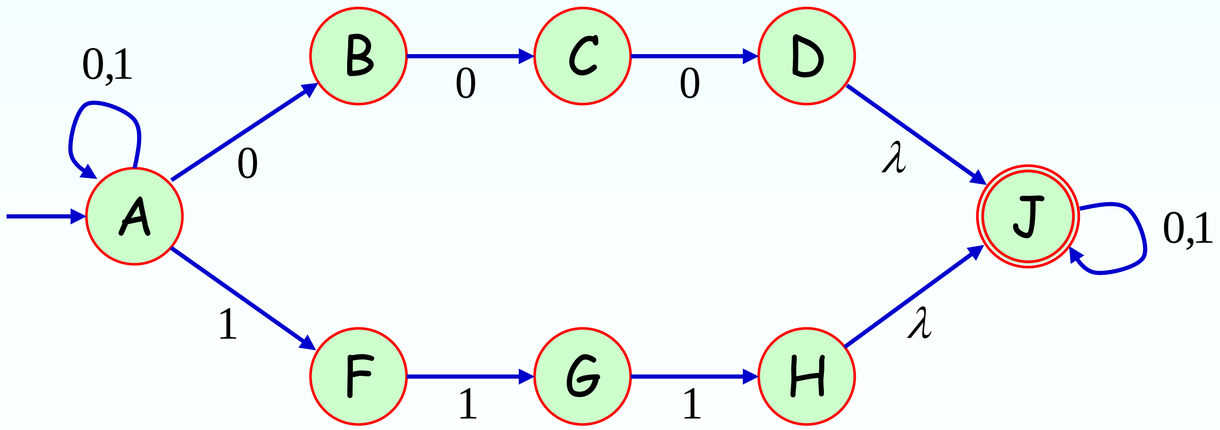
$\Delta^*(S,10) = \{C\}$

$\Delta^*(S,101) = \{C,D\}$

NFA Design

- Design of NFA is generally easier than design of DFA for a given language

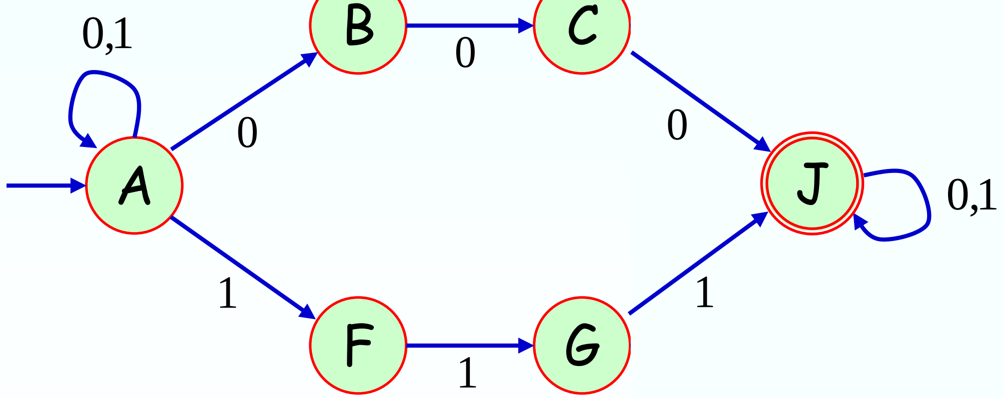
(Ex) 0 또는 1이 연이어 세 번 나오는 substring을 갖는 string들의 집합인 언어



NFA Design

- Design of NFA is generally easier than design of DFA for a given language

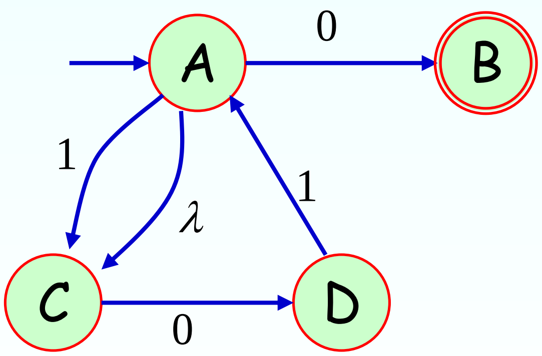
(Ex) 0 또는 1이 연이어 세 번 나오는 substring을 갖는 string들의 집합인 언어

