JOURNAL OF INFORMATION, KNOWLEDGE AND RESEARCH IN COMPUTER ENGINEERING
ISSN: 0975 – 6760 | NOV 16 TO OCT 17 | Volume 4, Issue 2

New method for generating helpful state of deterministic finite automata

Ms. Khushbu Doulani [1] Ms. Gitanjali Bhimrao Yadav[2] Mr.Sunil M.Jadhav [3] Ms. Poonam Suresh Nagale[4]

[1, 2, 3, 4] Rajgad Dnyanpeeth's technical Campus,
Shri Chhatrapati Shivajiraje College of Engineering
Dhangwadi,Pune.

[1]khushidoulani@gmail.com, [2]gitanjali3014@gmail.com, [3]suniljadhav02@gmail.com, [4]poohadke@gmail.com

Abstract-- *In deterministic finite automata, some state is not useful that means this type of state doesn't participant for generating useful string. This type of state is called dead state, inaccessible state or indistinguishable state. In deterministic finite automata, if we want to determine inaccessible state or (we can say unreachable state) and dead state then this is not easy for own if closure input is available. It is necessary for removing unreachable state and dead state from deterministic finite automata. Without removing unreachable state and dead state deterministic finite automata is not well form. Generally this paper is suitable for removing unreachable state in deterministic finite automata. If we will follow proposed technique then we can easily remove unreachable state in deterministic finite automata. Mainly, if we will apply this technique then we can easily remove unuseful state in deterministic finite automata. That means we can generate easily useful state in deterministic finite automata. After generating useful state we can minimize easily. So, it is important part of deterministic finite automata of generate useful state. In new deterministic finite automata (after generating of useful state), unreachable state will not available. So, we can say generation of useful state is a technique for removing unreachable state from deterministic finite automata. The presented paper, generates useful state by new technique or new approaches with taking less input symbol than running technique. Also in this paper proposed algorithm developed for removing unreachable state in deterministic finite automata. And, in this paper compares different running approaches with input symbol. Also, in this paper discussing about how java formal languages and automata package simulator useful for new (presented) technique.*

Keywords: Automata; Deterministic finite automata; Unreachable state; Dead state.

I. INTRODUCTION

A. What is automata

It is define as a system where energy, materials, information are transformed, transmitted and used for performing some function without direct participation of human. In second way we can define automata is a machine for generating regular expression, context free grammar, context sensitive grammar and recursive endurable language. In computer science, automaton means 'discrete automaton' [1,2].

A finite automaton (FA) M as the quintuple $M = (Q, \Sigma, \delta, q_s, F)$ where

Q is a finite set of states $\{q_i \mid i \text{ is a nonnegative integer}\}$

Σ is the finite input alphabet

δ is the transition function, $\delta : D \rightarrow 2^Q$ where D is a finite subset of $Q \times \Sigma^*$

q_s (is member of Q) is the initial state

F (is a subset of Q) is the set of final states

Note that, above definition includes both deterministic finite automata (DFAs), which we will be discussing shortly, and nondeterministic finite automata (NFAs), which we will discuss on later[4,7].

B. Definitions and Notations

1) **Alphabet and Language:** It (Alphabet) is defined as finite non-empty set of symbols on which the language is defined. Alphabets are denoted by Σ. Language is defined as a subset of Σ^*. Empty string and null language are denoted by ϵ and ϕ respectively. Various kinds of formal languages can be classified as regular language, context free language, context sensitive language and recursive language. Regular language can be described by regular expression, finite automata

2) (Deterministic or Non-deterministic). A language over 'a' and 'b' that will include all strings having length less than 2 is L={ ϵ,a,b,aa,ab,ba,bb}[3,8].

3) **Operations on languages:** Following are the operations that can be performed on languages.

Union: Union of two languages L1 and L2 is set of all the strings that are in also L1 or L2, or both. For example, if L1={001,10,111} and L2={ϵ,01,10}, then L1UL2={ϵ,01,10,001,10,111}.

Concatenation: Concatenation of two languages L1 and L2 is set of all the strings that can be formed by taking one string in L1 and concatenating it with any string in L2. Regular Language can be articulated by regular expression or DFA. i.e.

if L1={101,10,111} and L2={ϵ,01}, then L1L2={101,10,111,10101,1001,11101}[5,6].

