TOC Tutorial Questions

1. Construct a deterministic finite automata that accept the set of all strings in {a, b, c}\* such that the last symbol in input strings appears earlier in the string
2. Draw DFA for a language of all strings of odd number of 0’s and 1’s over the alphabet

= {0, 1}.

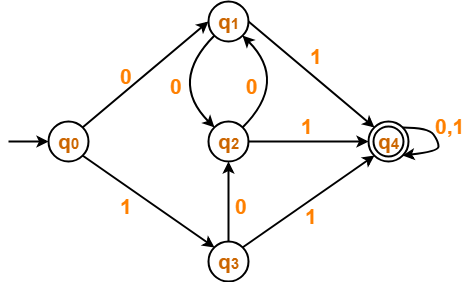
1. Construct a DFA that accepts a language L over input alphabets ∑ = {a, b} such that L is the set of all strings starting with ‘aa’ or ‘bb’.
2. Draw a DFA for the language accepting strings ending with ‘abba’ over input alphabets ∑ = {a, b}
3. Consider the following epsilon NFA. Compute the epsilon Closure of each state and find its equivalent DFA.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | a | b | c |
| ->p | {q,r} | Ф | {q} | {r} |
| q | Ф | {p} | {r} | {p,q} |
| \*r | Ф | Ф | Ф | {r} |

1. Construct a regular expression to the transition diagram.

|  |  |  |
| --- | --- | --- |
|  | 0 | 1 |
| ->q1 | {q1} | {q2} |
| q2 | {q3} | {q2} |
| \*q3 | {q1} | {q2} |

1. Write regular expressions for the following language over the alphabet = {0, 1}. “The set of all strings not containing 101 as a substring”.
2. Write a r.e to denote a language L which accepts all the strings which begin or end with either 00 or 11.
3. Construct a NFA for the regular expression (a+b)(a+b)\*
4. Minimize the DFA given below



1. Prove that L = {ww | w ∈ {0, 1} ∗} is not regular.
2. Prove that L = { 0n| n is a power of 2 } is not regular
3. Prove that L = {ww R | w ∈ {0, 1} ∗} is not regular.
4. Let S-> aB/bA , A->aS/bAA/a, B-> bS/aBB/B. Derive the string aaabbabba as left most derivation.Draw the parse tree
5. Find out the language of the Context Free grammar

S-> aSb | aAb , A-> bAa, A-> ba

1. Construct Context Free Grammars to accept the following languages over {0, 1}. The language {w|w starts and ends with the same symbol}.
2. Construct the grammar for the language L= { an b an | n>=1}.
3. Construct the grammar for the language { w ∈ {0, 1} ∗ | the length of w is odd and the middle symbol is 0 }
4. Convert the grammar into CNF S → 0A0 | 1B1 | BB, A → C, B → S | A, C → S | Ɛ
5. Convert the grammar into CNF S → SS | (S) | ()
6. Convert the following grammar to the Greibach Normal Form.

S -> a | CD | CS

A -> a | b | SS

C -> a

D -> AS

1. Convert the expression grammar to the Greibach Normal Form.

E → E + T / E – T / T

T → T x F / T ÷ F / F

F → id