Proceedings of International Conference on Advances in Computer Technology and Management (ICACTM)
In Association with Novateur Publications IJRPET-ISSN No: 2454-7875
ISBN No. 978-81-921768-9- 5
February, 23rd and 24th, 2018

SIMPLIFIED SOLUTION TO DESIGN DFA, THAT ACCEPT STRINGS OVER {A,B} HAVING AT LEAST X NUMBER OF 'A' OR Y NUMBER OF 'B' COMMON BETWEEN TWO CONDITION.

PROF.NILIMA SHINGATE
Department of Computer Science
Head of Department
Christ College, Pune
Nilima.gaikwad86@gmail.com
9766794701

MR.PRADIP SHINGATE
Senior Software Developer
WHIZ Technology, Pune
pradip.shingate@gmail.com
9766934468

ABSTRACT
Today, it is very difficult to understand the de- signing concepts of deterministic machine. One question is arise? How it is possible to understand the concepts of deterministic machine in a very easy manner. In this paper, we have design a DFA and develop an method with suit- able examples that how DFA machine works in a simply manner. For it, we consider that a DFA machine takes the input string {a, b} having at least x number of a or y number of b common between TWO condition. The objective of this paper to under- stand the concepts of deterministic machine in easy manner.

KEYWORD - DFA, Transition Table, Transition Table (TT)

I. INTRODUCTION

DFAs are a fundamental topic in computer science education. Besides being part of the standardized computer science curriculum, the concept of DFA is rich in structure and potential applications. It is useful in diverse settings such as control theory, text editors, lexical analyzers, and models of software interfaces.
A deterministic Finite automaton (also known as determin- istic finite state machines) is the system to accomplish many tasks in Computer Science. To increase the computational power of existing computers, it is based not only to increase the frequency of CPU but also we use other modern technologies. The finite automata implementations are used to consider these types of technologies. For example, multiple CPU core is one of the latest technologies which is used now. We can represent DFA by digraphs which is also called state transition diagram. In this digraph the vertices are denoted by single circles of a transition diagram which represent the states of the DFA and the arcs are labeled with an input symbol correspond to the transitions. We represent accepting states by double circles.
Efficient learning of DFA is a challenging research problem in grammatical inference. It is known that both exact and approximate (in the PAC sense) identifiability of DFA is hard.

II. MYTHOLOGY

Automata theory is a branch of theoretical computer sci- ence, DFA is known as Deterministic Finite Automata. A finite state machine accepts or rejects finite strings of sym- bols and gives a unique computation for each input string[15]. McCulloch and Pitts were among the first re-searchers to introduce a concept similar to finite automaton in 1943.[5]
A DFA is defined as an abstract mathematical concept, but due to the deterministic nature of a DFA, it is implementa- ble in hardware and software for solving various specific problems[15]. For example, a DFA can model software that decides whether or not online user-input such as email addresses are valid.

Finite Automata (M) is defined as a set of five tuples
(Q, Σ, δ, q , F)
0
Where

Q= a finite, non-empty set of states
Σ= a finite, non-empty set of inputs δ
is the state-transition function: δ:Q X
Σ → Q
q is the initial state
0

F is the set of final states, a subset of Q.

δ can be represents using either of three approach given below

•Transition Graph.

• Transition Function.

• Transition Table.

We had used the transition table as the approach to represent δ.

III. ALGORITHM

By applying this Algorithm we can design Deterministic Finite Automata that accept strings over input symbol a, b having atmost x number of a & y number of b
Algorithm to draw TG
Deterministic Finite Automata M= (Q, Σ, δ, q_0, F)
Where
Danish Ather, et al., JNIC, Vol. 1, No. 2, pp. 30-33, 2013 31
Q= {q11,q12,q13,….q21,q22,……qij}
Σ= {a,b}
δ:QXΣ→Q{Represented by Transition Graph }
Q0 = qij when i=j and i=j=1. i.e. q11
F={q11,q12,q13,….q21,q22,……qij}

