Formal Languages and Automata

Day 1: Basic concepts

Disclaimer

- Due to the speaker's lack of English proficiency, this material may contain Korean.

- 발표자의 한국어 능력 부족으로,
  이 발표자료에 영어가 섞여 있습니다(?)

두유노

https://www.sitepoint.com/demystifying-regex-with-practical-examples/

Regular expression (regex)

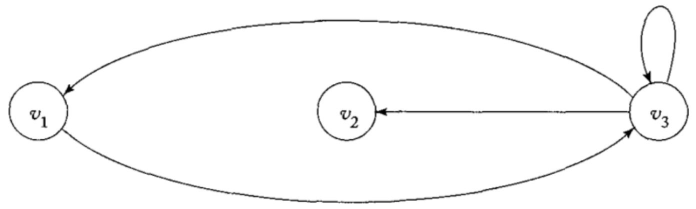
- Some examples (in Javascript `RegExp`)

- `/^[0-9]{2,3}-[0-9]{3,4}-[0-9]{4}$/`
- `/^(((http(s?))\:\/\/)?)([0-9a-zA-Z\-]+\.)+[a-zA-Z]{2,6}(\:[0-9]+)?(\/\S*)?$/`
- `/^(?:[0-9]{2}(?:0[1-9]|1[0-2])(?:0[1-9]|[1,2][0-9]|3[0,1]))-[1-4][0-9]{6}$/`

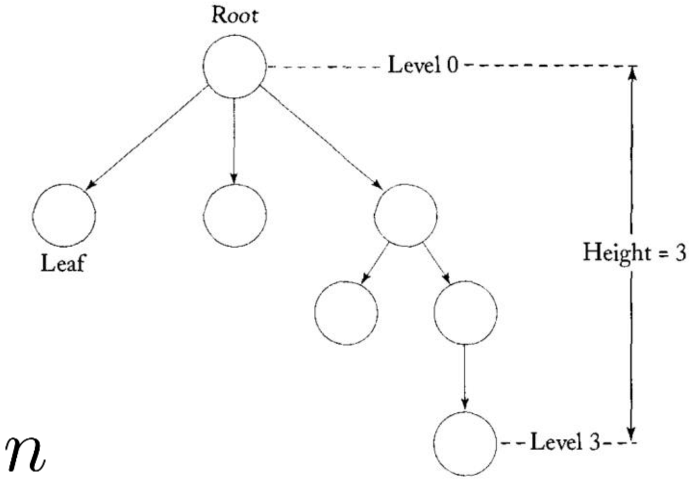
- What it means to be regular?

  → 정규 = 正規 = せいき = …

Mathematical preliminaries



- Sets
- Functions
- Graphs and trees
- Proof techniques (induction / contradiction)



[Example 1] Prove that a binary tree of height n has at most 2^n leaves.

Basic concepts

- Languages
- Grammars
- Automata

Languages

- 한국어, English, 日本語, 汉语, python, …
- **Formal** languages – <u>precise</u>!

- Alphabet Σ – finite, nonempty set
- Strings – finite sequences of symbols from the alphabet

$$\boxed{\Sigma = \{a, b\}} \rightarrow \boxed{abab,\ aaabbba,\ \cdots}$$

Strings

$$w = a_1 a_2 \cdots a_n$$
$$v = b_1 b_2 \cdots b_m$$

- Concatenation $wv = a_1 a_2 \cdots a_n b_1 b_2 \cdots b_m$
- Reverse $w^R = a_n \cdots a_2 a_1$
- Length $|w| = n$
- Empty string $\lambda \Rightarrow |\lambda| = 0,\ \lambda w = w\lambda = w$

Strings – cont'd

- Substring of w – any string of consecutive characters in w
- $w = vu \Rightarrow v$ is prefix, u is suffix

[Example 2] Prove that $|uv| = |u| + |v|$.

Strings – cont'd

- w^n – string obtained by repeating w n times
- Special case – $w^0 = \lambda$

- Σ^* – strings obtained by concatenating
  zero or more symbols from Σ
- $\Sigma^+ = \Sigma^* - \{\lambda\}$
- **Language**: subset of Σ^* (broad definition)
- Sentence: string in a language

Strings – cont'd

- Let $\Sigma = \{a, b\}$. Then
$$\Sigma^* = \{\lambda, a, b, aa, ab, ba, bb, aaa, aab, \cdots\}.$$

- The set $\{a, aa, aab\}$ is a language on Σ.
- The set $L = \{a^n b^n : n \geq 0\}$ is also a language on Σ.

