

Automata Theory

Chapter 6



부산대학교
PUSAN NATIONAL UNIVERSITY

Overview

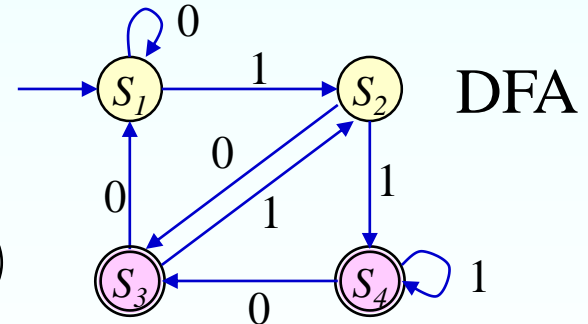
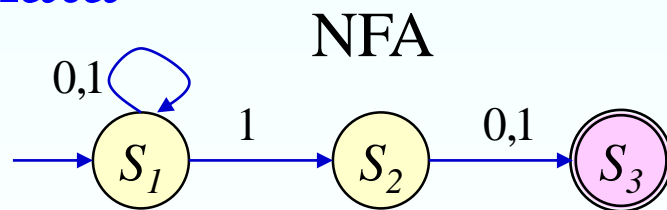
- ❑ Set Theory of Strings (Chapter 6.1)
- ❑ Regular Expressions & Regular Languages
- ❑ Finite State Machines
 - Deterministic Finite Automata (DFA)
 - Non-deterministic Finite Automata (NFA)
- ❑ Grammars and Languages
 - Types of Grammars and Languages
 - Associated Machines including Turing Machine
- ❑ Computation Theory
 - NP Problem

An Introduction Example

- $\{0,1\}$ 알파벳으로 구성된 스트링 중에서,
끝에서 두 번째 문자가 1인 것들의 집합 (언어)
 $\{ 10, 11, 010, 011, 110, 111, \dots \}$

Regular Expression $(0 + 1)^* 1(0 + 1)$

Finite Automata



Regular Grammar $G = (\{S, A, B\}, \{0,1\}, S, P)$
 $S \rightarrow 0S \mid 1S \mid 1A$
 $A \rightarrow 0B \mid 1B$
 $B \rightarrow \lambda$

An Introduction Example

- $\{0, 1\}$ 알파벳으로 구성된 스트링 중에서,
끝에서 두 번째 문자가 1인 것들의 집합 (언어)
 $\{ 10, 11, 010, 011, 110, 111, \dots \}$

Regular Expression $\underline{(0+1)^*} 1 \underline{(0+1)}$

(0 또는 1)을
0번 이상 반복
가능

0 또는 1

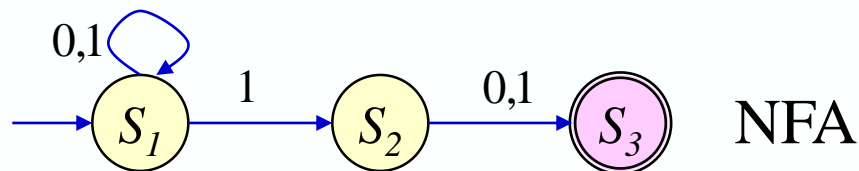
An Introduction Example

- $\{0,1\}$ 알파벳으로 구성된 스트링 중에서,
끝에서 두 번째 문자가 1인 것들의 집합 (언어)
 $\{ 10, 11, 010, 011, 110, 111, \dots \}$