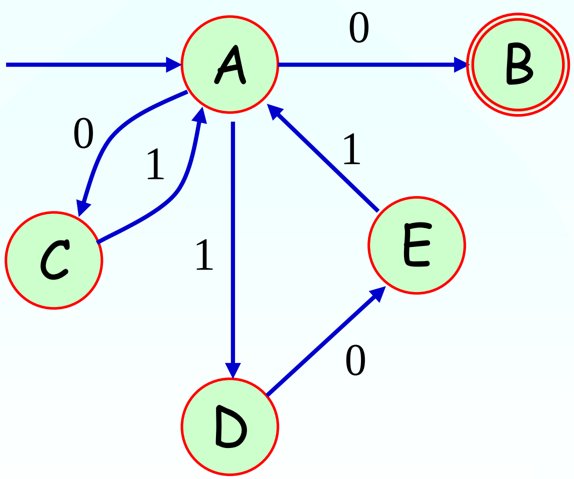


Another example of NFA design

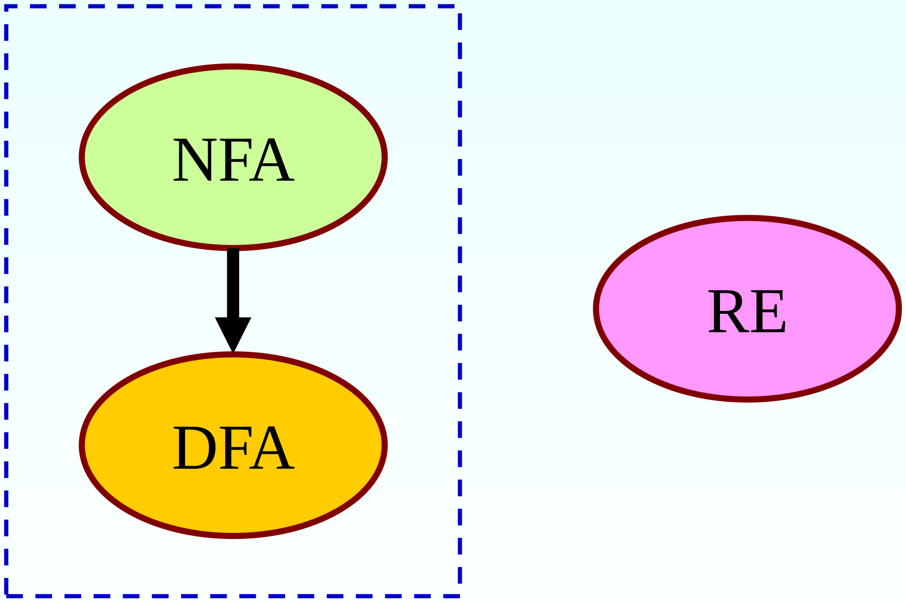
(Ex) Σ = {0,1}상의 정규식 (01+101)*0으로 표현되는 언어를 accept하는 NFA ?

$((\lambda+1)01)^*0$





Finite Automata vs. Regular Expression



Finite Automata

DFA from NFA

- Set of $L(M_{\text{DFA}})$'s $\subseteq$ Set of $L(M_{\text{NFA}})$'s
  - Because DFA is a kind of NFA.

- Theorem 1

  There exists a DFA for a given NFA such that $L(M_{\text{DFA}}) = L(M_{\text{NFA}})$.

  (Proof)

  (a) Show a DFA-derivation method from a NFA.

  (b) Prove that the language of the DFA is equivalent to that of the NFA.

DFA-derivation from NFA

Let $\boldsymbol{D} = (Q_D, \Sigma, \delta, \boldsymbol{q'}, \boldsymbol{F_D})$ be the derived DFA from an NFA $\boldsymbol{N} = (Q_N, \Sigma, \Delta, \boldsymbol{q_0}, \boldsymbol{F_N})$.

$q' = E(q_0)$; $Q_D \leftarrow q'$; mark q';

For a marked q_P ($= P \in 2^{Q_N}$) $\in Q_D$,

? ? ? ? ?

If $q_R \notin Q_D$, then $Q_D \leftarrow q_R$, mark q_R, $\delta(q_P, a) = q_R$;

else $\delta(q_P, a) = q_R$ ($\in Q_D$);

$F_D = \{ q_E \mid q_E (= E) \in Q_D$ and $(E \cap F_N) \neq \phi \}$;