Kleene Closure: Kleene closure of a language L represent the set of folks strings that can be formed by taking any number of strings from L, possibly with repetitions (same string can be selected more than once) and concatenating all of them. Kleene closure of a language L is denoted by L*. For example if L={0,1}, then kleene closure of L is all strings of 0"s and 1"s i.e. L* ={ϵ,0,1,01,001,0110,111010,110011.....}[9].

C. Deterministic Finite Automata

Deterministic finite automaton (DFA) is a finite state machine (system) accepting finite strings of symbols. For each state, there is a transition arrow primary out to a next state for each symbol. Deterministic finite automata (DFA) can be defined by 5-tuples (Q, Σ, δ, q0, F), where

Q is a finite set of states

Σ is a finite set of symbols

δ is the transition function, that is, δ: Q × Σ → Q.

q0 is the start state

F is a set of states of Q (i.e. F⊆Q) called accept states [10,11].

II. PROBLEM STATEMENT

The problem is finding the useful state in deterministic finite automata. Different technique or approaches are available for generating useful state in deterministic finite automata. Some approaches are available for removing useless state in deterministic finite automata. First of all we know about what is the unreachable state and dead state.

Unreachable state: In deterministic finite automata some state is call of unreachable state if no path present from initial state to unreachable state.

Dead state: in deterministic finite automata some state is call of dead state if a state that is not an accepting state and no any transition from another state than itself.

As shown in below figure:

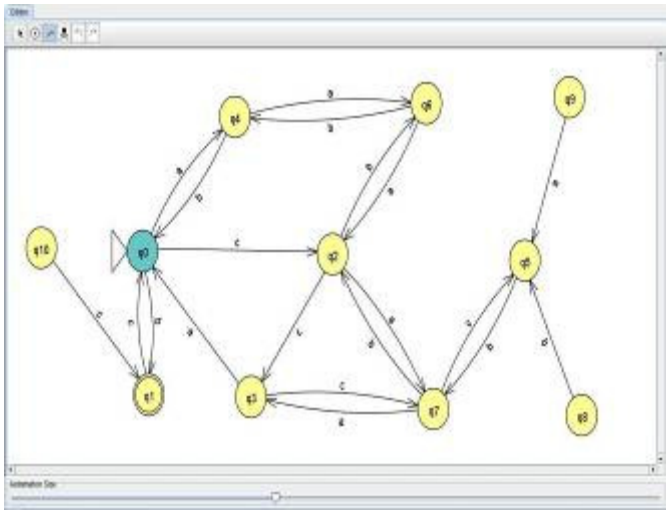

Fig 2.1: Deterministic finite automata with unreachable state.

State q8, q9 and q10 are unreachable state in above figure because if we will take any string from initial state (q0) to q8 or q9 or q10 then it is not possible. Some approaches for finding this state.

DFS(depth first search) technique: In this approach first of all take outgoing input symbol from initial state and if outgoing input symbol is more than one then it will follow depth first search technique that means, state q0 move q1,q2 and q4 so three path present from q0. But it will take q1 if it is follow DFS(depth first search). After completing total path from DFS(depth first search) root then we will move q2 way as well as q4. So problem of this technique or approach is taking time if loop available in DFS root.

Any path choosing approach: In this approach any one path select and move last possible state. In this approach if any state remaining for not participating in input symbol the this state is an unreachable state. Problem of this approach is it will taking more and more time for removing unreachable state.

III. PROPOSED TECHNIQUE

In this technique, first of all generate useful state before minimization. If we generate useful state then useless state (unreachable state) will remove simultaneously.

In proposed approach first of all take only one input symbol from initial state and move simultaneously with changing accepting input symbol state as shown in below figure.