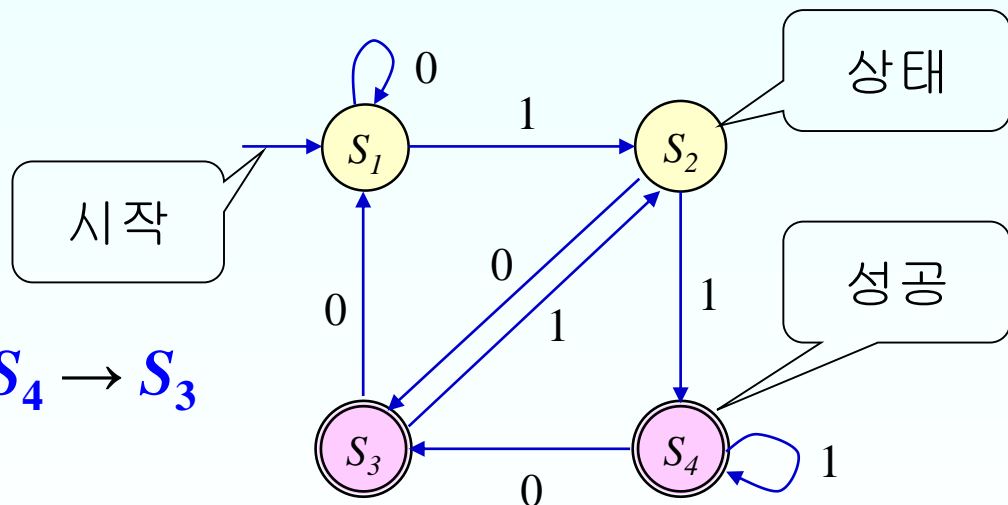
Finite Automata

(예) **0 1 1 1 0**

$S_1 \rightarrow S_1 \rightarrow S_2 \rightarrow S_4 \rightarrow S_4 \rightarrow S_3$



NFA



DFA

An Introduction Example

- $\{0,1\}$ 알파벳으로 구성된 스트링 중에서,
끝에서 두 번째 문자가 1인 것들의 집합 (언어)
 $\{ 10, 11, 010, 011, 110, 111, \dots \}$

Variables

Regular Grammar $G = (\{S, A, B\}, \{0,1\}, S, P)$

$S \rightarrow 0S \mid 1S \mid 1A$

$A \rightarrow 0B \mid 1B$

Production Rules

(예) 0 1 1 1 0

$B \rightarrow \lambda$

$S \rightarrow 0S \rightarrow 01S \rightarrow 011S \rightarrow 0111A \rightarrow 01110B \rightarrow 01110$

Set Theory of Strings

Section 6.1



부산대학교
PUSAN NATIONAL UNIVERSITY

Alphabets

□ Alphabet

We will use Σ to denote a nonempty finite set of symbols, collectively called an alphabet.

- Example: $\Sigma = \{0,1\}$ or $\Sigma = \{a,b,c,d,e\}$

In any alphabet Σ , we don't list elements that can be formed from other elements of Σ by juxtaposition.

- If $a, b \in \Sigma$, then the string ab is the juxtaposition of the symbols a and b .
- Alphabets such as $\Sigma = \{0,1,2,11,12\}$ and $\Sigma = \{a,b,c,ba,aa\}$ are not considered.

Powers of Σ

□ Definition 6.1

If Σ is an alphabet and $n \in \mathbb{Z}^+$, we define the **powers of Σ** recursively as follows.

1) $\Sigma^1 = \Sigma$ and

2) $\Sigma^{n+1} = \{ \mathbf{xy} \mid \mathbf{x} \in \Sigma, \mathbf{y} \in \Sigma^n \}$, where xy denote the juxtaposition of x and y .

➤ For $\Sigma = \{0,1\}$,

$$\Sigma^2 = \{00,01,10,11\} \text{ and}$$

$$\Sigma^3 = \{000,001,010,011,100,101,110,111\}.$$

Powers of Σ

□ In general, $|\Sigma^n| = |\Sigma|^n$

Because we are dealing with arrangements (of size $n \in \mathbb{Z}^+$) where we are allowed to repeat any of the $|\Sigma|$ objects.

$|\Sigma|$: Cardinal Number

➤ Suppose that $\Sigma = \{a, b, c, d, e\}$.

Then Σ^3 would contain $|\Sigma|^3 = 5^3 = 125$ three-symbol strings (e.g., *aaa*, *acb*, *ace*, *cdd*, *eda*, etc.)

a c e

Strings

□ If Σ is an alphabet,

$$\text{a) } \Sigma^+ = \bigcup_{k=1}^{\infty} \Sigma^k = \bigcup_{k \in \mathbb{Z}^+} \Sigma^k$$

$$\text{b) } \Sigma^* = \bigcup_{k=0}^{\infty} \Sigma^k = \boxed{\Sigma^0} \cup \Sigma^+$$

?

