Lecture 2
Languages, Grammars, and Automata

COT 4420

Theory of Computation

Section 1.2 , 1.3

Languages
Definitions

- Any finite, nonempty set of symbols is an alphabet or vocabulary.

  Σ = {A, B, C, D, …, Z}

  Σ = {0, 1}

  Σ = { □, if, then, else}

- A finite sequence of symbols from the alphabet is called a string or a word or a sentence.

  w = ALPHA

  w = 0100011101

Languages
Definitions

- Two strings can be concatenated to form another string:

  v = ALPHA, w = BETA

  Concat(v, w) = vw = ALPHABETA

- The length of a string w, denoted by $|w|$ is the number of symbols in the string.

  |ALPHA| = 5

- The empty string is denoted by λ or ε and its length is 0.

  $|\lambda| = 0$

Languages
Definitions

- If Σ is the alphabet, Σ^* is the set of all strings over Σ, including the empty string.
- Σ^* is obtained by concatenating zero or more symbols from Σ.

$$\Sigma^+ = \Sigma^* - \{\lambda\}$$

Let $\Sigma = \{a, b, c, d\}$, what is Σ^* ?

Can you specify a procedure to generate Σ^* ?

What is $|\Sigma^*|$?

Languages
Definitions

- A language over Σ is a subset of Σ^*.

  $L \subseteq \Sigma^*$

Example: $\Sigma = \{a, b\}$

$L_1 = \{ a, aa, aba\}$ a finite language

$L_2 = \{ a^n b^n : n \geq 1\}$ an infinite language

Ways to represent languages

1. Recognition point of view
   - Give a <u>procedure</u> which

     says Yes for sentences in the language,   and

     either does not terminate or says No for sentences NOT in the language.

❖ The procedure recognizes the language

Ways to represent languages

2. Generation point of view
   - Systematically generate (enumerate) all sentences of the language

- What's the relationship between these two points of view?

Ways to represent languages

Given a procedure to recognize L, we can give a procedure for generating L.

Steps

Strings	**1**	**2**	**3**	**4**	**…**
$\mathbf{x_1}$	1	3	6	10	15
$\mathbf{x_2}$	2	5	9	14	
$\mathbf{x_3}$	4	8	13		
$\mathbf{x_4}$	7	12			
⋮	11				

1. Run step 1 on x_1
2. Run step 1 on x_2
3. Run step 2 on x_1
4. Run step 1 on x_3
5. …

Ways to represent languages

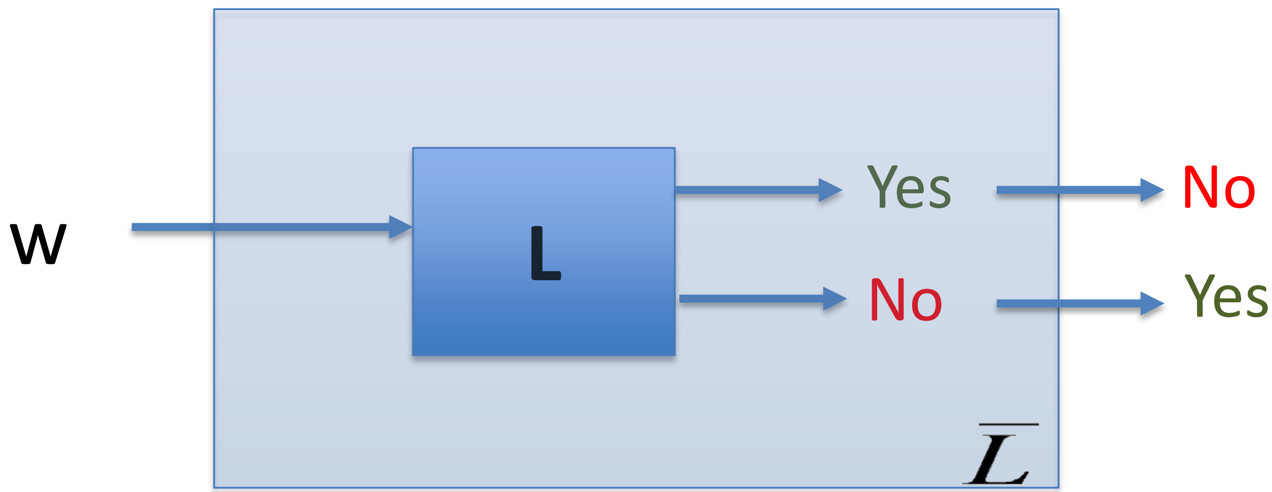
Given a procedure for generating L, we can give a procedure for recognizing L. what is it?

