310241: Theory of Computation

Theory of Computation

- Course Objectives:
  - To Study abstract computing models
  - To learn Grammar and Turing Machine
  - To learn about the theory of computability and complexity

Theory of Computation

- Course Outcomes: On completion of the course, student will be able to–
- design deterministic Turing machine for all inputs and all outputs
- subdivide problem space based on input subdivision using constraints
- apply linguistic theory

Unit - 1

- Introduction to Formal language,
- introduction to language translation logic, Essentials of translation,
- Alphabets and languages, Finite representation of language,
- Finite Automata (FA): An Informal Picture of FA, Finite State Machine (FSM), Language accepted by FA,
- Definition of Regular Language, Deterministic and Nondeterministic FA(DFA and NFA), epsilon- NFA,
- FA with output: Moore and Mealy machines - Definition, models, inter-conversion.
- Case Study: FSM for vending machine, spell checker Unit

Introduction to Theory of Computation

- One of the most fundamental course of Computer Engineering.
- Help to understand how people have thought about **computer science** as a **science** in past 50 years.
- Its is mainly about what kind of things can you **compute mechanically with machines** , **how fast** and **how much space** does it take to do so.

Example -1

- Lets consider a machine that **accepts all binary strings that ends with ‘0’** and reject all other strings that do not end with ‘0’

Eg. 11010010 [Accepts]

10011001 [Rejects]

Example-2

- Lets consider a machine that **accepts all valid java codes.**

Java code → Binary Equivalent of code -> Valid ? [Accepts]

Invalid [Rejects]

Can We design such a system ?????

Example-2 (Cont..)

Yes……

Eg. Compiler

We know compile only accepts valid code and if it is not written correctly ,then it gives error and says It's invalid.

By now u must have got sligh ıt of TOC and Compiler relation.

Basic Definitions

1. Alphabet - a finite set of symbols.
   - Notation: Σ .
   - Examples: Binary alphabet {0,1},
     English alphabet {a,...,z,!,?,...}

2. String over an alphabet Σ - a finite sequence of symbols
   from Σ.
   - Notation: (a) Letters u, v, w, x, y, and z denote strings.
     (b) Convention: concatenate the symbols. No parentheses or commas used.
   - Examples: 0000 is a string over the binary alphabet.
     a!? is a string over the English alphabet.

Definitions (contd.)

3. Empty string: e or ε denotes the empty sequence of symbols.
4. Language over alphabet Σ - a set of strings over Σ.
   - Notation: L.
   - Examples:
     - {0, 00, 000, ...} is an "infinite" language over the binary alphabet.
     - {a, b, c} is a "finite" language over the English alphabet.

Definitions (contd.)

5. Empty language - empty set of strings.
   Notation: Φ.
6. Binary operation on strings:
   Concatenation of two strings u.v - concatenate the symbols of u and v.
   - Notation: uv
   - Examples:
     - 00.11 = 0011.
     - ε.u = u.ε = u for every u. (identity for concatenation)

Languages

Language: a set of strings

String: a sequence of symbols from some alphabet

Example:

Strings: cat, dog, house

Language: {cat, dog, house}

Alphabet: $\Sigma = \{a, b, c, \ldots, z\}$

Alphabets and Strings

An alphabet is a set of symbols

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

A string is a sequence of symbols from the alphabet

Example Strings

a

ab

$abba$

$aaabbbaabab$

$u = ab$

$v = bbbaaa$

$w = abba$

Languages are used to describe computation problems:

$$PRIMES = \{2,3,5,7,11,13,17,\ldots\}$$

$$EVEN = \{0,2,4,6,\ldots\}$$

Alphabet: $\Sigma = \{0,1,2,\ldots,9\}$

Decimal numbers alphabet :

$$\Sigma = \{0,1,2,\ldots,9\}$$

String :

102345

567463386

Binary numbers alphabet :

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

String

10001000

1011011

String Operations

$$w = a_1 a_2 \cdots a_n \qquad abba$$

$$v = b_1 b_2 \cdots b_m \qquad bbbaa$$

Concatenation

$$wv = a_1 a_2 \cdots a_n b_1 b_2 \cdots b_m \qquad abbabbbaa$$

$w = a_1 a_2 \cdots a_n$ $\qquad$ $ababaaabb$

Reverse

$w^R = a_n \cdots a_2 a_1$ $\qquad$ $bbbaaaba$

String Length

$$w = a_1 a_2 \cdots a_n$$

Length: $|w| = n$

Examples:

$$|abba| = 4$$

$$|aa| = 2$$

$$|a| = 1$$

Length of Concatenation

$$|uv| = |u| + |v|$$

Example:

$$u = aab, \quad |u| = 3$$

$$v = abaab, \quad |v| = 5$$

$$|uv| = |aababaab| = 8$$

$$|uv| = |u| + |v| = 3 + 5 = 8$$

Empty String

A string with no letters is denoted: λ or ε

Observations:

$$|\lambda| = 0$$

$$\lambda w = w\lambda = w$$

$$\lambda abba = abba\lambda = ab\lambda ba = abba$$

Substring

Substring of string:

a subsequence of consecutive characters

String	Substring
$abba$	ab
$abba$	abb
$abba$	b
$abba$	bba

Prefix and Suffix

$$abbab$$

Prefixes	Suffixes
λ	$abbab$
a	$bbab$
ab	bab
abb	ab
$abba$	b
$abbab$	λ

$$w = uv$$

u: prefix

v: suffix

Another Operation

$$w^n = \underbrace{ww \cdots w}_{n}$$

Example: $(abba)^2 = abbaabba$

Definition: $w^0 = \lambda$

$$(abba)^0 = \lambda$$

The * Operation

Σ^* : the set of all possible strings from alphabet Σ

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

$$\Sigma^* = \{\lambda, a, b, aa, ab, ba, bb, aaa, aab, \ldots\}$$

The + Operation

Σ^+ : the set of all possible strings from alphabet Σ except λ

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

$$\Sigma^* = \{\lambda, a, b, aa, ab, ba, bb, aaa, aab, \ldots\}$$

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

$$\Sigma^+ = \{a, b, aa, ab, ba, bb, aaa, aab, \ldots\}$$

Languages

A language over alphabet Σ is any subset of Σ^*

Examples:

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

$$\Sigma^* = \{\lambda, a, b, aa, ab, ba, bb, aaa, \ldots\}$$

Language: $\{\lambda\}$

Language: $\{a, aa, aab\}$

Language: $\{\lambda, abba, baba, aa, ab, aaaaaa\}$

More Language Examples

Alphabet $\Sigma = \{a, b\}$

An infinite language $L = \{a^n b^n : n \geq 0\}$

$$\left.\begin{array}{l} \lambda \\ ab \\ aabb \\ aaaaabbbb \end{array}\right\} \in L$$

$$abb \notin L$$

Prime numbers

Alphabet $\Sigma = \{0,1,2,\ldots,9\}$

Language:

$$PRIMES = \{x : x \in \Sigma^* \text{ and } x \text{ is prime}\}$$

$$PRIMES = \{2,3,5,7,11,13,17,\ldots\}$$

Even and odd numbers

Alphabet $\Sigma = \{0,1,2,\ldots,9\}$

$$EVEN = \{x : x \in \Sigma^* \text{ and } x \text{ is even}\}$$

$$EVEN = \{0,2,4,6,\ldots\}$$

$$ODD = \{x : x \in \Sigma^* \text{ and } x \text{ is odd}\}$$

$$ODD = \{1,3,5,7,\ldots\}$$

Note that:

Sets $\emptyset = \{\} \neq \{\lambda\}$

Set size $|\{\}| = |\emptyset| = 0$

Set size $|\{\lambda\}| = 1$

String length $|\lambda| = 0$

Operations on Languages

The usual set operations

$$\{a, ab, aaaa\} \cup \{bb, ab\} = \{a, ab, bb, aaaa\}$$

$$\{a, ab, aaaa\} \cap \{bb, ab\} = \{ab\}$$

$$\{a, ab, aaaa\} - \{bb, ab\} = \{a, aaaa\}$$

Complement:

$$\overline{L} = \Sigma^* - L$$

$$\overline{\{a, ba\}} = \{\lambda, b, aa, ab, bb, aaa, \ldots\}$$

Reverse

Definition: $$L^R = \{w^R : w \in L\}$$

Examples: $$\{ab, aab, baba\}^R = \{ba, baa, abab\}$$

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

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

Concatenation

Definition: $L_1 L_2 = \{ xy : x \in L_1, y \in L_2 \}$

Example:

$$\{a, ab, ba\}\{b, aa\}$$

$$= \{ab, aaa, abb, abaa, bab, baaa\}$$

Another Operation

Definition: $$L^n = \underbrace{LL\cdots L}_{n}$$

$$\{a,b\}^3 = \{a,b\}\{a,b\}\{a,b\} =$$
$$\{aaa, aab, aba, abb, baa, bab, bba, bbb\}$$

Special case:

$$L^0 = \{\lambda\}$$

$$\{a, bba, aaa\}^0 = \{\lambda\}$$

Star-Closure (Kleene *)

L

$$L^* = L^0 \cup L^1 \cup L^2 \cdots$$

$$\{a, bb\}^* = \left\{ \begin{array}{l} \lambda, \\ a, bb, \\ aa, abb, bba, bbbb, \\ aaa, aabb, abba, abbbb, \ldots \end{array} \right\}$$

Positive Closure

Definition: $L^{+} = L^{1} \cup L^{2} \cup \cdots$

Same with L^{*} but without the λ

$$\{a, bb\}^{+} = \left\{ \begin{array}{l} a, bb, \\ aa, abb, bba, bbbb, \\ aaa, aabb, abba, abbbb, \ldots \end{array} \right\}$$

Questions ??
?