제 2장. Finite Automata

학습목표

가장 단순한 **Finite Automata**를 통해 **DFA**와 **NFA**를 이해하고 그 둘의 능력이 같음을 확인한다.

↓

No temporary storage

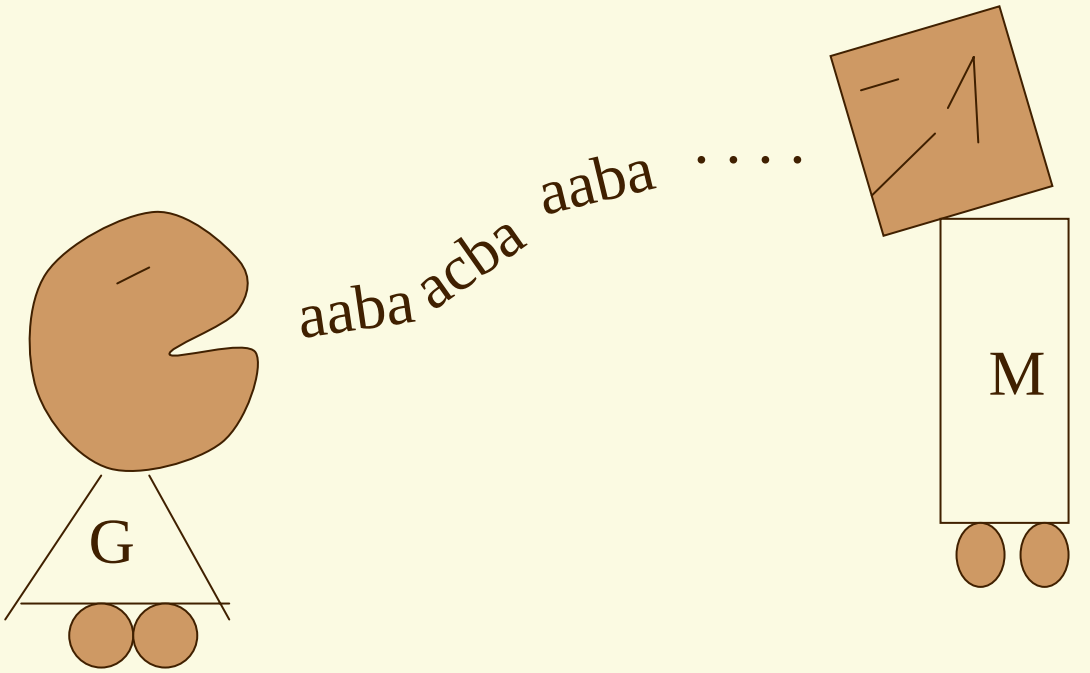
↓

닭머리 기계 **(**기억용량에 제약**)**

↓

CU내의 **finite**한 분량의 상태를 저장하여 처리

뭘 배우나?



언어 발생 모델: 문법
(Models of Language Generation: Grammars)

언어 인식 모델:계산기
(Models of Language Recognition: Machines)

개요

- Deterministic Finite Accepters (DFA)
  - DFA의 정의와 transition graph
  - DFA가 accept하는 언어 → regular language
- Nondeterministic Finite Accepters (NFA)
  - NFA의 정의와 nondeterminism의 필요성
- Equivalence of DFA & NFA
- Reduction of Number of States in FA
  - *mark & reduce algorithm*

Deterministic Accepters

DFA의 formal한 정의와 작동과정을 이해하자

· *Def*. DFA $M = (Q, \Sigma, \delta, q_0, F)$

Q : internal states

Σ : input alphabet

$\delta : Q \times \Sigma \to Q$ transition function

$q_0 \in Q$: initial state

$F \subseteq Q$: final states

$q_0 \to q_1 \cdots$ end of string $\begin{cases} w \text{ accepted if } q_i \in F \\ \text{rejected} \end{cases}$