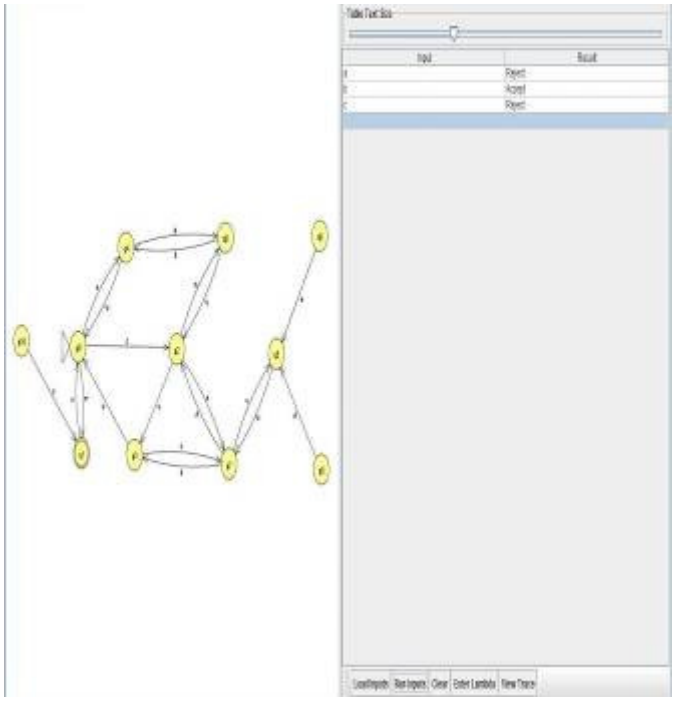

Fig. 3.1: Deterministic finite automata with 'a' ,'b' and 'c' input symbol.

In this figure input symbol is 'a', 'b' and 'c' because outgoing symbol of state q0 is 'a', 'b' and 'c'. Input symbol 'b' is

accepting but input symbol ‘a’and ‘c’ is rejecting so we can see View Trace by JFLAP simulator and finalized all state including accepting input symbol. After that take another input string, taking input string as per proposed rule that means all input take simultaneously with string length is one that means only one symbol include at a time. As shown in below figure.

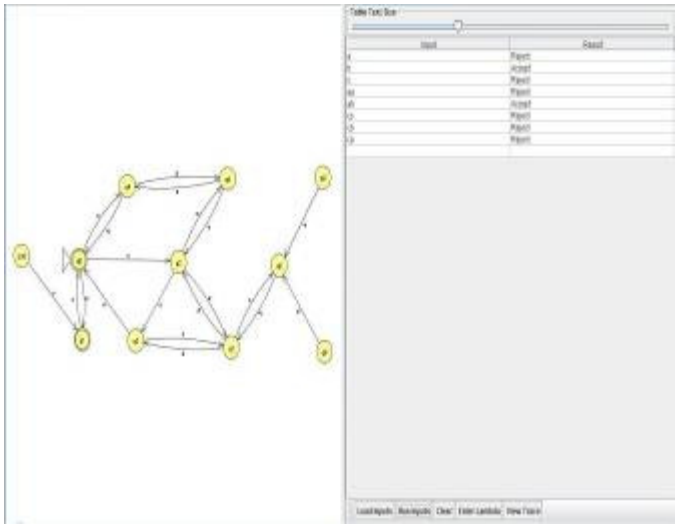

Fig. 3.2: Deterministic finite automata with ‘one length’ and ’two length input symbol.

In above figure we finalize state q0 because this state generating accepting input symbol with final state and take next input symbol as shown in above figure. In this figure if we are taking input symbol ‘aa’,’cc’,’ca’ and ‘cb’ are rejecting so we will take next input symbol. Remembering that all the input symbol starts with initial state. Also ‘ab’ is accepting string so state state q4 is also a final state. After that taking next input string as shown in below figure.

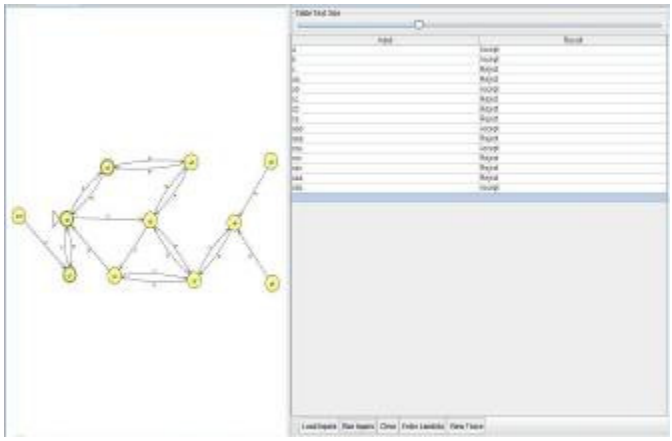

Fig. 3.3: Deterministic finite automata with ‘one, two and three length’ input symbol.