□ String

We shall refer to the elements of Σ^+ or Σ^* as **strings**, words, or sentences.

➤ For $\Sigma = \{0,1,2\}$, we find such strings as 0, 01, 102, and 1122 in both Σ^+ and Σ^* .

Empty String

□ Definition 6.2

For an alphabet Σ we define $\Sigma^0 = \{\lambda\}$,
where λ denotes the **empty string**
that consists of **no symbols** taken from Σ .

The symbol λ is never an element in alphabet Σ .
We should **not** mistake it for the **blank** (space)
that is found in many alphabets.

Although $\lambda \notin \Sigma$, we do have the **empty set** $\emptyset \subseteq \Sigma$.

$$1) \{\lambda\} \not\subseteq \Sigma$$

$$2) \{\lambda\} \neq \emptyset$$

Equality of Strings

□ Definition 6.4

If $w_1, w_2 \in \Sigma^+$, then we may write

$$w_1 = x_1 x_2 \dots x_m \text{ and } w_2 = y_1 y_2 \dots y_n,$$

for $m, n \in \mathbb{Z}^+$ and $x_1, \dots, x_m, y_1, \dots, y_n \in \Sigma$.

We say that the strings w_1 and w_2 are **equal**, and we write $w_1 = w_2$,

if $m = n$ and $x_i = y_i$ for all i .

Length of String

□ Definition 6.5

Let $w = x_1x_2\dots x_n \in \Sigma^+$, where $x_i \in \Sigma$, ($1 \leq i \leq n$).

We define the **length** of w , which is denoted by $\|w\|$, as the value n .

For the case of λ , we have $\|\lambda\| = 0$.

- If $w \in \Sigma^*$ and $\|w\| \geq 1$, then $w \in \Sigma^+$.
- For all $y \in \Sigma^*$, $\|y\| = 1$ if and only if $y \in \Sigma$.
- If an alphabet Σ contains the symbol β (blank), then $\|\beta\| = 1$.

Concatenation of String

□ Definition 6.6

Let $x, y \in \Sigma^+$ with $x = x_1x_2\cdots x_m$ and $y = y_1y_2\cdots y_n$, so that each x_i , for $1 \leq i \leq m$, and each y_j , for $1 \leq j \leq n$, is in Σ .

The **concatenation** of x and y , which we write as xy , is the string $x_1x_2\cdots x_m y_1y_2\cdots y_n$.

Note that

$$x\lambda = x_1x_2\cdots x_m\lambda = x_1x_2\cdots x_m = x,$$

$$\lambda x = \lambda x_1x_2\cdots x_m = x_1x_2\cdots x_m = x.$$

$$\lambda\lambda = \lambda$$

Concatenation of String

□ Length of Concatenated String

$$\|xy\| = \|x\| + \|y\|, \text{ for all } x, y \in \Sigma^*$$

(Examples)

➤ For $\Sigma = \{0,1,2\}$ and elements $x = 01$, $y = 212$,
and $z = xy = 01212$ (in Σ^*),

$$\begin{aligned}\|z\| &= \|01212\| = 5 = 2 + 3 \\ &= \|01\| + \|212\| = \|x\| + \|y\|\end{aligned}$$

➤ $\|x\| = \|x\| + 0 = \|x\| + \|\lambda\| = \|x\lambda\| \quad (= \|\lambda x\|)$

Powers of String

□ Definition 6.7

For each string $x \in \Sigma^*$, we define the **powers of x** by $x^0 = \lambda$, $x^1 = x$, $x^2 = xx$, $x^3 = xx^2$, ... , $x^{n+1} = xx^n$, ... , where $n \in \mathbb{N}$.

(Ex.)

For $\Sigma = \{0,1\}$ and $x = 01$,

- $x^0 = \lambda$, $x^1 = 01$, $x^2 = 0101$, $x^3 = 010101$, ...
- $\|x^0\| = 0$, $\|x^1\| = 2$, $\|x^2\| = 4$, $\|x^3\| = 6$, ...
- For all $n \in \mathbb{N}$, $\|x^n\| = n \|x\|$.