$w \in \Sigma^*$

$\to$ 1 input

$\delta(q_0, a) = q_1$

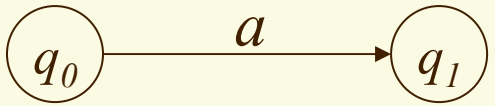
DFA : Transition Graph

DFA를 시각적으로 쉽게 볼 수 있도록 하는 그래프 표현방법 이해

· vertex : states $|Q|$ q_i

edge : transition

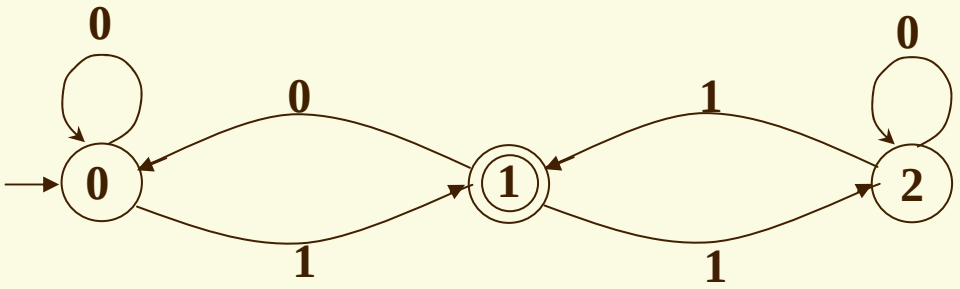
(label : input symbol) a



· initial state

· final states :

Ex. 2.1



01?

00 ?

- Extended transition function δ^* : Q x Σ^* → Q

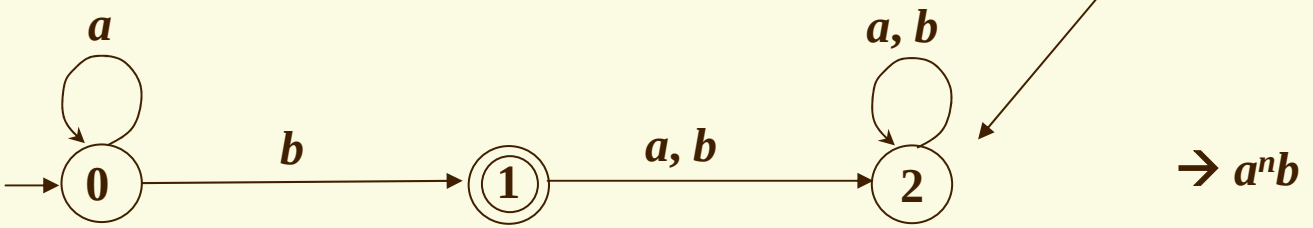
DFA : Languages

DFA가 accept하는 언어의 정의와 Transition graph와의 관계 이해

L(M) = {w | M은 input string w를 모두 읽고 accepting state에 들어간다}

- $L(M) = \{w \in \Sigma^* : \delta^*(q_0, w) \in F\}$

- **Nonacceptance?**
- **Ex 2.2 : multiply labeled edges**



→ $a^n b$

- **Thm : $M = (Q, \Sigma, \delta, q_0, F)$ ←→ G_M**

→ **induction on $|w|$**

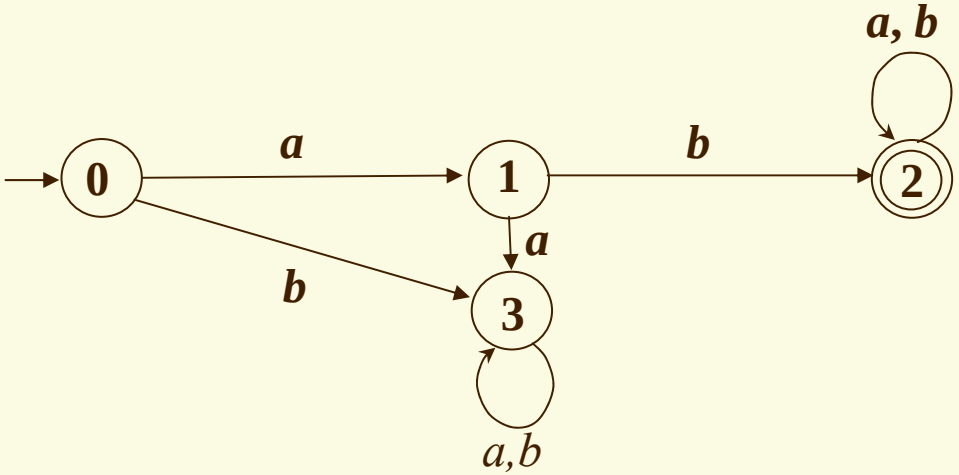
$\delta^*(q_i, w) = q_j$ **iff** $q_i \overset{w}{\rightarrow} q_j$