An Example

$S_0 = E(A) = \{A\}$

$E_s(\Delta_s(S_0,0)) = \{B\} = S_1$

$E_s(\Delta_s(S_0,1)) = \{\ \} = S_2$

$E_s(\Delta_s(S_1,0)) = \{\ \} = S_2$

$E_s(\Delta_s(S_1,1)) = \{A,C\} = S_3$

$E_s(\Delta_s(S_2,0)) = \{\ \} = S_2$

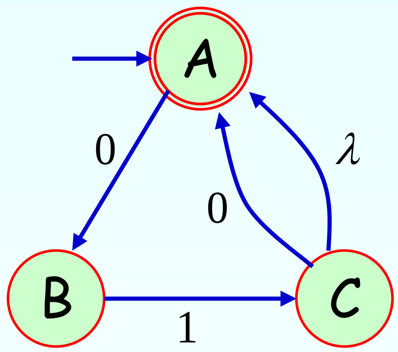
$E_s(\Delta_s(S_2,1)) = \{\ \} = S_2$

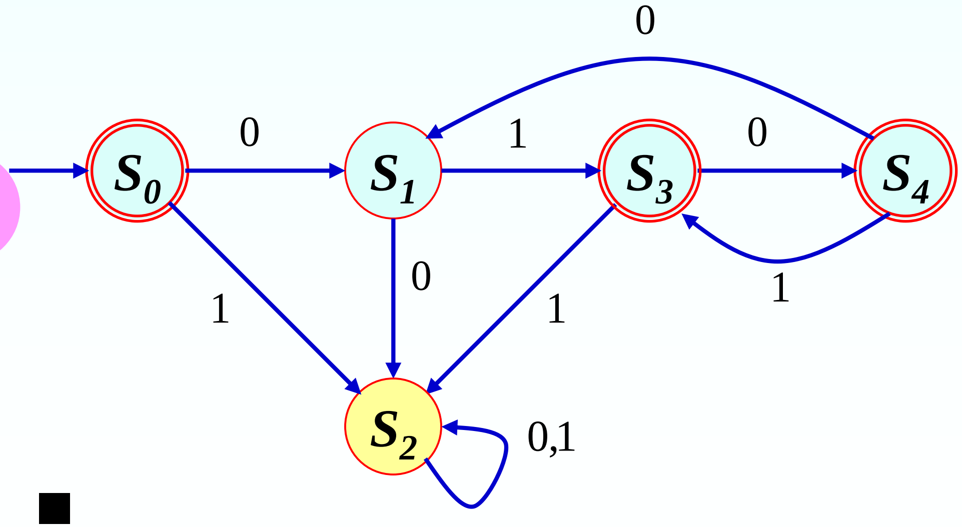
$E_s(\Delta_s(S_3,0)) = \{A,B\} = S_4$

$E_s(\Delta_s(S_3,1)) = \{\ \} = S_2$

$E_s(\Delta_s(S_4,0)) = \{B\} = S_1$

$E_s(\Delta_s(S_4,1)) = \{A,C\} = S_3$ ■





Another Example

$S_0 = E(A) = \{A, B\}$

$E_s(\Delta_s(S_0, 0)) = \{C, D\} = S_1$

$E_s(\Delta_s(S_0, 1)) = \{B\} = S_2$

$E_s(\Delta_s(S_1, 0)) = \{\ \} = S_3$

$E_s(\Delta_s(S_1, 1)) = \{A, B\} = S_0$

$E_s(\Delta_s(S_2, 0)) = \{C\} = S_4$

$E_s(\Delta_s(S_2, 1)) = \{\ \} = S_3$

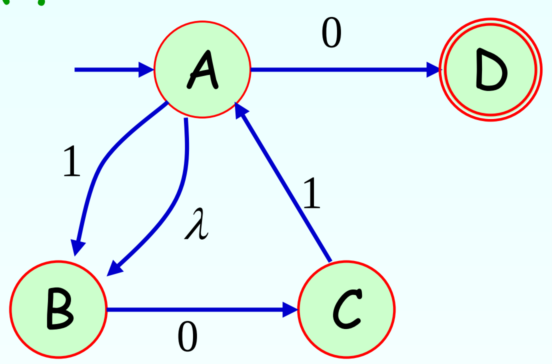
$E_s(\Delta_s(S_3, 0)) = \{\ \} = S_3$

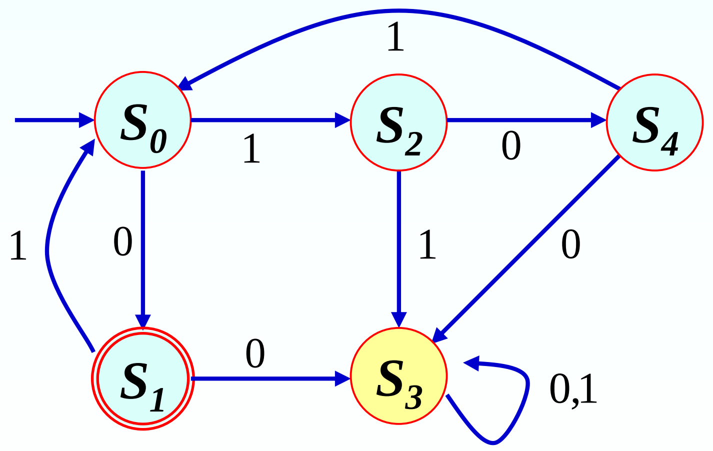
$E_s(\Delta_s(S_3, 1)) = \{\ \} = S_3$

$E_s(\Delta_s(S_4, 0)) = \{\ \} = S_3$

$E_s(\Delta_s(S_4, 1)) = \{A, B\} = S_0$ ■

$\Sigma = \{0,1\}$상의 정규식 $(01+101)^*0$ 으로 표현되는 언어를 accept하는 NFA ?





Proof of $L(M_{DFA}) = L(M_{NFA})$

For an input string w, we will show that

$$\Delta^*(q_0, w) = \delta^*(E(q_0), w)$$

where for $P \in 2^{Q_N}$, $a \in \Sigma$, and $x \in \Sigma^*$,

$\delta^*(P,a)=\delta_s(P,a)=E_s(\Delta_s(P,a))$ and $\delta^*(P,xa)=\delta_s(\delta^*(P,x),a)$.

(i) $\|w\| = 0$ 일 때, $\Delta^*(q_0,\lambda) = \delta^*(E(q_0),\lambda) = E(q_0)$.

(ii) $\|w\| < k$ 일 때, $\Delta^*(q_0,x) = \delta^*(E(q_0),x)$ 성립 가정

(iii) $\|w\| = k \geq 1$ 일 때, $w = xa$ 라고 하자.

$\Delta^*(q_0,w) = E_s(\Delta_s(\Delta^*(q_0,x),a))$ 이고,

$\delta^*(E(q_0),w)=\delta_s(\delta^*(E(q_0),x),a)=E_s(\Delta_s(\delta^*(E(q_0),x),a)$이다.

따라서, (ii)의 가정에 의해 $\Delta^*(q_0,w) = \delta^*(E(q_0),w)$.

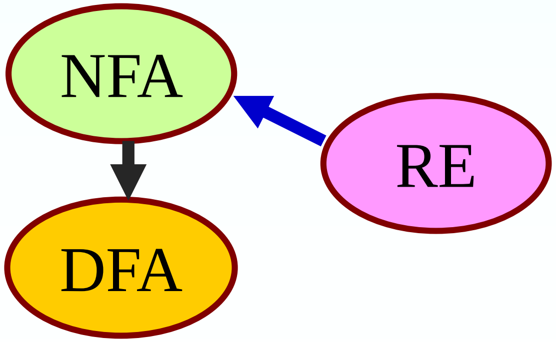
Regular Expression → NFA

- Theorem 2

  There exists a NFA that accepts the regular language $L(r)$ for a regular expression r.

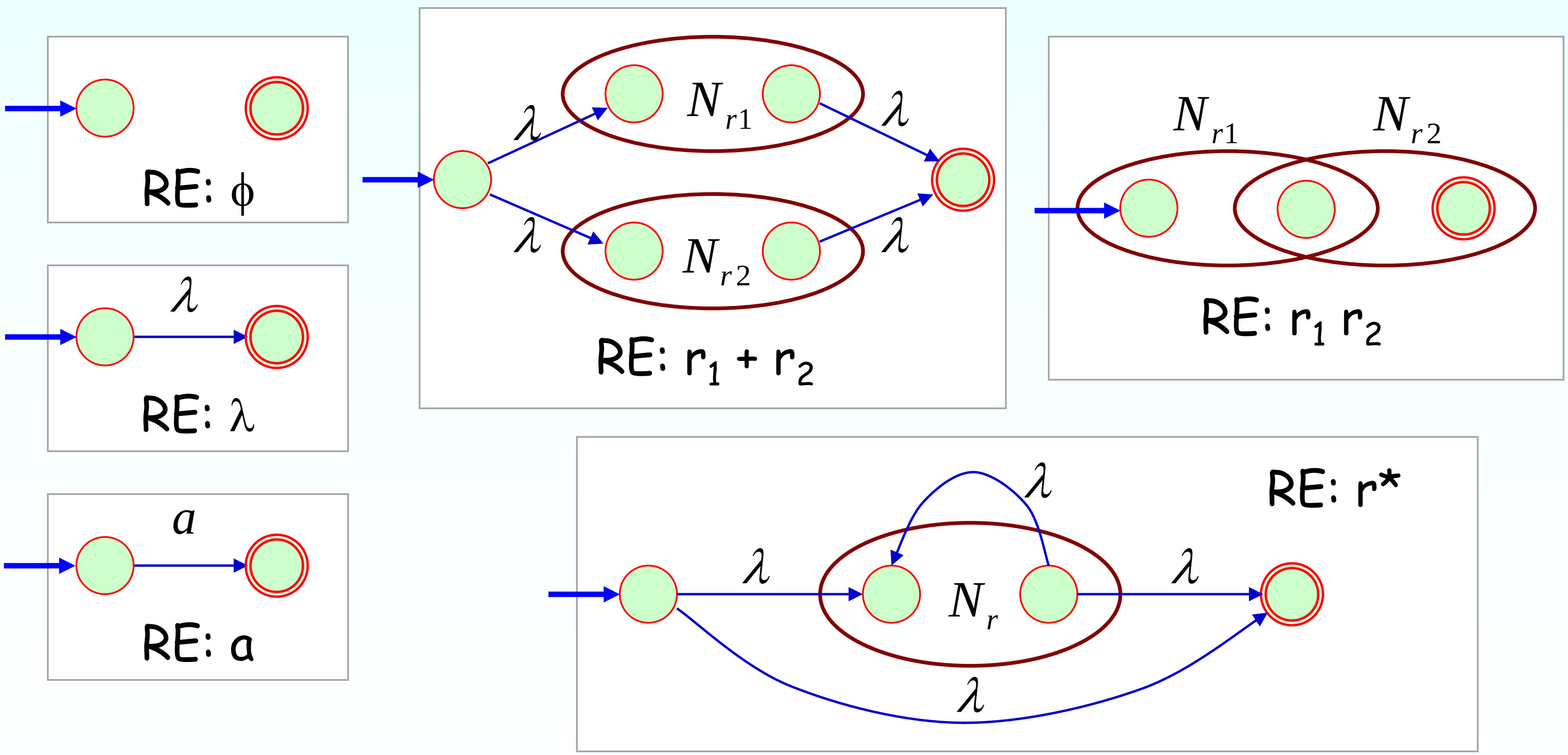
  If we can show the corresponding NFA for each type in the definition of a regular expression, then the theorem can be proved.