Prefix & Suffix

□ Definition 6.8

If $x, y \in \Sigma^*$ and $w = xy$, then the string x is called a **prefix** of w , and if $y \neq \lambda$, then x is called a **proper prefix**.

Similarly, the string y is called a **suffix** of w , and it is a **proper suffix** when $x \neq \lambda$.

Examples

- For $\Sigma = \{a, b, c\}$ and string $w = abbcc$,
 - Prefix $\rightarrow \lambda, a, ab, abb, abbc, \textbf{abbcc}$
 - Suffix $\rightarrow \lambda, c, cc, bcc, bbcc, \textbf{abbcc}$
- Let $x = x_1x_2x_3\dots x_n$ where $n \in \mathbb{Z}^+$, $x_i \in \Sigma$.
 - Prefix : $\lambda, x_1, x_1x_2, \dots, x_1x_2x_3\dots x_n$
 - Suffix : $\lambda, x_n, x_{n-1}x_n, x_{n-2}x_{n-1}x_n, \dots, x_1x_2x_3\dots x_n$

So x has $n+1$ prefixes, n of which are proper prefixes. How about suffixes ?

Examples

- If $\|x\| = 5$, $\|y\| = 4$ and $w = xy$, then
 - w has x as a proper prefix.
 - w has y as a proper suffix.
 - w has 9 proper prefixes and 9 proper suffixes.
 - xy is both a prefix and a suffix, but in neither case is it proper.

- For a given alphabet Σ , let $w, a, b, c, d \in \Sigma^*$.
If $w = ab = cd$, then
 - 1) a is a prefix of c , or c is a prefix of a .
 - 2) b is a suffix of d , or d is a suffix of b .

Substring

□ Definition 6.9

If $x, y, z \in \Sigma^*$ and $w = xyz$, then y is called a **substring** of w .

When at least one of x and z is different from λ (so that y is different from w), we call y a **proper substring**.

(Ex) For $\Sigma = \{0,1\}$, let $w = 00\boxed{1011}\boxed{10} \in \Sigma^*$.

- 1011 : $w=xyz$, with $x=00$, $y=1011$, and $z=10$.
- 10 : two ways
 - $w=xyz$ where $x=00$, $y=10$, and $z=1110$.
 - $w=xyz$ for $x=001011$, $y=10$, and $z=\lambda$.

Summary 1

	Alphabet	String
Concatenation	(juxtaposition)	w_1w_2
Power	$\Sigma^n, \Sigma^+, \Sigma^*,$ $\Sigma^0 = \{\lambda\}$	$w^n, w^0 = \lambda$ (Empty string)
Substring		y from xyz
Prefix (Suffix)		x (y) from xy
Length, Equality		$\ x\ , x = y$
etc		

Language

□ Definition 6.10

For a given alphabet Σ ,

any subset of Σ^* is called a language over Σ .

This includes the subset \emptyset , which we call the empty language.

(Example)

With $\Sigma = \{0,1\}$, the sets $A = \{1, 01, 001\}$ and $B = \{1, 01, 001, 0001, \dots\}$ are examples of languages over Σ .

Concatenation of Languages

□ Definition 6.11

For an alphabet Σ and languages $A, B \subseteq \Sigma^*$, the **concatenation** of A and B , denoted AB , is
 $\{ ab \mid a \in A, b \in B \}$

(Ex.) Let $\Sigma = \{x, y, z\}$, and let A, B be the finite languages $A = \{x, xy, z\}$, $B = \{\lambda, y\}$.

$AB = \{x, xy, z, xyy, zy\}$ and $BA = \{x, xy, z, yx, yxy, yz\}$

- $|AB| = 5 \neq 6 = |BA|$
- $|AB| = 5 \neq 6 = 3 \cdot 2 = |A||B|$

In general, $|AB| \leq |A||B|$.