- 구현 **: simple table-lookup, sequence of "if" statements**

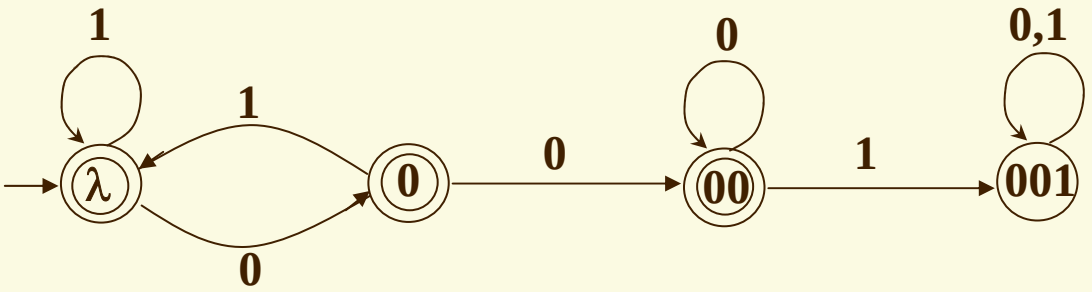
DFA : 예들

주어진 언어를 **accept**하는 **DFA**의
설계방법을 여러 가지 연습을 통해 이해할 것

- Ex 2.3 : ab로 시작하는 모든 **string**의 집합을 **accept**하는 **DFA** 설계



- Ex 2.4 : 001을 포함하지 않는 모든 **string**을 **accept**하는 **DFA** 설계

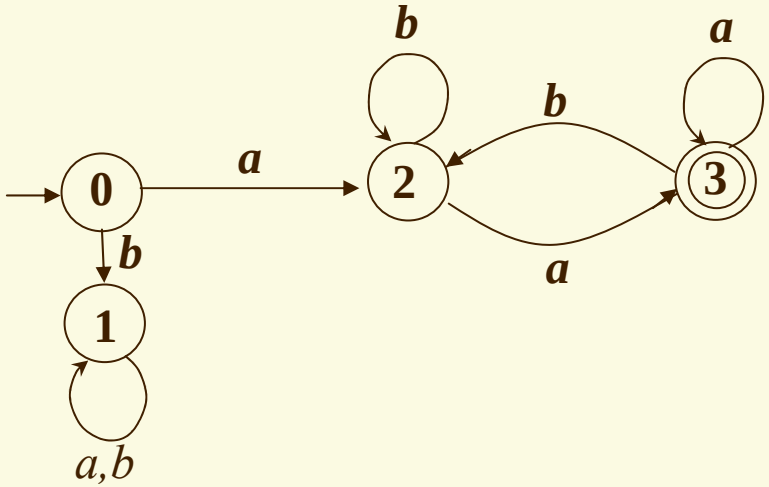


DFA : Regular Languages

DFA가 accept하는 언어 Family를 정규언어라 함

- ***L*** **: regular *iff* there exists some DFA *M* such that *L* = *L*(*M*)**
- **To show L = regular → find a DFA for it**
- **Ex 2.5 : *L* = {*awa* : *w* in {*a*, *b*}*} is regular?**

  •**Point : no explicit way of testing the end of string**
  → 두번째 **a**가 나올 때마다 일단 **final state**로 가본다



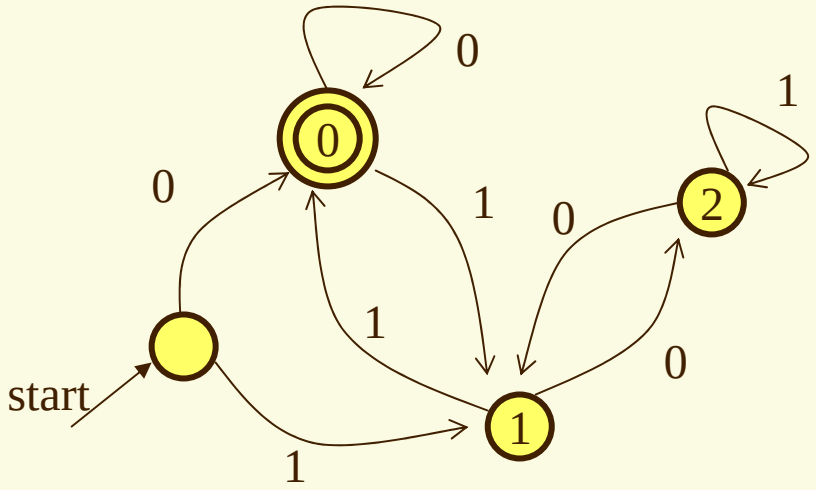
- **Ex 2.6 : L^2 is regular**

→**Point : aa**가 나오면 두번째 **DFA**로
→그림 2.7 → ***L*=regular -> L^2, L^3, … is regular**

실전연습: 다음 언어 **L**을 인식하는 **DFA**를 설계하시오.