Languages

- Complement $\bar{L} = \Sigma^* - L$
- Reverse $L^R = \{w^R : w \in L\}$
- Concatenation $L_1 L_2 = \{xy : x \in L_1, y \in L_2\}$
- L^n : easy $L^0 = \{\lambda\}$

- Star-closure $L^* = L^0 \cup L^1 \cup L^2 \cdots$ [Kleene closure]
- Positive closure $L^+ = L^1 \cup L^2 \cdots$ [Kleene plus]

Languages – cont'd

[Example 3] If $L = \{a^n b^n : n \geq 0\}$,

$$L^2 =$$

$$L^R =$$

$$\bar{L} =$$

$$L^* =$$

Limitation of set notation?

Grammars

- A **grammar** G is defined as a quadruple

$$G = (V, T, S, P),$$

where V is a finite set of objects called **variables**,
T is a finite set of objects called **terminal symbols**,
$S \in V$ is a special symbol called the **start** variable,
P is a finite set of **productions**.

Sets V, T are nonempty and disjoint.

Grammars – cont'd

- $x \in (V \cup T)^+,\, y \in (V \cup T)^*$

$$w = uxv \xrightarrow{\text{Production rule } x \to y} z = uyv$$

- This is written as $w \Rightarrow z$: w derives z

Grammars – cont'd

$$w_1 \Rightarrow w_2 \Rightarrow \cdots \Rightarrow w_n$$

- This is written as $w_1 \overset{*}{\Rightarrow} w_n$: w_1 derives w_n
- $w \overset{*}{\Rightarrow} w$

Grammars – cont'd

- Let $G = (V, T, S, P)$ be a grammar. Then the set

$$L(G) = \left\{ w \in T^* : S \overset{*}{\Rightarrow} w \right\}$$

is the language generated by G.

- If $w \in L(G)$, then the sequence

$$S \Rightarrow w_1 \Rightarrow w_2 \Rightarrow \cdots \Rightarrow w_n \Rightarrow w$$

is a **derivation** of the sentence w.

Grammars – cont'd

sentential forms of the derivation

$$S \Rightarrow w_1 \Rightarrow w_2 \Rightarrow \cdots \Rightarrow w_n \Rightarrow w$$

Grammars – cont'd

Consider the grammar

$$G = (\{S\}, \{a, b\}, S, P),$$

with P given by

$$\begin{aligned} &S \to aSb, \\ &S \to \lambda. \end{aligned}$$

Then $S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aabb$, so we can write

$$S \stackrel{*}{\Rightarrow} aabb.$$

Conjecture: $L(G) = \{a^n b^n : n \geq 0\}$ – How can we prove it?

Grammars – cont'd

[Example 4] Find a grammar that generates

$$L = \{a^n b^{n+1} : n \geq 0\}.$$

Grammars – cont'd

[Example 5]
Take $\Sigma = \{a, b\}$, and let $n_a(w)$, $n_b(w)$ denote the number of a, b in the string w. Prove that the grammar G with productions

$$\begin{array}{ll} S \to SS, & S \to \lambda, \\ S \to aSb, & S \to bSa, \end{array} \qquad (= S \to SS|\lambda|aSb|bSa)$$