Let Q be the set of states in Deterministic Finite Automata such that Q={q11,q12,q13,....q21,q22,.....qij}
Where i = 1 to x+1
j= 1 to y+1
Input Symbol Σ={a,b}
q11 is the initial state.
Design a directed transition graph having (x+1)*(y+1) states and mark all states as final states.
Label each node as q11,q12,q13,....q21,q22,.....qij
Where i = 1 to x+1; j= 1 to y+1; x= na & y=nb
FOR i= 1 to x
do
FOR j= 1 to y
do
if i=j=1 then qij ε Q0 (Initial State)
else there exist a edge E such that δ(qij,a)→qij+1
done inner loop
done outer loop
FOR i= 1 to x
do
FOR j= 1 to y
do
if i=j=1 then qij ε Q0 (Initial State)
else there exist a edge E such that δ(qji,b)→qj+1,i
if i=x+1 and j=y+1 then there exist δ(qij, a)→qij and δ(qij, a)→qij.
done inner loop
done outer loop
q11 being the initial state
DFA "M" will strings over input symbol a, b having atmost x number of a & y number of b if all the input is consumed and halting state is the final state.

IV.IMPLEMENTATION

Example 1:

Design a DFA,over a language {a,b,c} which start with 'ab' and end with 'bc'.

Let the resultant DFA is M = (Q, Σ, δ, q_0, F)
Where

Q = {q_0, q_1, q_2, q_3, q_4}
Σ = {a, b, c}
δ = Q X Σ → Q
q_0 = { q_0}
F = {q_3}

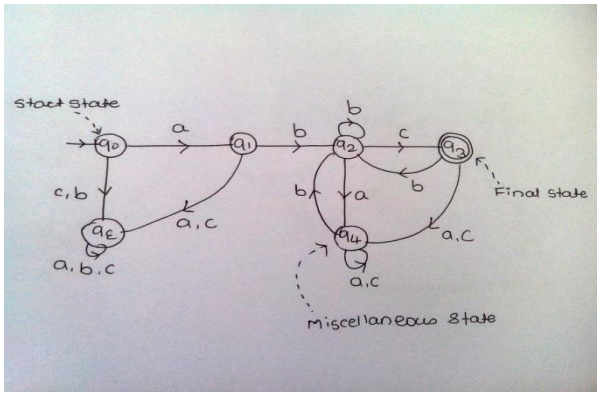

Figure 1: DFA,over a language {a,b,c} which start with 'ab' and end with 'bc'

TRANSITION TABLE:

M	a	b	c
q_0	q_1	q_E	q_E
q_1	q_E	q_2	q_E
q_2	q_4	q_2	q_3
q_3	q_4	q_2	q_4
q_4	q_4	q_2	q_4
q_E	q_E	q_E	q_E

Example 2:

Design a DFA,over a language {p,q,r} which start with 'pq' having sub string 'qpr' and end with 'rq'.

Let the resultant DFA is M = (Q, Σ, δ, q_0, F)
Where

Q = {q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7}
Σ = {p, q, r}
δ = Q X Σ → Q
q_0 = { q_0}
F = {q_5}

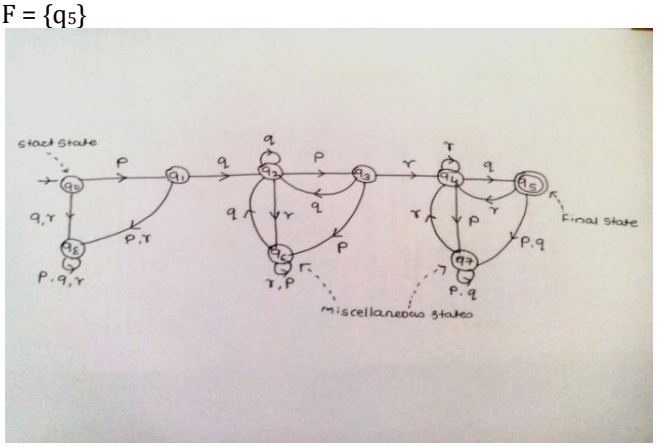

Figure 2: DFA,over a language {p,q,r} which start with 'pq' having sub string 'qpr' and end with 'rq'.

