

**19CSE214- THEORY OF COMPUTATION**

**TITLE: CONVERSION OF REGULAR EXPRESSION TO NON-DETERMINISTIC FINITE AUTOMATA (NFA)**

**GROUP:-1**

Github URL**:** <https://github.com/SanyamB0912/Expr-to-NFA>

**Team Members**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S.No | Name | Roll Number | Contribution | Photo |
| 1 | Manasa R | CB.EN.U4CSE22029 | Module 3 |  |
| 2 | Rohan Prkash N | CB.EN.U4CSE22039 | Graphics | C:\Users\sbhar_\AppData\Local\Packages\5319275A.WhatsAppDesktop_cv1g1gvanyjgm\TempState\BD7EAD7664EC93DD08719F2D91D354F8\WhatsApp Image 2024-04-04 at 12.56.15_12edb21d.jpg |
| 3 | Sanyam Bhardwaj | CB.EN.U4CSE22044 | Module 1 |  |
| 4 | Sriya Simran Dash | CB.EN.U4CSE22045 | Module 2 | C:\Users\sbhar_\Downloads\simran.jpg |
| 5 | Katari Amrutha Varshini | CB.EN.U4CSE22065 | Module 4 | C:\Users\sbhar_\Downloads\Amrutha.jpg |

**About the Modules**

1. Module 1:

* Focuses on receiving input from the user and defining a 2- dimensional array for it
* Each index of the 2-D array represents a table.
* The corresponding list for each state highlights the transitions to a particular state given inputs a, b and e.

1. Module 2:

* This code segment handles parsing of regular expressions with alternations, like 'a|b' or 'b|a'.
* It constructs a Non-deterministic Finite Automaton (NFA) to represent the regular expression.
* When an alternation is encountered, epsilon transitions are set up.
* Epsilon transitions don't consume input and indicate choices.
* For example, in 'a|b', transitions are set for 'a' and 'b' options.
* The current state 'j' tracks NFA states during construction.
* After setting transitions, it moves to the next part of the expression.
* This process iterates until the entire expression is parsed.
* Both the transition table 'q' and the NFA graph are updated.
* The code efficiently handles alternations in regular expressions.

1. Module 3:

* This part of the code handles the Kleene star (\*), which allows the preceding character (a or b in this case) to appear zero or more times in the regular expression.
* If the current character is 'a' or 'b', and the next character is '\*', it signifies that there can be zero or more occurrences of 'a' or 'b'.
* The transition table `q` is updated to reflect epsilon transitions (`e`) from the current state to the next state and the state after that.
* An epsilon transition is added from the current state to the next state.
* Then, an epsilon transition is added from the current state to the state before the current state (previous state).
* This effectively creates a loop in the NFA, allowing for zero or more occurrences of 'a' or 'b' in the input string.
* If the current character is ')' and the next character is '\*', it signifies that the preceding expression within the parentheses can repeat zero or more times.

1. Module 4:

(Continuation of Module 3)

* The transition table `q` is updated to reflect epsilon transitions (`e`) from the starting state to the next state and from the current state to the starting state and the state after that.
* An epsilon transition is added from the starting state to the next state.
* Then, epsilon transitions are added from the current state to the next state and to the starting state.
* This ensures that after the expression within the parentheses finishes its rep
* This Python script generates a transition table for a Non-deterministic Finite Automaton (NFA) represented by a list of states 'q'. It iterates through the states to determine transitions based on inputs 'a', 'b', and 'e', constructing a formatted transition table. The script then identifies the starting and final states and saves an NFA diagram to a file for visualization.