As shown in above figure on some input symbol is accepting and some input symbol is rejecting then we will see view trace of JFLAP simulation and finalised all the state including in accepting input symbol . As shown in below figure.

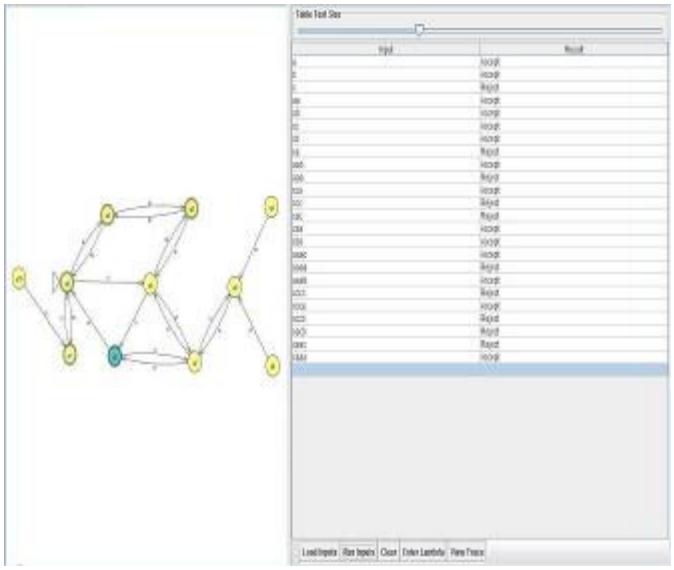

Fig. 3.4: Deterministic finite automata with more input symbol input symbol.

In above figure some state is finalising because this state including accepting input symbol again we will take next input symbol. If no any input symbol is accepting then we will take next possible input symbol. Again if no input symbol

accepting then repeat it whenever all possible input not finished.

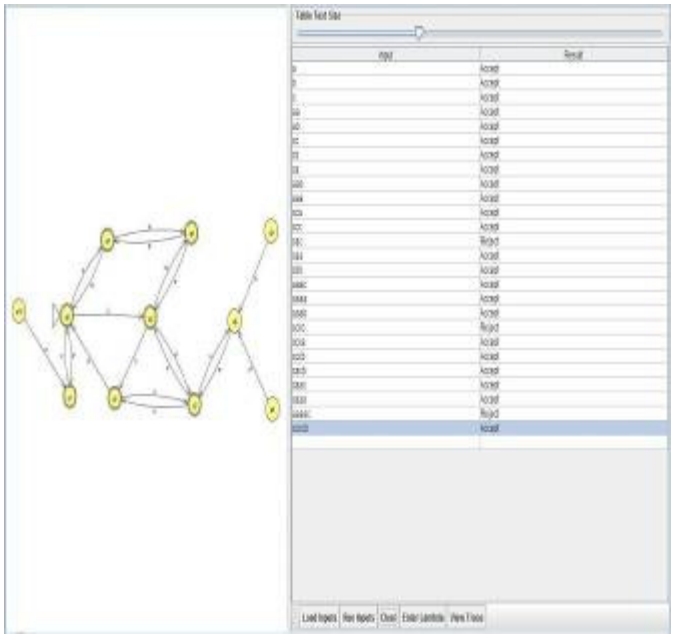

Fig. 3.5: Deterministic finite automata with maximum possible input symbol.

In above figure maximum string is accepting but some string is not accepting then again apply proposed technique that means state including accepted string is finalize and then go to taking next string.as sown in below figure.