Theorem 6.1

□ For an alphabet Σ , let $A, B, C \subseteq \Sigma^*$.

a) $A\{\lambda\} = \{\lambda\}A = A$

b) $(AB)C = A(BC)$

c) $A(B \cup C) = AB \cup AC$

d) $(B \cup C)A = BA \cup CA$

e) $A(B \cap C) \subseteq AB \cap AC$

f) $(B \cap C)A \subseteq BA \cap CA$

$$\Sigma = \{x, y, z\}, \quad A = \{y, yy\}$$

$$B = \{x, xx, y\}, \quad C = \{y, xy\}$$

$$B \cap C = \{y\}$$

$$(B \cap C)A = \{yy, yyy\}$$

$$BA = \{xy, xyy, xxy, xxyy, yy, yyy\}$$

$$CA = \{yy, yyy, xyy, xyxy\}$$

$$BA \cap CA = \{xyy, yy, yyy\}$$

$$\therefore (B \cap C)A \subseteq BA \cap CA$$

Kleene Closure

□ Definition 6.12

For a given language $A \subseteq \Sigma^*$ we can construct other languages as follows:

a) $A^0 = \{\lambda\}$, $A^1 = A$, and

$A^{n+1} = \{ ab \mid a \in A, b \in A^n \}$ for all $n \in \mathbb{Z}^+$

b) $A^+ = \bigcup_{n \in \mathbb{Z}^+} A^n$, positive closure of A

c) $A^* = A^+ \cup A^0 = A^+ \cup \{\lambda\}$.

The language A^* is called the **Kleene closure** of A , in honor of the American logician Stephen Cole Kleene (1909-1994).

Example

□ If $\Sigma = \{x, y, z\}$ and $A = \{x\}$, then

1) $A^0 = \{\lambda\}$

2) $A^n = \{x^n\}$, for each $n \in \mathbb{N}$

3) $A^+ = \{x^n \mid n \geq 1\}$

4) $A^* = \{x^n \mid n \geq 0\}$

Example

- $\Sigma = \{x, y\}$, $A = \{xx, xy, yx, yy\} = \Sigma^2$, $B = \{x, y\}$
- A^* : language of all even-length strings in Σ^*
 - BA^* : language of all odd-length strings in Σ^*
 - $BA^* = A^*B$
 - $\Sigma^* = A^* \cup BA^*$
 - $\{x\}\{x, y\}^*$: language of all strings in Σ^* for which x is a prefix
 - $\{x, y\}^*\{yy\}$: all strings with a suffix yy
 - $\{x, y\}^*\{xxy\}\{x, y\}^*$: those with a substring xxy
 - $\{x\}^*\{y\}^* \subseteq \{x, y\}^*$

Summary 2

	Alphabet	String	Language
Concatenation	(juxtaposition)	$w_1 w_2$	AB
Power	$\Sigma^n, \Sigma^+, \Sigma^*,$ $\Sigma^0 = \{\lambda\}$	$w^n, w^0 = \lambda$	$A^n, A^+, A^*,$ $A^0 = \{\lambda\}$
Substring		y from xyz	
Prefix (Suffix)		x (y) from xy	
Length, Equality		$\ x\ , x = y$	
etc		Empty String	Empty Lang. \emptyset

$$A \subseteq B \rightarrow A^n \subseteq B^n$$

□ Lemma 6.1

Let Σ be an alphabet, with languages $A, B \subseteq \Sigma^*$. If $A \subseteq B$, then for all $n \in \mathbb{Z}^+$, $A^n \subseteq B^n$.

❖ Proof by the mathematical induction

(Basis step) For $n=1$, $A^1=A \subseteq B=B^1$.

(Inductive step) Assume that for $n=k$, $A \subseteq B \rightarrow A^k \subseteq B^k$. Let $x \in A^{k+1}$. From Definition 6.12, $x = x_1 x_k$, where $x_1 \in A$, $x_k \in A^k$. From the induction hypothesis, $x_1 \in B$ and $x_k \in B^k$. Consequently, $x = x_1 x_k \in BB^k = B^{k+1}$ and $A^{k+1} \subseteq B^{k+1}$.