L = {x | x 는 binary 수이며 (즉, x ∈ {0, 1}$^+$), x mod 3 = 0}

힌트: 이 **DFA**의 **idea**는 입력으로 주어진 **string**을 한 **bit** 씩 우측으로 차례로 읽으며 지금까지 읽은 수에 대한 나머지를 계산해 두는 방법이다. **Graph**에서 **state** 상의 수는 지금까지 읽은 수를 **3**으로 나눈 나머지이다. 지금까지 읽은 수에 대한 나머지를 **i** 라 하자. 다음 읽은 **bit**가 **j∈ {0,1}** 이면, 나머지는 **(2i + j) mod 3** 이다. 따라서 현 상태 **i** 상에서 **j∈ {0,1}** 를 읽으면, 상태 **(2i + j) mod 3** 로 가도록 하면 된다.



DFA : Exercises (2.1)

-2 : grammar와 오토메타에 대한 기초이해 문제

-7 : DFA에 의한 **modular counting** 연습 **(**참고**: 1.2**절의 **Exercise 14)**

-11, 12 : regular 언어란 → 해당 **DFA** 구축하기**!!**

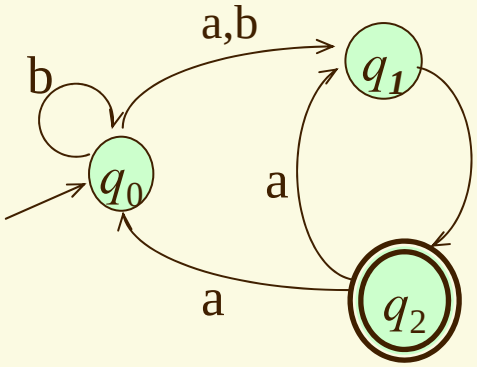
-22 (20) : 정규언어의 특성**(4**장**)** 중 중요한 **closure property**에 대한 문제

NFA: 정의

Unique move대신에 set of possible move 허용

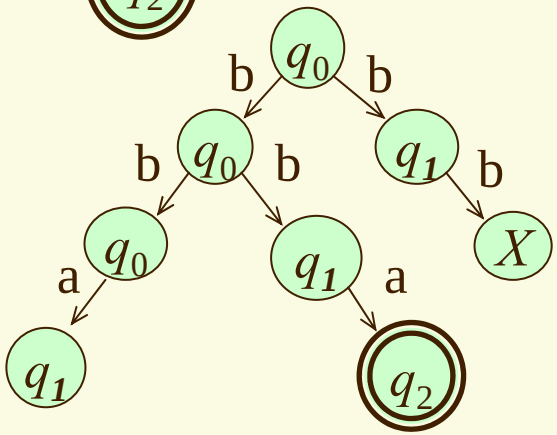
· $M = (Q, \Sigma, \delta, q_0, F)$

- Q, Σ, q_0, F : DFA 와 동일
- $\delta : Q \times (\Sigma \cup \{\lambda\}) \rightarrow 2^Q$



· 차이점

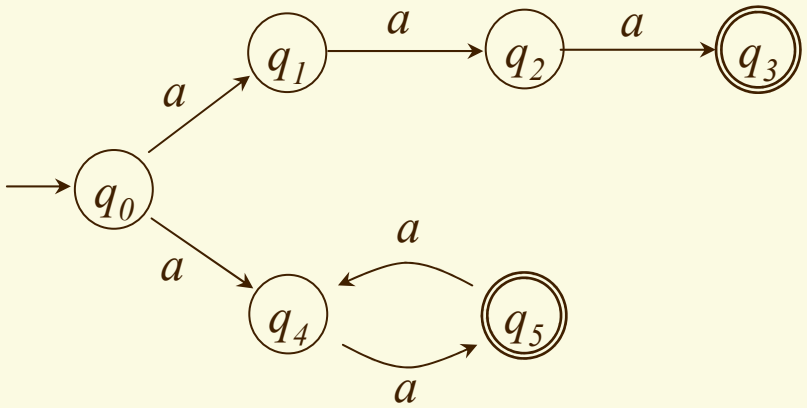
- δ의 range (2^Q) : subset of states
  ex) $\delta(q_1, a) = \{q_0, q_2\}$
- input symbol 이 λ일 수 있음
- $\delta(q_i, a)$가 empty 일 수 있음