TRANSITION TABLE

M	a	b	c
q_0	q_1	q_E	q_E
q_1	q_E	q_2	q_E
q_2	q_3	q_2	q_6
q_3	q_6	q_2	q_4
q_4	q_7	q_5	q_4
q_5	q_7	q_7	q_4
q_6	q_6	q_2	q_6
q_7	q_7	q_7	q_4
q_E	q_E	q_E	q_E

V.RESULT ANALYSIS AND DISCUSSION

In the FIRST example - DFA,over a language {a,b,c} which start with 'ab' and end with 'bc'.

Input symbol 'b' is common so, smallest accepted string over {a,b,c} is **abc** .

RULE or CLUE:

Total number of states in DFA = total number of condition input symbol + 1(our side)

So, In this example **we should have 5 states.**

But when we draw the basic DFA, it contain only 4 states,
Because 'b' is common.

After solving we come to know that **1 state is miscellaneous.**
So, we have to keep in mind that whenever we are solving such problem, **one extra or miscellaneous state will come.**

In the SCECOND example - DFA,over a language {p,q,r} which start with 'pq' having sub string 'qpr' and end with 'rq'.

Input symbol 'q' and 'r' are common so, smallest accepted string over {p,q,r} is **pqprq**.

According to above **RULE or CLUE:**

we should have 8 states.
But when we draw the basic DFA, it has only 6 states,
Because 'q' and 'r' are common.

After solving we come to know that **2 state is miscellaneous.**
So, we have to keep in mind that whenever we are solving such problem, **two extra or miscellaneous state will come.**

VI. CONCLUSION

This research will definitely enhance the teaching learning environment of theory of computation and helps students to design DFA.

VII.REFERENCES

1) Hopcroft, John E.; Motwani, Rajeev; Ullman, Jeffrey D. (2001). Introduction to Automata Theory, Languages, andComputation (2 ed.). Addison Wesley. ISBN 0-201-44124-1.Retrieved 19 November 2012.
2) Lawson, Mark V. (2004). Finite automata. Chapman and Hall/CRC. ISBN 1-58488-255-7. Zbl 1086.68074.
3) McCulloch, W. S.; Pitts, E. (1943). "A logical calculus of the ideas imminent in nervous activity". Bulletin of Mathematical Biophysics: 541–544.
4) Rabin, M. O.; Scott, D. (1959). "Finite automata and their decision problems.". IBM J. Res. Develop.: 114–125.
5) Sakarovitch, Jacques (2009). Elements of automata theory. Translated from the French by Reuben Thomas. Cambridge: Cambridge University Press. ISBN 978-0-521-84425-3. Zbl 1188.68177.
6) Sipser, Michael (1997). Introduction to the Theory of Computation. Boston: PWS. ISBN 0-534-94728-X.. Section 1.1: Finite Automata, pp. 31–47. Subsection "Decidable Problems Concerning Regular Languages" of section 4.1: Decidable Languages, pp. 152–155.4.4 DFA can accept only regular language
7) C. Allauzen and M. Mohri, Finitely subsequential transducers, International J. Foundations Comp. Sci. 14 (2003), 983-994
8) M.-P. Béal and O. Carton, Determinization of transducers over finite and infinite words, Theoret. Comput. Sci. 289 (2002), 225-251
9) Bruggemann-Klein, Regular expressions into finite automata, Lecture Notes in Computer Science 583 (1992), 87-98
10) J. Carroll and D. Long, Theory of Finite Automata, Prentice- Hall, Englewood Cliffs, NJ, 1989.
11) M. V. Lawson, Finite Automata, CRC Press, Boca Raton, FL,2003.
12) D. Perrin, Finite automata, in Handbook of Theoretical Computer Science, Volume B (editor J. Van Leeuwen), Elsevier, Amsterdam, 1990, 1-57.
13) D. Perrin and J. E. Pin, Infinite Words, Elsevier, Amsterdam, 2004.
14) Ch. Reutenauer, Subsequential functions: characterizations, minimization, examples, Lecture Notes in Computer Science 464 (1990), 62-79.
15) J. Sakarovitch, Kleene's theorem revisited, Lecture Notes in Computer Science 281(1987), 39-50.
16) E. Roche and Y. Schabes (editors), Finite-State Language Processing, The MIT Press, 1997.