  Let's see the Tompson's algorithm.

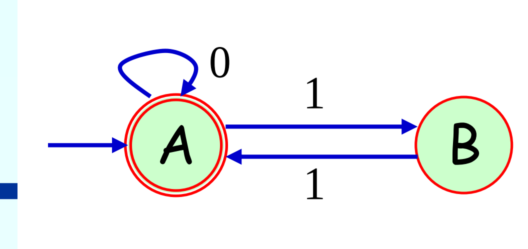


Tompson's Algorithm

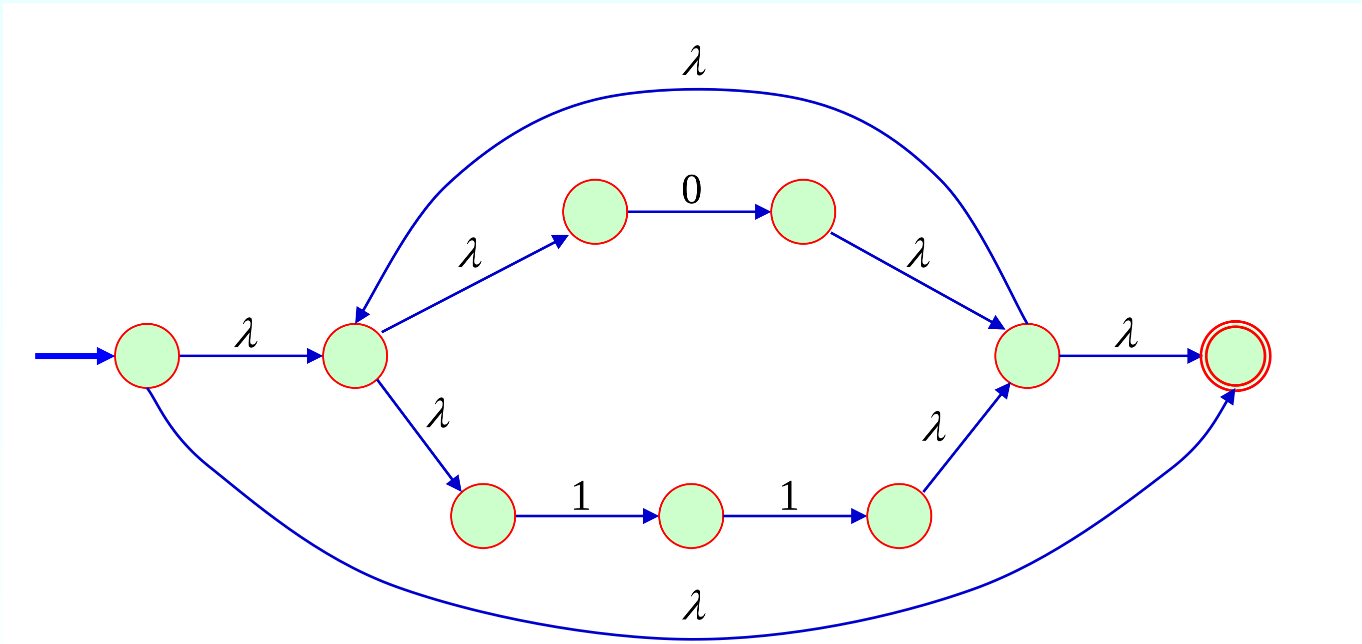
- Recursive construction of an NFA by using the following basic NFA's

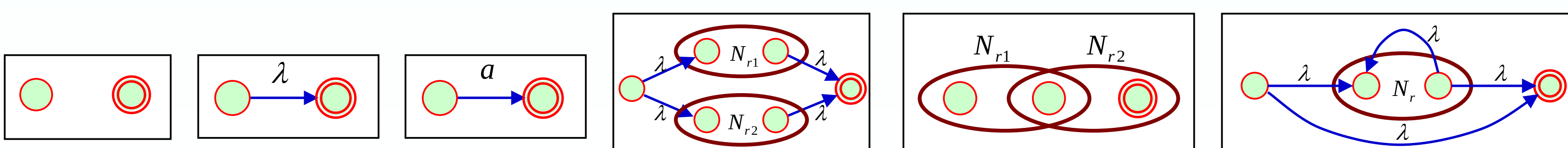


An Example

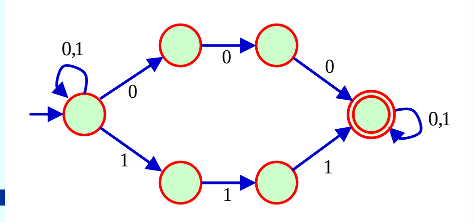


- RE = (0 + 11)* 에 대한 NFA ?