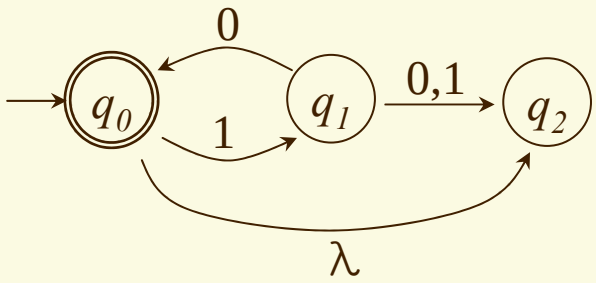
$\Rightarrow$ unique move $\rightarrow$ a set of possible states

$\Rightarrow$ accept a string if $\exists$ sequence of possible moves to a final state

NFA: 예



$\{a^3\} \cup \{a^{2n} \mid n \geq 1\}$



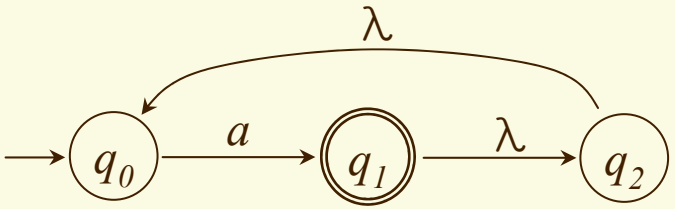
$\{(10)^n \mid n \geq 0\}$

NFA : Extended Transition Function

.정의

$\delta^*(q_i, w)$ contains q_j *iff* $\exists$ a walk in transition graph from q_i to q_j labeled w

.예



$\delta^*(q_1, a) = \{q_0, q_1, q_2\}$

$\delta^*(q_2, \lambda) = \{q_0, q_2\}$

$\delta^*(q_2, aa) = \{q_0, q_1, q_2\}$

NFA : Language Acceptance

· 정의

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

· A method for computing $\delta^*(q_i, w)$

① evaluate all walks of length at most $\Lambda + (1 + \Lambda)\,|w|$ originating at v_i

② select from them labeled w

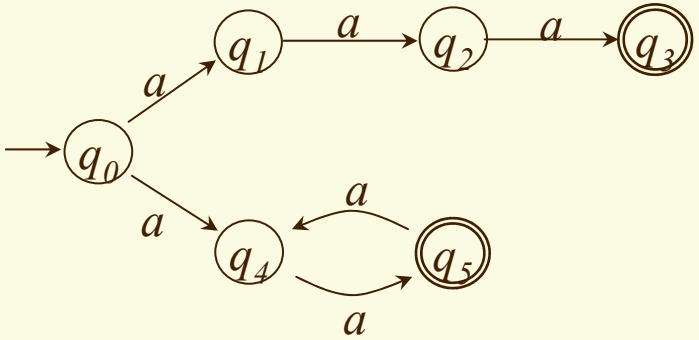
NFA: Nondeterminism?

왜 굳이 **NFA**를 사용하려 하는지에 대한 이해필요

·방법:search – and – backtrack without exhaustive search

·이유

– helpful in solving problems easily



– 복잡한 언어를 간결하게 표현

ex) $S \rightarrow aSb \mid \lambda$

– DFA보다 NFA로 쉽게 설명가능한 사실 존재

ex) theorem, lemma

$\Rightarrow$ Simplify formal arguments without affecting the generality of conclusion

NFA: Exercises (2.2)

