بسم الله الرحمن الرحيم

Introduction

Dr. Nasir Jalal PhD (Computer Science)

Lecturer

CS & IT Department, CUVAS Bahawalpur

Course Detail

BS CS 5th Semester Fall 2022-2026

Course: Theory of Programming Languages

Course Code: CS IT 507

Lecture: 5 (19-09-2024)

Outline Lecture 5

- Revision of Lecture 4
- Ways of Defining Languages
- Descriptive definition of language

Defining Languages

The languages can be defined in different ways , such as

- Descriptive definition,
- Recursive definition,
- using Regular Expressions(RE) and
- using Finite Automaton(FA) etc.

Descriptive definition of language

The language is defined, describing the conditions imposed on its words.

- **Example:**

  The language L of strings of odd length, defined over Σ={a}, can be written as

  L={a, aaa, aaaaa,…..}

- **Example:**

  The language L of strings that does not start with a, defined over Σ={a,b,c}, can be written as

  L={b, c, ba, bb, bc, ca, cb, cc, …}

Descriptive definition of language

- **Example:** The language **EQUAL**, of strings with number of a's equal to number of b's, defined over Σ={a,b}, can be written as

  {Λ ,ab,aabb,abab,baba,abba,…}
- **Example:** The language **EVEN-EVEN**, of strings with even number of a's and even number of b's, defined over Σ={a,b}, can be written as

  {Λ, aa, bb, aaaa, aabb, abab, abba, baab, baba, bbaa, bbbb,…}

Descriptive definition of language

- **Example:** The language **factorial**, of strings defined over Σ={1,2,3,4,5,6,7,8,9} *i.e.* {1,2,6,24,120,…}
- **Example:** The language **FACTORIAL**, of strings defined over Σ={a}, as $\{a^{n!} : n=1,2,3,\ldots\}$, can be written as {a,aa,aaaaaa,…}. It is to be noted that the language FACTORIAL can be defined over any single letter alphabet.

Kleene Star Closure

- Given Σ, then the Kleene Star Closure of the alphabet Σ, denoted by Σ^*, is the collection of all strings defined over Σ, including Λ.
- It is to be noted that Kleene Star Closure can be defined over any set of strings.

Examples

- If $\Sigma = \{x\}$

  Then $\Sigma^* = \{\Lambda, x, xx, xxx, xxxx, \ldots.\}$
- If $\Sigma = \{0,1\}$

  Then $\Sigma^* = \{\Lambda, 0, 1, 00, 01, 10, 11, \ldots.\}$
- If $\Sigma = \{aaB, c\}$

  Then $\Sigma^* = \{\Lambda, aaB, c, aaBaaB, aaBc, caaB, cc, \ldots.\}$

Note

- Languages generated by Kleene Star Closure of set of strings, are infinite languages. (By infinite language, it is supposed that the language contains infinite many words, each of finite length).

PLUS Operation ($^+$)

- Plus Operation is same as Kleene Star Closure except that it does not generate Λ (null string), automatically.

Example:

- If $\Sigma = \{0,1\}$

  Then $\Sigma^+ = \{0, 1, 00, 01, 10, 11, \ldots.\}$

- If $\Sigma = \{aab, c\}$

  Then $\Sigma^+ = \{aab, c, aabaab, aabc, caab, cc, \ldots.\}$

Remarks

- It is to be noted that Kleene Star can also be operated on any string *i.e.* a^* can be considered to be all possible strings defined over {a}, which shows that a^* generates

  Λ, a, aa, aaa, …

- It may also be noted that a^+ can be considered to be all possible non empty strings defined over {a}, which shows that a^+ generates

  a, aa, aaa, aaaa, …

Defining Languages Continued...

Recursive definition of languages

The following three steps are used in recursive definition

1. Some basic words are specified in the language.
2. Rules for constructing more words are defined in the language.
3. No strings except those constructed in above, are allowed to be in the language.

Example-**Recursive definition of language**

- **Defining language of INTEGER**

<u>Step 1:</u>

1 is in **INTEGER**.

<u>Step 2:</u>

If x is in **INTEGER** then x+1 and x-1 are also in **INTEGER**.

<u>Step 3:</u>

No strings except those constructed in above, are allowed to be in **INTEGER**.

Example-**Recursive definition of languag**

- **Defining language of EVEN**

<u>Step 1:</u>

2 is in **EVEN**.

<u>Step 2:</u>

If x is in **EVEN** then x+2 and x-2 are also in **EVEN**.

<u>Step 3:</u>

No strings except those constructed in above, are allowed to be in **EVEN**.

Example-**Recursive definition of languages**

- **Defining the language factorial**

Step 1:

As 0!=1, so 1 is in **factorial**.