Theorem 6.2

□ For an alphabet Σ and languages $A, B \subseteq \Sigma^*$

a) $A \subseteq AB^*$

b) $A \subseteq B^*A$

c) $A \subseteq B \rightarrow A^+ \subseteq B^+$

d) $A \subseteq B \rightarrow A^* \subseteq B^*$

e) $AA^* = A^*A = A^+$

f) $A^*A^* = A^* = (A^*)^* = (A^*)^+ = (A^+)^*$

g) $(A \cup B)^* = (A^* \cup B^*)^* = (A^*B^*)^*$

Proof of (c) and (d)

$$\square \quad A \subseteq B \rightarrow A^+ \subseteq B^+$$

Let $A \subseteq B$ and $x \in A^+$

Then $x \in A^+ \rightarrow x \in A^n$, for some $n \in \mathbb{Z}^+$

From Lemma 6.1, $x \in B^n \subseteq B^+$

Therefore, $A \subseteq B \rightarrow A^+ \subseteq B^+$

$$\square \quad A \subseteq B \rightarrow A^* \subseteq B^*$$

$A^* = A^+ \cup \{\lambda\}$ and $B^* = B^+ \cup \{\lambda\}$

$A^+ \subseteq B^+ \rightarrow (A^+ \cup \{\lambda\}) \subseteq (B^+ \cup \{\lambda\}) = A^* \subseteq B^*$

Therefore, $A \subseteq B \rightarrow A^* \subseteq B^*$

Proof of (g)

$$\square (A \cup B)^* = (A^* \cup B^*)^*$$

$$\begin{aligned} A \subseteq A^*, B \subseteq B^* &\rightarrow (A \cup B) \subseteq (A^* \cup B^*) \\ &\rightarrow (A \cup B)^* \subseteq (A^* \cup B^*)^* \text{ by (d)} \end{aligned}$$

Conversely,

$$A, B \subseteq A \cup B$$

$$A \subseteq B \rightarrow A^* \subseteq B^*$$

$$\begin{aligned} &\rightarrow A^*, B^* \subseteq (A \cup B)^* \text{ by (d)} \\ &\rightarrow (A^* \cup B^*) \subseteq (A \cup B)^* \text{ by def. of union} \\ &\rightarrow (A^* \cup B^*)^* \subseteq (A \cup B)^* \text{ by (d) and (f)} \end{aligned}$$

$$\text{Therefore, } (A \cup B)^* = (A^* \cup B^*)^*.$$

$$A^* = (A^*)^*$$

Proof of (g)

$$\square (A^* \cup B^*)^* = (A^* B^*)^*$$

$$A^*, B^* \subseteq A^* B^* \quad \text{by (a) and (b)}$$

$$A \subseteq AB^*, \quad A \subseteq B^* A$$

$$\rightarrow (A^* \cup B^*) \subseteq A^* B^* \quad \text{by def. of union}$$

$$\rightarrow (A^* \cup B^*)^* \subseteq (A^* B^*)^* \quad \text{by (d)}$$

Conversely,

For any $xy \in A^* B^*$ where $x \in A^*$ and $y \in B^*$,

$$x, y \in A^* \cup B^* \quad \text{and} \quad xy \in (A^* \cup B^*)^*$$

$$\text{so, } A^* B^* \subseteq (A^* \cup B^*)^*$$

$$\rightarrow (A^* B^*)^* \subseteq (A^* \cup B^*)^* \quad \text{by (d) and (f)}$$

Recursive Def. of Languages

□ A language $L_1 \subseteq \Sigma^*$ where $\Sigma = \{0,1\}$

1) $0 \in L_1$ and

2) $1x, x1 \in L_1$, for each string $x \in L_1$.

□ A language $L_2 \subseteq \Sigma^*$ where $\Sigma = \{ (,) \}$

1) $() \in L_2$ and

2) For all $x, y \in L_2$, $xy \in L_2$ and $(x) \in L_2$.

Regular Expression & Regular Languages



부산대학교
PUSAN NATIONAL UNIVERSITY

Regular Expression

□ Definition

The **regular expression** E over an alphabet Σ is defined recursively as one of the following types.