-2 : DFA와 **NFA**의 **equivalence**에 대한 사전이해

-12~15 (11~14) : NFA와 언어에 대한 기초적인 이해

-16 (15) : 약간 어려운 문제일까**?** 뒤의 해답을 참조하여 이해할 것

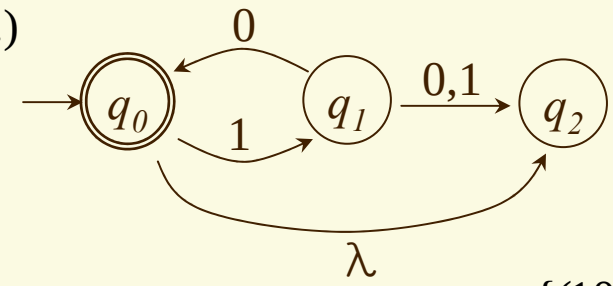
Equivalence of DFA & NFA

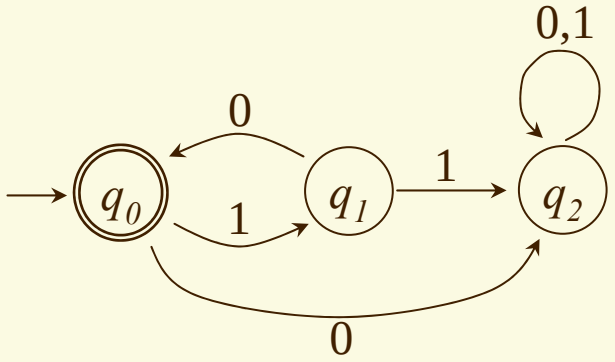
$$M_D = (Q_D, \Sigma, \delta_D, q_0, F_D) \qquad M_N = (Q_N, \Sigma, \delta_N, q_0, F_N)$$

$$\delta_D : Q \times \Sigma \to Q \qquad \delta_N : Q \times (\Sigma \cup \{\lambda\}) \to 2^Q$$

· *Def*. $M_1 \equiv M_2$ *iff* $L(M_1) = L(M_2)$

· *Ex*.)





$\{(10)^n : n \geq 0\}$

· powerfulness :

$$\forall w \in L(M_N), \quad \exists M_D \ni w \in L(M_D)$$

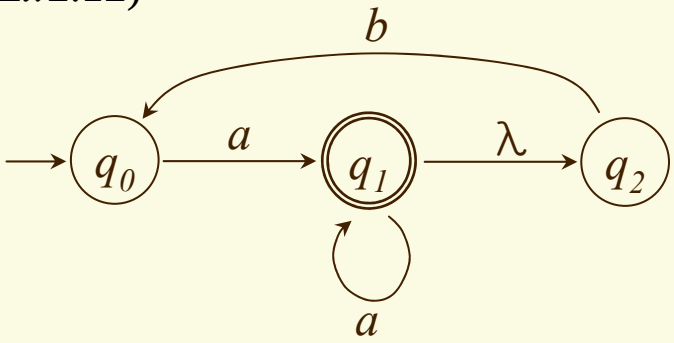
NFA → DFA

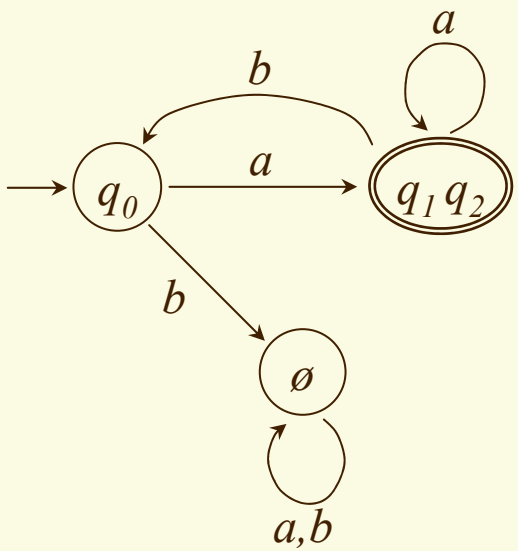
기본 아이디어는 **NFA**의 상태수가 아무리 많아도 $2^{|Q|}$개의 유한 수임을 이용한다

· NFA → DFA

$|Q| \rightarrow 2^{|Q|}$

*Ex*2.12)





Theorem for NFA→DFA (1)

알고리즘과
종료함의 증명

· Thm : $L = L(M_N) \rightarrow \exists M_D \ni L = L(M_D)$

pf.) nfa → dfa

① $\{q_0\}$: initial state

② repeat

$\{q_i, q_j, \cdots, q_k\}$ of G_D for $a \in \sum$

$\delta^*(q_i, a) \cup \delta^*(q_j, a) \cup \cdots \cup \delta^*(q_k, a) \rightarrow \{q_l, q_m, \cdots, q_n\}$

state

$\{q_i, q_j, \cdots, q_k\} \xrightarrow{a} \{q_l, q_m, \cdots, q_n\}$

③ $\{q_f \in F_n\}$ in G_D : final state

④ $\lambda \in L(M_N) \rightarrow \{q_0\}$: final state

- terminate : G_D has at most $2^{|Q_N|} |\sum|$ edges

Theorem for NFA→DFA (2)