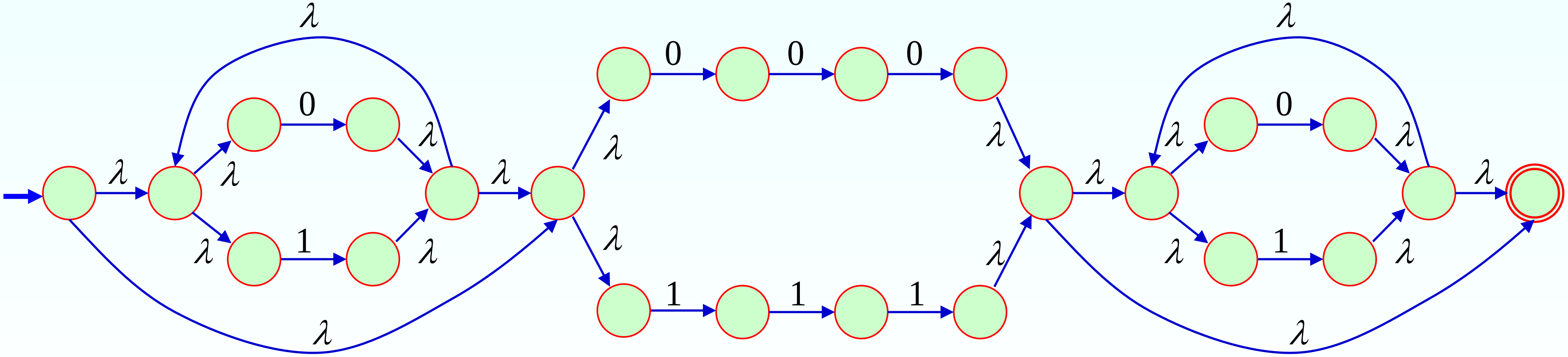


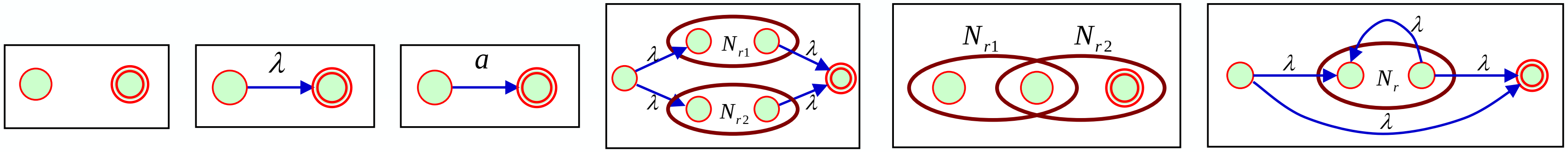
Another Example



- 0 또는 1이 연이어 세 번 나오는 substring을 갖는 string들의 집합인 언어

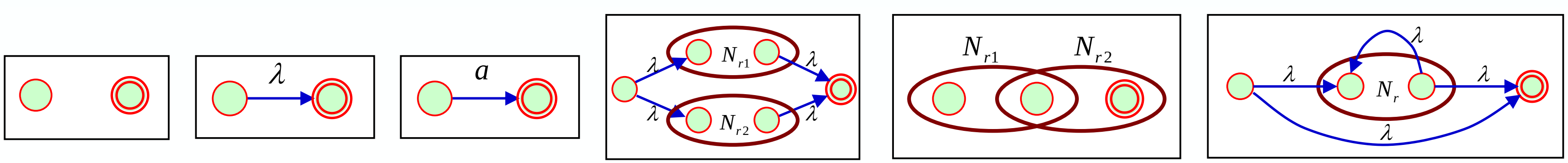
$$(0+1)^*(000+111)(0+1)^*$$





Properties of Tompson's NFA

- An NFA ***N*** constructed as above has the following properties.
  - (Number of states) ≤ 2 × (number of steps), since each step creates at most two new states.
  - The ***N*** has one start state and one accepting state, and the accepting state has no outgoing transitions.
  - Each state has either one outgoing edge labeled by a character or at most two outgoing λ-edges.



Regular Expression ← DFA

- **Theorem 3**

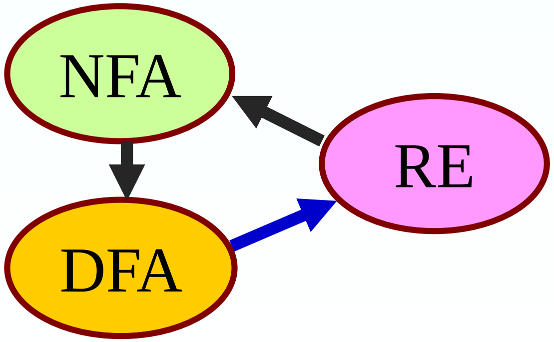
  There exists a regular expression for the language of a DFA M.

(Algorithm)

Let a DFA be $M = (\{q_1, q_2, \ldots, q_n\}, \Sigma, \delta, q_1, F)$.

