23MT2014

THEORY OF COMPUTATION

Topic:

REGULAR EXPRESSIONS

Session - 6

AIM OF THE SESSION

To introduce students to the theory and applications of regular expressions, enabling them to understand the fundamental concepts and techniques related to regular expressions and their relationship with automata theory.

INSTRUCTIONAL OBJECTIVES

This Session is designed to:

1. To enable students to construct regular expressions that represent specific languages and patterns.
2. To provide students with the ability to convert regular expressions to equivalent finite automata and vice versa.

LEARNING OUTCOMES

At the end of this session, you should be able to:

1. Understand the concept of regular expressions and their syntax and semantics.
2. Construct regular expressions that represent specific languages and patterns.
3. Convert regular expressions to equivalent finite automata and vice versa, demonstrating the equivalence between the two formalisms.

Regular Expressions

- Regular expressions
- describe regular languages

- Example:

$$(a+b\cdot c)^*$$

- describes the language

$$\{a,bc\}^* = \{\lambda, a, bc, aa, abc, bca, \ldots\}$$

Recursive Definition

Primitive regular expressions: $\varnothing, \quad \lambda, \quad \alpha$

Given regular expressions r_1 and r_2

$r_1 + r_2$

$r_1 \cdot r_2$

r_1^*

(r_1)

Are regular expressions

Examples

A regular expression:

$$(a+b\cdot c)^*\cdot(c+\varnothing)$$

Not a regular expression:

$$(a+b+)$$

Languages of Regular Expressions

- $L(r)$: language of regular expression r

- Example

- $L((a+b\cdot c)^*)=\{\lambda,a,bc,aa,abc,bca,\ldots\}$

Definition

- For primitive regular expressions:

$$L(\varnothing) = \varnothing$$

$$L(\lambda) = \{\lambda\}$$

$$L(a) = \{a\}$$

Definition (continued)

- For regular expressions r_1 and r_2
-

$$L(r_1 + r_2) = L(r_1) \cup L(r_2)$$

$$L(r_1 \cdot r_2) = L(r_1)\, L(r_2)$$

$$L(r_1 *) = (L(r_1))*$$

$$L((r_1)) = L(r_1)$$

Example

- Regular expression: $(a+b)\cdot a*$

$$
\begin{aligned}
L((a+b)\cdot a*) &= L((a+b))\,L(a*) \\
&= L(a+b)\,L(a*) \\
&= (L(a)\cup L(b))(L(a))* \\
&= (\{a\}\cup\{b\})(\{a\})* \\
&= \{a,b\}\{\lambda,a,aa,aaa,\ldots\} \\
&= \{a,aa,aaa,\ldots,b,ba,baa,\ldots\}
\end{aligned}
$$

Example

- Regular expression

$$r = (a+b)^*(a+bb)$$

$$L(r) = \{a, bb, aa, abb, ba, bbb, \ldots\}$$

Example

- Regular expression

$$r = (aa)^*(bb)^*b$$

$$L(r) = \{a^{2n}b^{2m}b : \quad n, m \geq 0\}$$

Example

- Regular expression

$$r = (0+1)^*00(0+1)^*$$

$L(r)$ = { all strings with at least two consecutive 0 }

Example

- Regular expression

$$r = (1+01)^*(0+\lambda)$$

$L(r)$ = { all strings without two consecutive 0 }

Equivalent Regular Expressions

- Definition:
  - Regular expressions r_1 and r_2
  - are **equivalent** if $L(r_1) = L(r_2)$

Example

- L = { all strings without two consecutive 0 }

$$r_1 = (1+01)^*(0+\lambda)$$

$$r_2 = (1^*011^*)^*(0+\lambda)+1^*(0+\lambda)$$

$$L(r_1) = L(r_2) = L \Rightarrow$$

r_1 and r_2 are equivalent regular expr.

QUIZ TIME

Which of the following statements is true about regular expressions?

a) Regular expressions can only describe regular languages.

b) Regular expressions can describe both regular and non-regular languages.

c) Regular expressions can only describe context-free languages.

d) Regular expressions can describe any type of language.

Answer: a) Regular expressions can only describe regular languages.

QUIZ TIME

Which operator in regular expressions represents zero or more occurrences of the preceding element?

a) +

b) ?

c) *

d) .

Answer: c) *

QUIZ TIME

Which of the following regular expressions matches any single character?

a) .

b) *

c) +

d) ?

Answer: a) .

QUIZ TIME

Which operator in regular expressions represents the logical OR operation?

a) |

b) ^

c) $

d) !

Answer: a) |

QUIZ TIME

Which of the following regular expressions matches the beginning of a line?

a) ^

b) $

c) %

d) !

Answer: a) ^

Question 1: What is the purpose of using anchors in regular expressions? Answer: Anchors allow us to match specific positions within a string. The "^" anchor matches the start of a line or string, while the "$" anchor matches the end of a line or string.

Question 2: What is the difference between the "*" and "+" operators in regular expressions? Answer: The "*" operator matches the preceding character or subexpression zero or more times, while the "+" operator matches the preceding character or subexpression one or more times.

Question 3: What does the "|" operator represent in regular expressions? Answer: The "|" operator allows for the logical OR operation. It matches either the expression on its left or the expression on its right.

Question 4: What is the purpose of using quantifiers in regular expressions? Answer: Quantifiers allow us to specify the number of times a character or subexpression should occur. Examples include "*", which matches zero or more occurrences, and "{n}", which matches exactly n occurrences.

Question 5: What is the purpose of using character sets in regular expressions? Answer: Character sets allow us to specify a range or set of characters that can be matched at a particular position. For example, [a-z] matches any lowercase letter.

Question 6: What is the significance of the "\b" metacharacter in regular expressions? Answer: The "\b" metacharacter matches a word boundary, indicating the position between a word character and a non-word character.

Question 7: What is the purpose of using capturing groups in regular expressions? Answer: Capturing groups allow us to capture and store matched substrings for later use. They are defined by enclosing the desired pattern in parentheses.

Terminal Questions

1. What are regular expressions?
2. How are regular expressions used in text processing?
3. What is the role of regular expressions in pattern matching?
4. What are regular languages?
5. Can regular expressions represent all regular languages?
6. How are regular expressions constructed?
7. What are some commonly used operators in regular expressions?
8. Can regular expressions handle complex patterns and structures?
9. How do regular expressions contribute to efficient text processing?
10. Are regular expressions language-specific?

THANK YOU

(DEEMED TO BE UNIVERSITY)

Team - TOC