동일함의 증명
by induction (8:2☺)

- induction on |input string|

i) $n = 1$ true

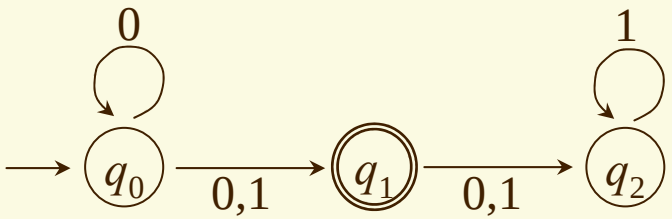
$$\delta_N^*(q_0, w) \in F \;\rightarrow\; \delta_D^*(q_0, w) \in F$$

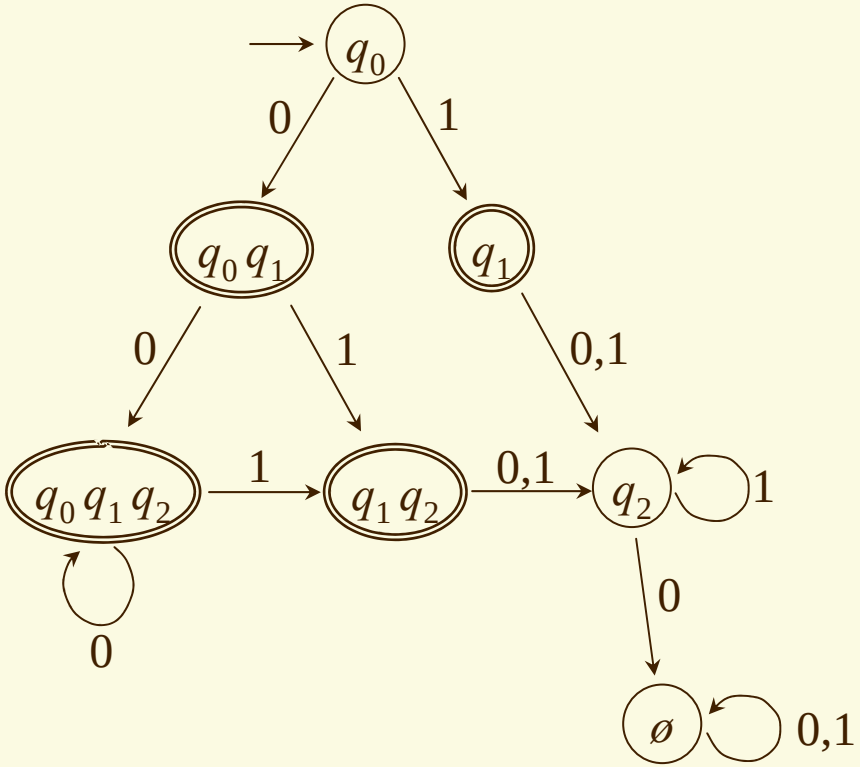
ii) Q_i

$$\begin{cases} \forall |v| \le n & q_0 \xrightarrow[v]{} q_i \text{ in } G_N \Rightarrow \{q_0\} \xrightarrow[v]{} \{\cdots q_i \cdots\} \text{ in } G_D \\ w = va & q_0 \xrightarrow[v]{} q_i \xrightarrow[a]{} q_l \Rightarrow \{q_0\} \xrightarrow[v]{} Q_i \xrightarrow[a]{} \{\ , q_l, \ \} \end{cases}$$

NFA→DFA : 예

Ex 2.13 :





· $L(M_N)$: *regular*

NFA로 accept되는 언어도 regular임!