R_{ij}^{k} : a set of strings that transit from q_i to q_j without passing any state numbered greater than k

Then, $L(M) = \bigcup_{q_j \in F} R_{1j}^{n}$



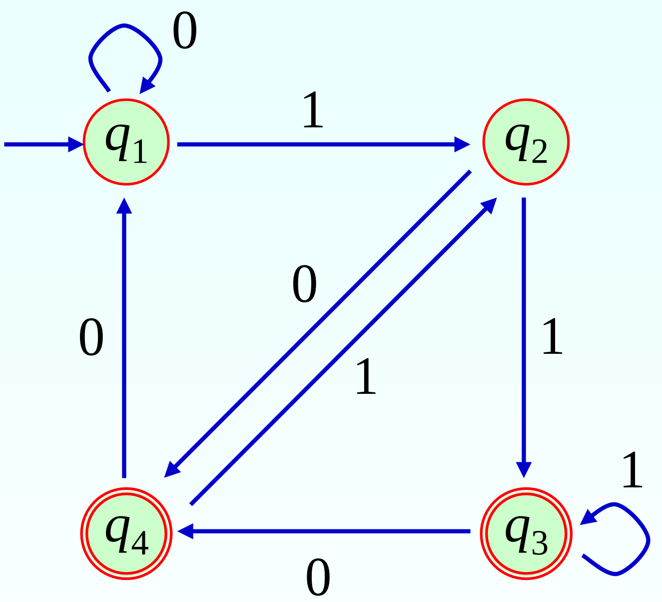
An Example

$R_{13}^{1} = \{\ \}$

$R_{13}^{2} = \{0\}^* \{11\}$

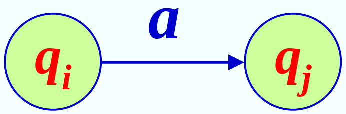
$R_{13}^{3} =$?

$R_{13}^{4} =$?



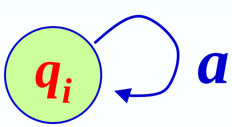
Regular Expression ← DFA

$$R_{ij}^{0} = \begin{cases} \{a \mid \delta(q_i, a) = q_j\}, & \text{if } i \neq j \\ \{a \mid \delta(q_i, a) = q_j\} \cup \{\lambda\}, & \text{if } i = j \end{cases} \qquad R_{ii}^{0}$$

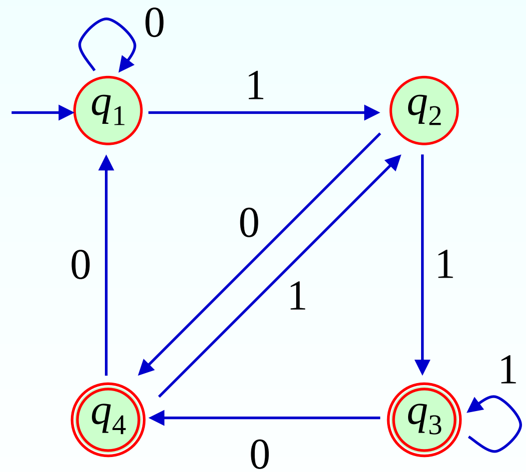


$\{a\}$

$\{\lambda\}$



$\{a, \lambda\}$



$R_{11}^{0} = \{0, \lambda\}$

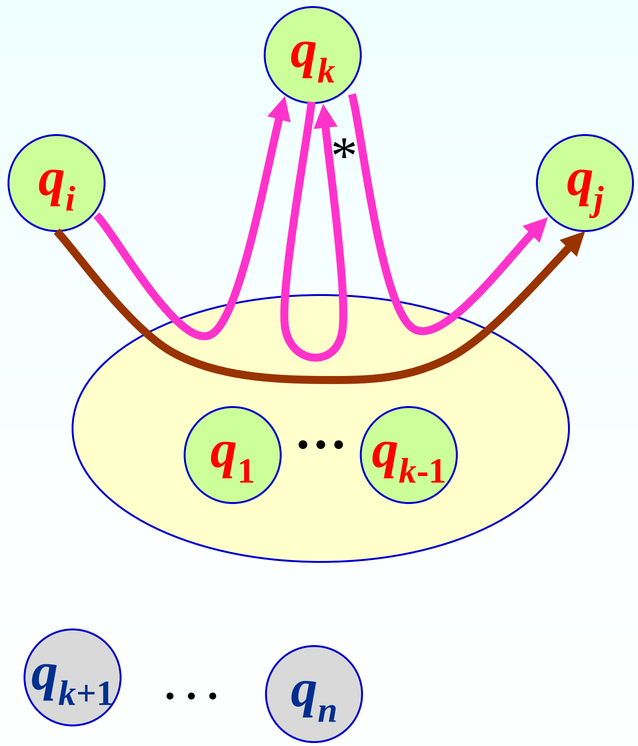
$R_{12}^{0} = \{1\}$

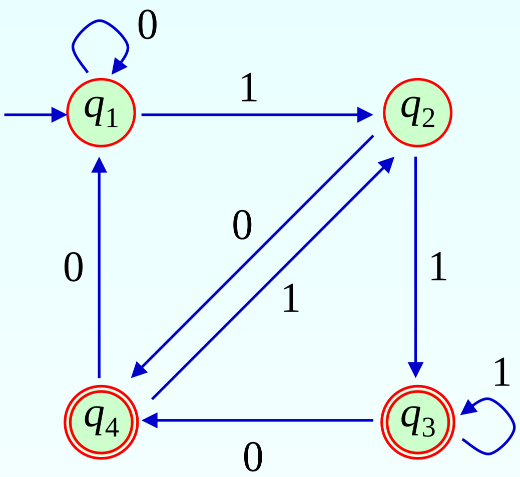
$R_{13}^{0} = \{\ \}$

$R_{14}^{0} = \{\ \}$

Regular Expression ← DFA

$$R_{ij}^{k} = \underbrace{R_{ij}^{k-1}} \cup \underbrace{R_{ik}^{k-1}\left(R_{kk}^{k-1}\right)^{*} R_{kj}^{k-1}}$$