Step 2:

n!=n*(n-1)! is in **factorial**.

Step 3:

No strings except those constructed in above, are allowed to be in **factorial**.

Example-**Recursive definition of languages**

- **Defining the language PALINDROME, defined over** Σ = {a,b}

Step 1:

a and b are in **PALINDROME**

Step 2:

if x is palindrome, then s(x)Rev(s) and xx will also be palindrome, where s belongs to Σ^*

Step 3:

No strings except those constructed in above, are allowed to be in palindrome

Regular Expression

	Regular Expression	Regular Languages
set of vowels	(a U e U i U o U u)	{a, e, i, o, u}
a followed by 0 or more b	$(a.b^*)$	{a, ab, abb, abbb, abbbb,….}
any no. of vowels followed by any no. of consonants	$v^*.c^*$ (where v – vowels and c – consonants)	{ ε , a ,aou, aiou, b, abcd…..} where ε represent empty string (in case 0 vowels and o consonants)

Regular Expression

- As discussed earlier that a^* generates

  Λ, a, aa, aaa, …

  and a^+ generates a, aa, aaa, aaaa, …, so the language L_1 = {Λ, a, aa, aaa, …} and L_2 = {a, aa, aaa, aaaa, …} can simply be expressed by a^* and a^+, respectively.

  a^* and a^+ are called the regular expressions (RE) for L_1 and L_2 respectively.

  Note: a^+, aa^* and a^*a generate L_2.

Defining Languages (continued)…

- **Method 3 (Regular Expressions)**
  - Consider the language L={Λ, x, xx, xxx,…} of strings, defined over Σ = {x}.

    We can write this language as the Kleene star closure of alphabet Σ or $L=\Sigma^{*}=\{x\}^{*}$

    this language can also be expressed by the regular expression x^{*}.
  - Similarly the language L={x, xx, xxx,…}, defined over Σ = {x}, can be expressed by the regular expression x^{+}.

– Now consider another language L, consisting of all possible strings, defined over Σ = {a, b}. This language can also be expressed by the regular expression

$(a + b)^*$.

– Now consider another language L, of strings having exactly double a, defined over Σ = {a, b}, then it's regular expression may be

b^*aab^*

- Now consider another language L, of even length, defined over Σ = {a, b}, then it's regular expression may be

$$((a+b)(a+b))^*$$

- Now consider another language L, of odd length, defined over Σ = {a, b}, then it's regular expression may be

$$(a+b)((a+b)(a+b))^* \text{ or}$$

$$((a+b)(a+b))^*(a+b)$$

Remark

- It may be noted that a language may be expressed by more than one regular expressions, while given a regular expression there exist a unique language generated by that regular expression.

Task

- Q1) Let S={ab, bb} and T={ab, bb, bbb} Show that $S^* \neq T^*$ But $S^* \subset T^*$

  Solution: Since $S \subset T$, so every string belonging to S^* , also belongs to T^* but bbb is a string belongs to T^* but does not belong to S^*.

Task

- 2) Let S={a, bb, bab, abaab} be a set of strings. Are abbabaabab and baabbbabbaabb in S^*? Does any word in S^* have odd number of b’s?

  Solution: since abbabaabab can be grouped as (a)(bb)(abaab)ab , which shows that the last member of the group does not belong to S, so abbabaabab is not in S^*, while baabbbabbaabb can not be grouped as members of S, hence baabbbabbaabb is not in S^*. Since each string in S has even number of b’s so there is no possiblity of any string with odd number of b’s to be in S^*.

Task

Q3)Is there any case when S^+ contains Λ? If yes then justify your answer.

Solution: consider S={Λ,a} then

S^+ ={Λ, a, aa, aaa, …}

Here Λ is in S^+ as member of S. Thus Λ will be in S^+, in this case.

Task

Q4) Prove that for any set of strings S

i. $(S^+)^*=(S^*)^*$

Solution: In general Λ is not in S^+, while Λ does belong to S^*. Obviously Λ will now be in $(S^+)^*$, while $(S^*)^*$ and S^* generate the same set of strings. Hence $(S^+)^*=(S^*)^*$.

Q4) continued…

ii) $(S^+)^+=S^+$

Solution: since S^+ generates all possible strings that can be obtained by concatenating the strings of S, so $(S^+)^+$ generates all possible strings that can be obtained by concatenating the strings of S^+, will not generate any new string.

Hence $(S^+)^+=S^+$

Q4) continued…

iii) Is $(S^*)^+=(S^+)^*$

Solution: since Λ belongs to S^* ,so Λ will belong to $(S^*)^+$ as member of S^* .Moreover Λ may not belong to S^+, in general, while Λ will automatically belong to $(S^+)^*$.

Hence $(S^*)^+=(S^+)^*$

THANKS