NFA→DFA : Exercises (2.3)

-1 : 진짜 연습문제

-5 : 이럴 경우**, 8:2 rule**에 의하면…**?**

- 13 : 주어진 언어가 무엇인지 먼저 파악할 것**.**

Reduction of Number of States

- Simplicity, Storage efficiency

Ex 2.14 : p.63 { Inaccessible / reachable

- *Def* : (p,q) : indistinguishable if $\forall w \in \sum^*$

$$\begin{cases} \delta^*(p,w) \in F \rightarrow \delta^*(q,w) \in F \\ \delta^*(p,w) \notin F \rightarrow \delta^*(q,w) \notin F \end{cases}$$

cf) distinguishable $\delta^*(p,w) \in F \rightarrow \delta^*(q,w) \notin F$, or vice versa

cf) equivalence relation $(p,q)(q,r) \rightarrow (p,r)$

Mark Algorithm

Distinguishable states를 모두 찾아 내는 알고리즘

· Idea : finding & combining indistinguishable states

mark *distinguishable pairs*

① remove all inaccessible

② $\forall (p,q)$ ∋ $p \in F \,\&\, q \notin F \rightarrow mark(p,q)$ distinguishable states

③ repeat $\forall (p,q) \,\&\, a \in \Sigma$

$$\delta(p,a) = p_a,\ \delta(q,a) = q_a \quad (p_a, q_a) : \text{distinguishable} \rightarrow mark(p,q)$$

·Thm : mark terminates & determines all distinguishable states

→ 증명 : **8:2**☺

Reduce Algorithm

구별이 안 되는 **states**를 모아서
State의 수를 줄이는 알고리즘

- *reduce* *indistinguishable state* $M_D = (Q, \Sigma, \delta, q_0, F) \rightarrow \hat{M} = (\hat{Q}, \hat{\Sigma}, \hat{\delta}, \hat{q}_0, \hat{F})$

① find all pairs of distinguishable state with *mark* algorithm
find indistinguishable states $\underbrace{\{q_i, \cdots, q_k\}}_{\text{a state}}, \underbrace{\{q_l, \cdots, q_n\}}$

② $\delta(q_r, a) = q_p$ in M

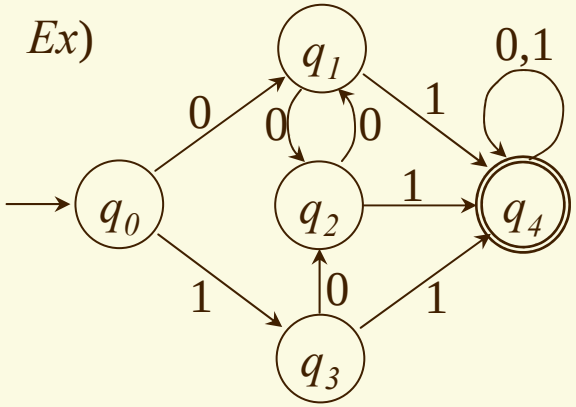
$\hat{\delta}(ij \cdots k, a) = lm \cdots n$ in $\hat{M} \rightarrow q_r \in \{q_i, \cdots, q_k\}$
$q_p \in \{q_l, \cdots, q_n\}$

③ initial state $\hat{q}_0$ including q_0

④ final states $\hat{F}$ containing $q_i \in F$

State Reduction : 예

Ex)



distinguishable pairs

(q_0, q_4) (q_0, q_1)

(q_1, q_4) (q_0, q_2)

(q_2, q_4) (q_0, q_3)

(q_3, q_4)

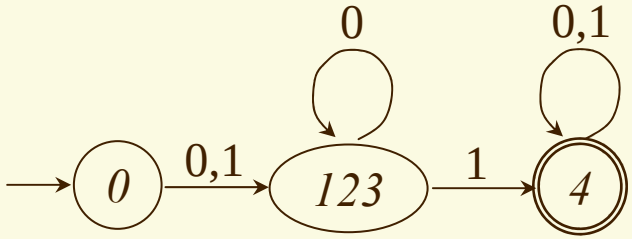
$\because \begin{cases} \delta(q_1,1) = q_4 \\ \delta(q_0,1) = q_3 \end{cases}$

$\Rightarrow$ indistinguishable pairs

(q_1, q_2) (q_1, q_3) (q_2, q_3)

$\Rightarrow$ $q_1\ q_2\ q_3$: indistinguishable

$\Rightarrow$ $\{q_0\},\ \{q_1, q_2, q_3\},\ \{q_4\}$



· Thm $\begin{cases} \text{Given } M, \textit{ reduce} \text{ yields } \hat{M} \ni L(M) = L(\hat{M}) \\ \hat{M} : \text{ minimal} \end{cases}$

p.67

→증명 : **1) 동일한가? 8:2☺ by induction**

2) Minimal? 이제 슬슬 **8:2☺면?**

State Reduction : Exercises (2.4)

-2 : 골라서 2문제 정도는 연습할 것

-4 : 뒤의 힌트를 참고하면…