Definitions

- A language L that can be generated by a <u>procedure</u> is said to be a recursively enumerable set or **RE**.
  - – It accepts $w \in L$, but we do not know what happens for $w \notin L$. (It may halt or goes into an infinite loop)

- A language L that can be recognized by an <u>algorithm</u> is said to be recursive or **R**.
  - – Halts on every $w \in \Sigma^+$.

- Recursive sets are a subset of RE.
- Suppose *L* is recursive, how about $\overline{L}$?



Automata

- An automaton is an abstract model of a digital computer.
- Reads the input (string over the alphabet)
- Has a control unit which can be in any of the finite number of internal states and can change state in some defined manner.
- Given an input string, it outputs yes or no meaning that it either accepts the string or rejects it.

Grammars
Definitions

- A grammar is a method to describe and generate the sentences of a language.
- A **grammar** G is defined as a quadruple

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

V is a finite set of **variables**

T is a finite set of **terminal** symbols

S $\in$ V is a special variable called **start symbol**

P is a finite set of **production rules** of the form

$$x \rightarrow y$$

$$\text{where } x \in (V \cup T)^+ ,\ \ y \in (V \cup T)^*$$

Grammars
Example

S → <noun phrase> <verb phrase>

<noun phrase> → <article> <noun>

<article> → the

<noun> → dog

<verb phrase> → is <adjective>

<adjective> → happy

S => <noun phrase><verb phrase> => <article><noun><verb phrase>
=> the <noun><verb phrase> => the <noun> is <adjective> =>
the dog is <adjective> => the dog is happy

Grammars
Definitions

- We say that w **derives** z if $w = uxv$, and $z = uyv$

and $x \rightarrow y \in P$

$$w \Rightarrow z$$

- If $w_1 \Rightarrow w_2 \Rightarrow \ldots \Rightarrow w_n$ we say $w_1 \Rightarrow^* w_n$ (derives in zero or more steps)
- The set of **sentential forms** is

$$S(G) = \{\alpha \in (V \cup T)^* \mid S \Rightarrow^* \alpha\}$$

- The language generated by grammar G is

$$L(G) = \{ w \in T^* \mid S \Rightarrow^* w \}$$

Grammars
Example

G = (V, T, P, S) V = {S, B, C} T = {a, b, c}

P:

$S \rightarrow aSBC$ $bB \rightarrow bb$

$S \rightarrow aBC$ $bC \rightarrow bc$

$CB \rightarrow BC$ $cC \rightarrow cc$

$aB \rightarrow ab$

S =>* aaBCBC sentential form

What is L(G) ? $L(G) = \{ a^n b^n c^n \mid n \geq 1\}$

Grammars
Example

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

Productions:

$S \rightarrow aSb$

$S \rightarrow \lambda$

What is the L(G)? $L = \{a^n b^n : n \geq 0\}$

Grammars
Example

Find a grammar that generates

$$L = \{ a^n b^{2n} : n \geq 0 \}$$

$$S \rightarrow aSbb \mid \lambda$$

Summary

- An automaton recognizes (or accepts) a language
- A grammar generates a language
- For some grammars, it is possible to build an automaton M_G from the grammar G so that M_G recognizes the language L(G) generated by the grammar G.