generates the language

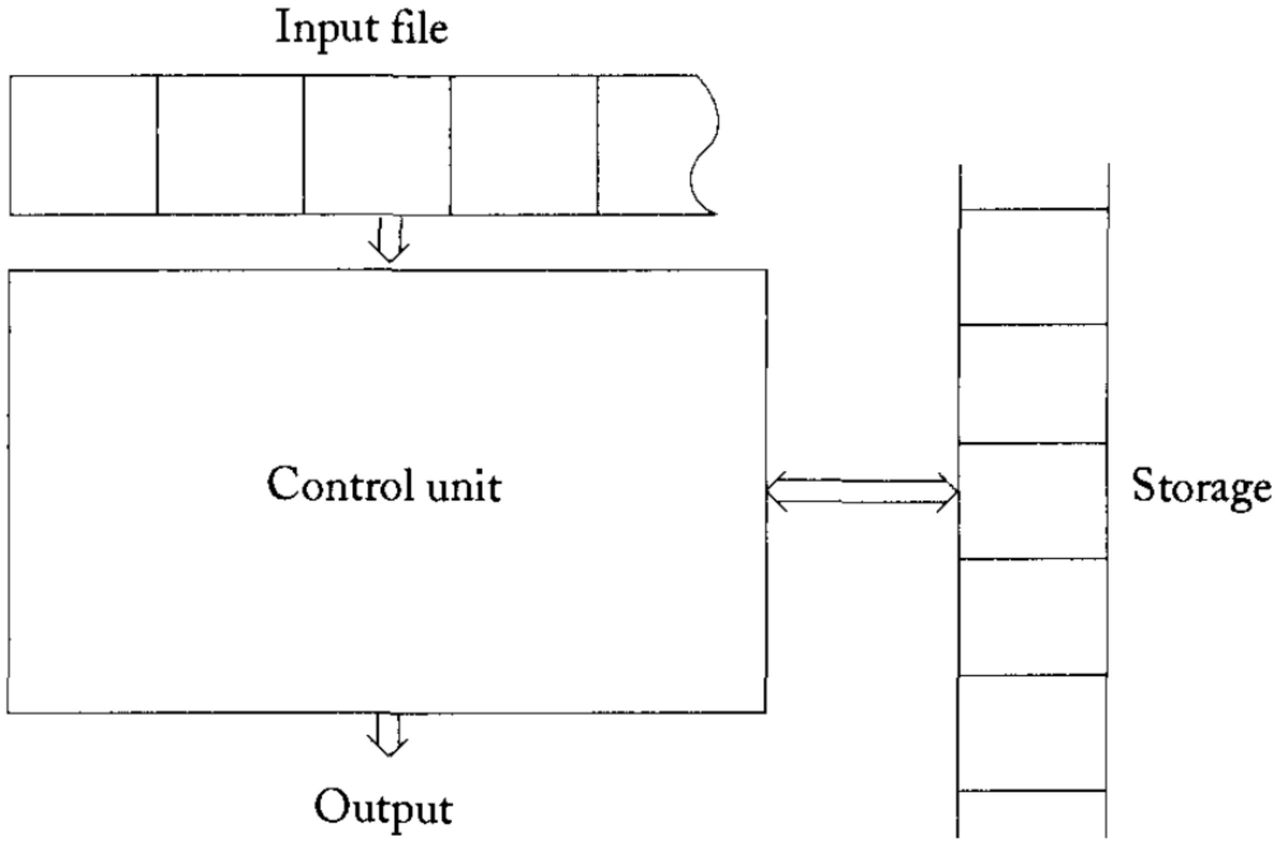
$$L = \{w : n_a(w) = n_b(w)\}.$$

Automata

- Input file
- Storage
- Control unit

- Internal states
- Transition function

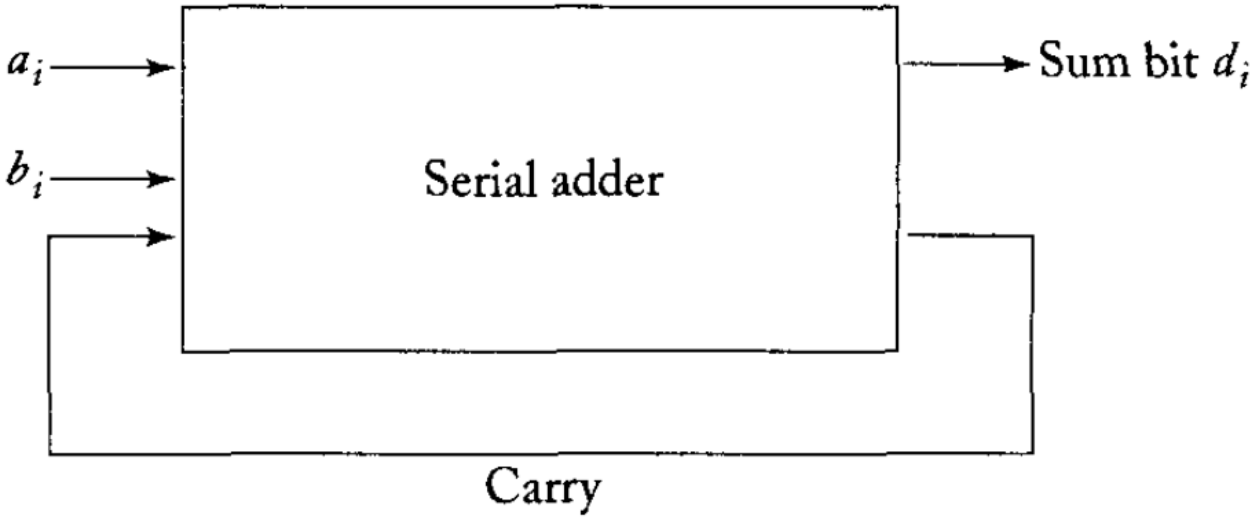
- Configuration
- Move



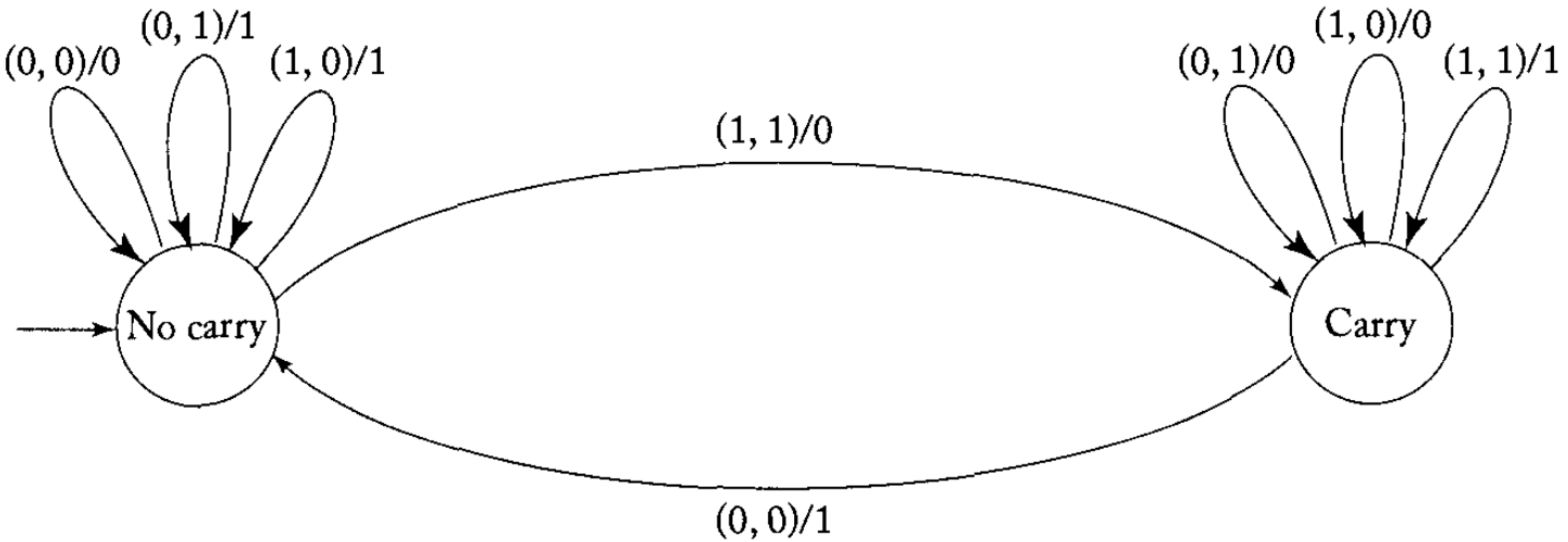
Automata – cont'd

- Bit string $x = a_0 a_1 \cdots a_n \Rightarrow v(x) = \sum_{i=0}^{n} a_i 2^i$
- Binary adder: adding $x = a_0 a_1 \cdots a_n,\ y = b_0 b_1 \cdots b_n$ bit by bit

a_i \ b_i	0	1
0	0 No carry	1 No carry
1	1 No carry	0 Carry



Automata – cont'd



Bonus – Little abstract algebra

Monoid $(S, \cdot)$

- Associativity $a, b, c \in S \Rightarrow (a \cdot b) \cdot c = a \cdot (b \cdot c)$
- Identity element $\exists e \in S \text{ s.t. } a \in S \Rightarrow e \cdot a = a \cdot e = a$

- Σ^* : free monoid generated by Σ [Kleene star of Σ]
- λ : identity element 1_{Σ^*}

$$(\Sigma^*, \cdot) \xrightarrow[\text{“Monoid Homomorphism"}]{\text{length function } |\cdot|} (\mathbb{N}_0, +)$$

Next seminar

- Deterministic Finite Accepters (DFA)
- Transition Graph
- <u>Regular</u> Languages

- Nondeterministic Finite Accepters (NFA)
- Equivalence between [ ] and [ ]

- Reduction of the Number of States