**CSC301**

**Lecture 5 summary**

1910456

---

---

**What is the aim of this course?**

---

We are trying to understand what a language is. In computers and programming languages, what goes on. What the building blocks and fundamentals are.

We will be using them in the compiler courses in detail.

We have discussed what the language of a DFA or NFA means: The strings that are accepted by the DFA. Conversely, if we have a language, we can make a DFA from it.

A DFA/NFA therefore, are associated with a language.

What those associated languages are is the topic of this lecture.

---

String Concatenation

---

For strings "11100" + "1001" = "111001001"

$L_1 L_2 = \{st : s \in L_1,\ \ t \in L_2\}$ Means a set of where the first part of each string comes from $L_1$ and the second part comes from $L_2$.

If $L_1$ = {110, 01} and $L_2$ = {aa, bbb} then $L_3 = L_1L_2$ = {110aa, 110bbb, 01aa, 01bbb}.

$L_3$ is constructed under concatenation operation.

---

Nth power of L, $L^n$

---

Basically, concatenation with itself.

If $L_1$ = {110, 01} $L_1^2$ = {110110, 11001, 01110, 0101}

---

Union

---

$L_1 \cup L_2 = \{\ s : s \in L_1,\ \ s \in L_2\}$.

If $L_1$ = {110, 01} and $L_2$ = {aa, bbb}, then $L_3 = L_1 \cup L_2$ = {110, 01, aa, bbb}

$L_3$ is constructed under Union operation.

---

Reversal

---

Reversal of string w is $w^r$.

$L^r = \{s_1 s_2 \dots s_n : s_n s_{n-1} \dots s_1 \in L\}$

If L = {110, 01, aa, bbb}, then $L^r$ = {011, 10, aa, bbb}.

----------------------------------

Complement meaning discussed, same as set definition.

----------------------------------

Star operation

----------------------------------

The star of a language are all the strings that are made up of zero or more chunks from the language.

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

If $L_1$ = {0, 01}

${L_1}^0 = \{\varepsilon\}$

${L_1}^1$ = {0, 01}

${L_1}^0$ = {00. 001. 010. 0101}

And so on.

Here, ${L_1}^*$ is the set of all strings that start with 0 and do not contain consecutive 1s plus the empty string.

----------------------------------

Combining languages

----------------------------------

{0}( {0}∪{1} )* → 0(0+1)* [“+” represents Union/ can be thought of as OR ]

= {0}({0, 1})* = {0}{$\varepsilon$, 0, 1, 00, 01, 10,......} = {0, 00, 01, 000, 001, 010, .....} which means any string that starts with 0.

We can also make a DFA/NFA from this.

----------------------------------

({0}{1}*)∪({1}{0}*) → 01* + 10*

We think of it in two parts:

01* is the set of all strings that start with 0 and has any number of consecutive 1s (including $\varepsilon$).

10* is the set of all strings that start with 1 and has any number of consecutive 0s (including $\varepsilon$).

Therefore 01* + 10* = {0, 1, 01, 10, 011, 100,.....}

----------------------------------

**Regular Expressions**

----------------------------------

A regular expression over Σ is an expression formed using:

- $\varnothing$ *and* $\varepsilon$ are regular expressions

$L_1 = \varnothing$ and $L_2 = \{\varepsilon\}$ are separate languages.

$L_1$ can be represented as a DFA which has a single state, the start state, that is not a final state. (it doesn't accept anything)

$L_2$ can be represented as a DFA which has a single state, the start state, that itself is a final state. (It accepts $\varepsilon$)

- Every a in $\Sigma$ is a regular expression. It means every symbol over an alphabet themselves are regular expressions. (0 and 1 are regular expressions)
- If R and S are regular expressions, so are R+S (union), RS (concatenation) and R* (star).

If R = 1 + 0 and S = 01, then SR = 01(1+0) = P. P* = [01(1+0)]*.

All of them are languages.

The languages that we can make from these operations are called regular languages which can be expressed through regular expressions (which can be expressed through automata).

---

(0+1)*01(0+1)* = {01, 010101, 10101111,....}

Any string that contains 01 as substring.

---

(01*)(01)

Any string that starts with 0 and has any number of consecutive 1s and also ends with 01.

---

0+1

Strings of length 1

---

(0+1)*

Any string including $\varepsilon$.

---

(0+1)*010

Any string that ends with 010

---

((0+1)(0+1))* + ((0+1)(0+1)(0+1))*

Breaking it down,

((0+1)(0+1))* = ({00, 01, 10, 11})*

Which means any strings of even length.

((0+1)(0+1)(0+1))* = ({000, 001, 010, 011, ...})*

Any strings of length which is a multiple of 3