$R_{14}^{3} = R_{14}^{2} \cup R_{13}^{2}\,(R_{33}^{2})^{*}\,R_{34}^{2}$

$= \{0\}^{*}\{10\} \cup \{0\}^{*}\{11\}\,\{1, \lambda\}^{*}\,\{0\}$

$r_{14}^{3} = 0^{*}10 + 0^{*}11(1+\lambda)^{*}0$

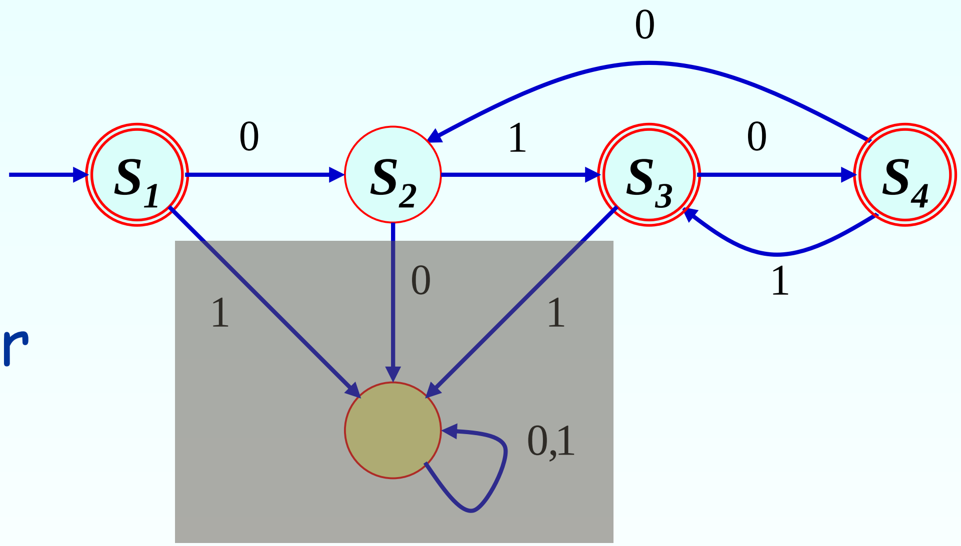
$= 0^{*}1\,(0 + 11^{*}0)$

$= 0^{*}1\,(\lambda + 11^{*})\,0$

$= 0^{*}11^{*}0 = 0^{*}1^{*}10$

An Example

- Find a regular expression for the following DFA M.



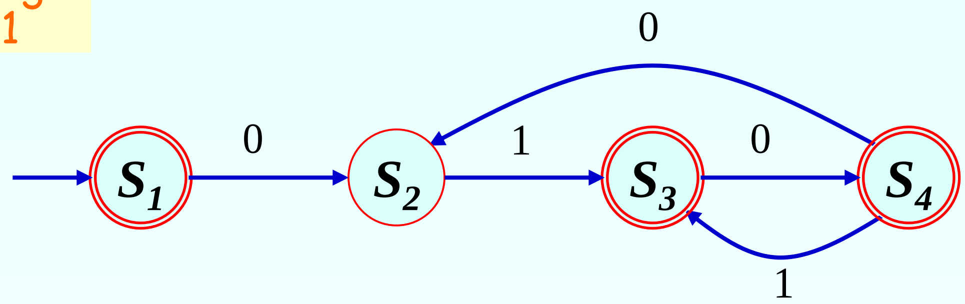
We don't have to consider dead states that are not final states.

We need to find R_{11}^4, R_{13}^4, and R_{14}^4, because there are three final states, S_1, S_3, and S_4.

$R_{11}^4 = \{\lambda\}$ $\quad R_{13}^4 = \{01\}\{001,01\}^*$ $\quad R_{14}^4 = \{010\}\{010,10\}^*$

continued

$R_{11}^{4} = R_{11}^{3} \cup R_{14}^{3} (R_{44}^{3})^{*} R_{41}^{3}$



$R_{11}^{3} = \{\lambda\}$

$$R_{ij}^{k} = R_{ij}^{k-1} \cup R_{ik}^{k-1} \left(R_{kk}^{k-1}\right)^{*} R_{kj}^{k-1}$$

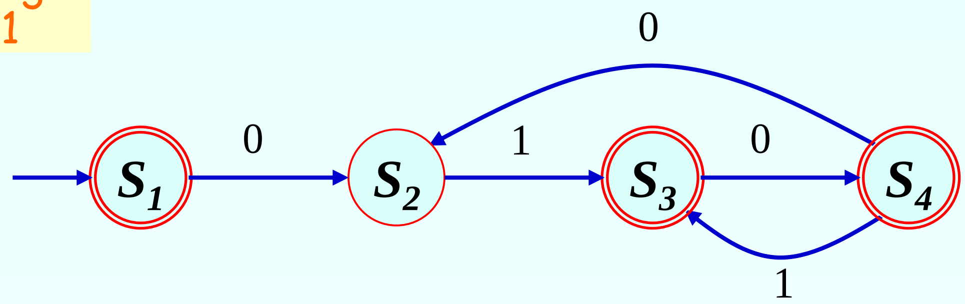
$R_{14}^{3} = R_{14}^{2} \cup R_{13}^{2} (R_{33}^{2})^{*} R_{34}^{2}$