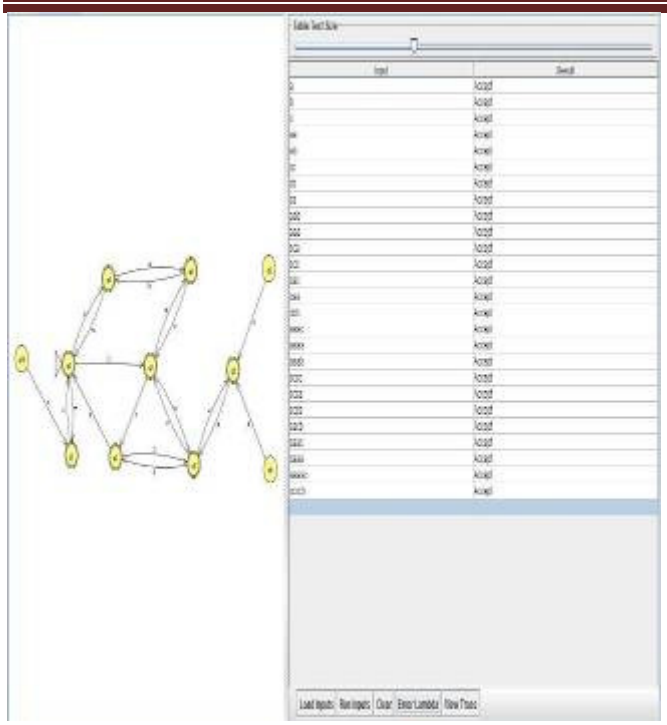

Fig. 3.6: Deterministic finite automata with all possible input symbol.

In above figure we can see all taking string is accepted that means no any useful state remaining for generating technique of useful state in deterministic finite automata. When all possible inputs are taken so we will remove all non-final state as shown in below figure. In above figure final state indicate accepting string generated by using these state and non- final state indicated no any string generated by using these state.

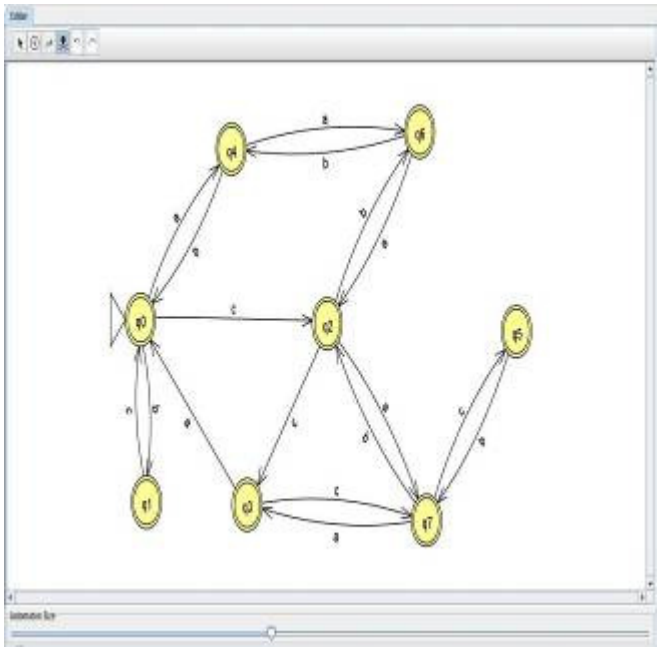

Fig. 3.7: Deterministic finite automata with without unreachable state.

In this figure all final state available and no any non- final state available. Final state shows these state are including in accepting string. Now nest step for proposing technique, remove all finalized label of state except initialized form means in initial form of deterministic finite automata initial state was q0 and final state was q1 so these state are not same as a previous form and all updated label should be remove. As shown in below figure.