Base	1. \emptyset ;	(Ex.) $\Sigma = \{0, 1\}$
	2. λ ;	
	3. $a \in \Sigma$;	
	4. $(E_1 + E_2)$;	Recursive process
	5. $(E_1 E_2)$; or	
	6. (E_1^*)	
		0, 1 (0*) (1(0*)) ((1(0*)) + 0)

where E_1 and E_2 are regular expressions.

Operation Priority

□ Priority

- 연산의 우선순위를 다음과 같이 정하면 많은 괄호를 생략할 수 있다.

1. Power (*)

2. Concatenation

3. Or 또는 Union (+)

(Example)

$((1(0^*)) + 0) \rightarrow 10^* + 0$

▪ $\{0, 1, 10, 100, 1000, \dots\}$

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

$((10^*) + 0)$

$(10^* + 0)$

$10^* + 0$ vs. $(10)^* + 0$

Operation Properties

□ Properties

1. Union : commutative & associative

- $r + s = s + r, (r + s) + t = r + (s + t)$

2. Concatenation : associative, not commutative

- $(rs)t = r(st), rs \neq sr$

3. Concatenation over Union : distributive

- $r(s + t) = rs + rt, (r + s)t = rt + st$

Regular Language

□ Definition

A language $L \subseteq \Sigma^*$ is called a **regular language** if there is a regular expression E over Σ with $L = \boxed{L(E)}$. ?

1. If $E = \emptyset$, then $L(E) = \emptyset$,
2. If $E = \lambda$, then $L(E) = \{\lambda\}$,
3. If $E = a \in \Sigma$, then $L(E) = \{a\}$,
4. If $E = (E_1 + E_2)$, then $L(E) = L(E_1) \cup L(E_2)$,
5. If $E = (E_1 E_2)$, then $L(E) = L(E_1)L(E_2)$, and
6. If $E = (E_1^*)$, then $L(E) = L(E_1)^*$.

Language $L(E)$ is the set of all the strings that can be represented as E

Examples

Let $\Sigma = \{0,1\}$ in following examples.

□ Regular expression $(00)^*(11)^*1$ 의 의미는?

➤ 짝수개 0 다음에 홀수개 1이 나오는 스트링 집합

□ Regular language $\{0^n1^m \mid (n+m) \text{은 짝수}\}$ 를 나타내는 regular expression?

1) 0이 짝수개 후, 1이 짝수개인 경우 : $(00)^*(11)^*$

2) 0이 홀수개 후, 1이 홀수개인 경우 : $0(00)^*(11)^*1$

▪ Answer : $(00)^*(11)^* + 0(00)^*(11)^*1$

Examples

□ 1이 한 개 또는 두 개 있는 string의 set을 표현하는 Regular expression?

1) $0^* 10^*$: 1이 한 개

2) $0^* 10^* 10^*$: 1이 두 개 $0^* 110^*$?

▪ Answer : $0^* 10^* + 0^* 10^* 10^*$

□ 길이가 3의 배수인 string의 집합을 나타내는 regular expression?

➤ 길이가 3인 string : $(0+1)(0+1)(0+1) \equiv (0+1)^3$

▪ Answer : $((0+1)^3)^*$

Examples

□ 11을 substring으로 가지는 string의 set을 표현하는 regular expression?

➤ $(0+1)^*11(0+1)^*$

□ 11을 substring으로 가지지 않는 string의 set을 표현하는 regular expression?

1) 0으로만 구성되든지, 또는 1이 나오는 경우 그 다음의 1이 나오기 전에 0이 한 개 이상 있어야 함.

▪ Answer(1) : $0^* + 0^*1(00^*1)^*0^*$

2) 1 다음에는 반드시 0이 나와야 하므로 10이나 0이 반복된 후에 1이 붙든지 붙지 않든지 할 수 있다.

▪ Answer(2) : $(10+0)^*(1+\lambda)$

Equivalence of RE's

□ Equivalence of Regular Expressions

Let E_1 and E_2 be two regular expressions over Σ .
Then E_1 and E_2 are **equivalent** if $L(E_1) = L(E_2)$.

(Ex.)

Are they equivalent?

$0^* + 0^* 1(00^*1)^*0^*$ vs. $(10+0)^*(1+\lambda)$