$= \{\ \} \cup \{01\} \{\lambda\}^{*} \{0\}$

$= \{010\}$

continued

$R_{11}^4 = R_{11}^3 \cup R_{14}^3 (R_{44}^3)^* R_{41}^3$



$R_{44}^3 = R_{44}^2 \cup R_{43}^2 (R_{33}^2)^* R_{34}^2$

$= \{\lambda\} \cup \{01,1\} \{\lambda\}^* \{0\}$

$= \{\lambda,010,10\}$

$R_{44}^3 \neq \{ \lambda, (010)^*, (10)^* \}$

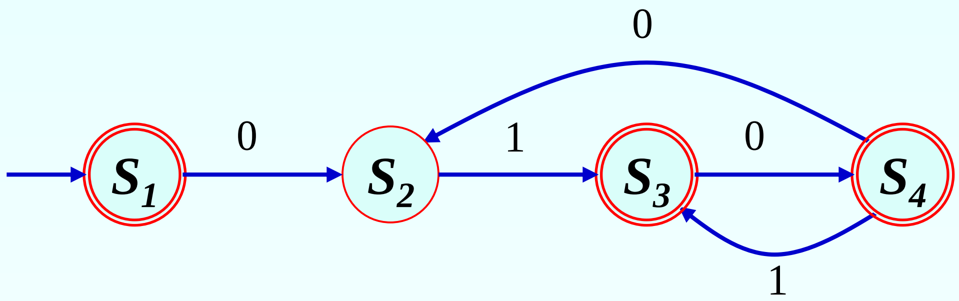
$R_{41}^3 = \{ \}$

$R_{11}^4 = \{\lambda\} \cup \{010\} \{\lambda,010,10\}^* \{ \}$

$= \{\lambda\} \cup \{ \} = \{\lambda\}$

$\therefore r_{11}^4 = \lambda$

continued



$$R_{13}^{4} = R_{13}^{3} \cup R_{14}^{3} (R_{44}^{3})^{*} R_{43}^{3}$$

$$= \{01\} \cup \{010\} \{\lambda, 010, 10\}^{*} \{01, 1\}$$

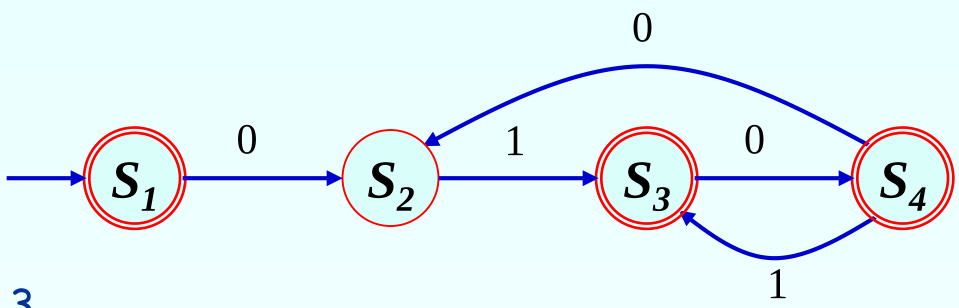
$$\therefore\ r_{13}^{4} = 01 + 010 (\lambda + 010 + 10)^{*} (01 + 1)$$

continued

$$r_{13}^4 = 01 + 010(\lambda+010+10)^*(01+1)$$
$$= 01 + 010(010+10)^*(01+1)$$
$$= 01 + 010((01+1)0)^*(01+1)$$
$$= 01 + 01(0(01+1))^*0(01+1)$$
$$= 01 + 01(001+01)^*(001+01)$$
$$= 01(\lambda+(001+01)^+)$$
$$= 01(001+01)^*$$

$$(st)^*s = (\lambda+st+stst+\ldots)s$$
$$= s+sts+ststs+\ldots$$
$$= s(\lambda+ts+tsts+\ldots)$$
$$= s(ts)^*$$

continued



$$R_{14}^{4} = R_{14}^{3} \cup R_{14}^{3} (R_{44}^{3})^{*} R_{44}^{3}$$

$$= \{010\} \cup \{010\} \{\lambda, 010, 10\}^{*} \{\lambda, 010, 10\}$$

$$= \{010\} \cup \{010\} \{\lambda, 010, 10\}^{+}$$

$$\therefore\ r_{14}^{4} = 010 + 010\ (\lambda + 010 + 10)^{+} = 010 + 010\ (010 + 10)^{*}$$

$$= 010\ (\lambda + (010 + 10)^{*})$$

$$= 010\ (010 + 10)^{*}$$

continued

RE $r = r_{11}^{4} + r_{13}^{4} + r_{14}^{4}$

$= \lambda + 01(001+01)^* + 010(010+10)^*$

$= \lambda + 01((0+\lambda)01)^* + 010((0+\lambda)10)^*$

$= \lambda + (01(0+\lambda))^*01 + 0(10(0+\lambda))^*10$

$= \lambda + (010+01)^*01 + 0((10+1)0)^*10$

$= \lambda + (010+01)^*01 + (0(10+1))^*010$

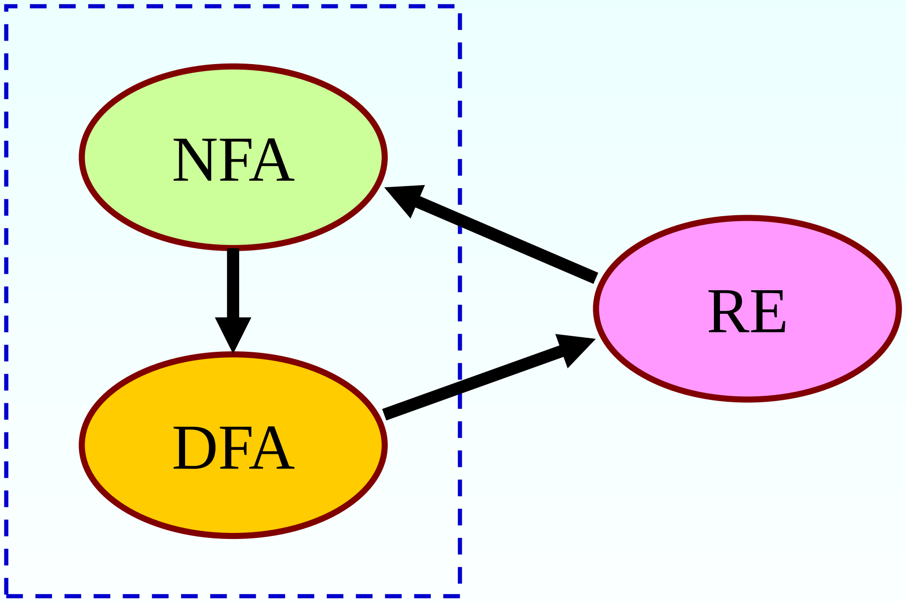
$= \lambda + (010+01)^*01 + (010+01)^*010$

$= \lambda + (010+01)^*(010+01)$

$= \lambda + (010+01)^+$

$= (010+01)^*$

Summary of Conversions



Finite Automata

H/W #3

- 다음의 언어에 대하여,

  Alphabet $\Sigma = \{0,1\}$

  (L_1) 1의 개수가 두 개 이하이면서 1은 연이어 나타나는 string의 집합

  (L_2) Prefix 01 또는 11을 가지지 않는 string의 집합

  (L_3) Substring 01 또는 11을 가지지 않는 string의 집합

H/W #3

(1) 각 언어에 대한 regular expression을 구해 보시오.

(2) 각 언어에 대한 DFA를 구해 보시오.

(3) 각 regular expression에 대한 NFA를 Tompson's algorithm을 이용하여 구해 보시오.

(4) Tompson's algorithm에 의한 NFA보다 효과적인 NFA를 직관적인(intuitive) 방법으로 각각 구해 보시오.

(5) Regular expression $(\lambda+0+01)1^*$ 에 의한 regular language를 인식하는 NFA, DFA, regular grammar를 구해 보시오.