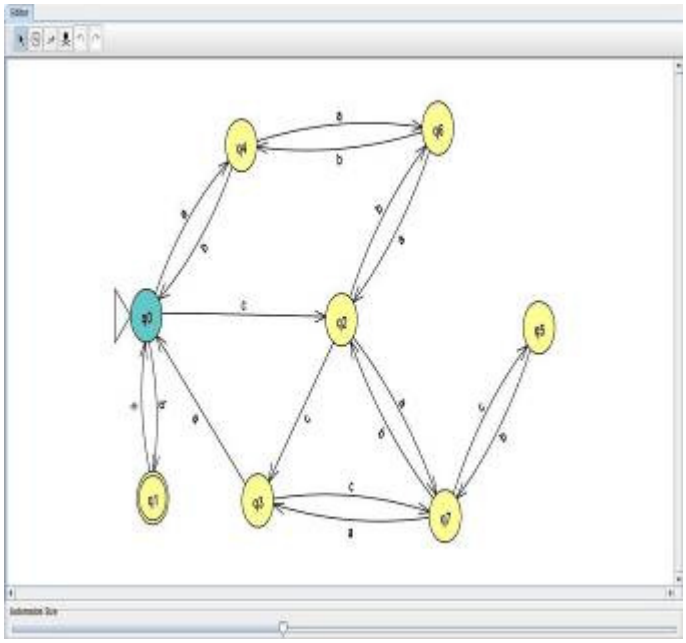

Fig. 3.8: Deterministic finite automata with useful state.

In above figure all useful state available means only whose state is available those are including in accepting string. After that our proposed technique finised.

IV. CONCLUSION:

Choosing the useful state of deterministic finite state automaton is one of the challenging concepts for students at an introductory level to understand and learn. In this paper mainly removing of unreachable and dead state of deterministic finite automata. We can follow simple approach for generating useful state by given approach in this thesis. We can choose JFLAP simulation for generating useful state of deterministic finite automata. If given technique apply for generating useful state then both unreachable state and dead state is simply remove. Also if we will follow given technique or approach, number of step for taking input is lesser than running technique. After selecting useful state we can minimise simply of deterministic finite automata.

V. FUTURE WORK

In this paper only unreachable state and dead state of deterministic state is removing. But in this paper indistinguishable state is not removing of deterministic finite automata. So for future work we can work on this part that

means for removing indistinguishable state of deterministic finite automata. Also we can work on minimization of deterministic finite automata in future time. We can work on state label in future work.

REFERENCES

[1] Alfred V. Aho, "Constructing a Regular Expression from a DFA", Lecture notes in Computer Science Theory, September 27, 2010, Available at http://www.cs.columbia.edu/~aho/cs3261/lectures.

[2] S. H. Rodger. Jap web site, 2011. www.jflap.org.

[3] Hao Wang, Student Member, IEEE, Shi Pu, Student Member, IEEE, Gabe Knezek, Student Member, IEEE, and Jyh-Charn Liu, Member, IEEE, MIN-MAX: A Counter-Based Algorithm for Regular Expression Matching, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 24, NO. 1, JANUARY 2013.

[4]Domenico Ficara, Member, IEEE, Andrea Di Pietro, Student Member, IEEE, Stefano Giordano, Senior Member, IEEE, Gregorio Procissi, Member, IEEE, Fabio Vitucci, Member, IEEE, and Gianni Antichi, Member, IEEE, Differential Encoding of DFAs for Fast Regular Expression Matching, IEEE/ACM TRANSACTIONS ON NETWORKING, VOL. 19, NO. 3, JUNE 2011.

[5]Domenico Ficara, Member, IEEE, Andrea Di Pietro, Student Member, IEEE, Stefano Giordano, Senior Member, IEEE, Gregorio Procissi, Member, IEEE, Fabio Vitucci, Member, IEEE, and Gianni Antichi, Member, IEEE, Differential Encoding of DFAs for Fast Regular Expression Matching, IEEE/ACM TRANSACTIONS ON NETWORKING, VOL. 19, NO. 3, JUNE 2011.

[6] Jean-Charles Delvenne and Vincent D. Blonde, Complexity of Control on Finite Automata, IEEE TRANSACTIONS ON AUTOMATIC CONTROL, VOL. 51, NO. 6, JUNE 2006.

[7]Attila Csenki, Flowgraph Models in Reliability and Finite Automata: IEEE TRANSACTIONS ON RELIABILITY, VOL. 57, NO. 2, JUNE 2008.

[8] R. W. Butler, "Reliabilities for feedback systems and their saddlepoint approximation," Statistical Science, vol. 15, pp. 279–298, 2000.

[9]Gruber H. and Holzer, M., "Provably shorter regular expressions from deterministic finite automata", LNCS, vol. 5257, pages 383–395. Springer, Heidelberg (2008).

[10] H. Gruber and J. Johannsen, "Optimal lower bounds on regular expression size using communication complexity", In Proceedings of the 11th International Conference Foundations of Software Science and Computation Structures, volume 4962 of LNCS, pages 273–286, Budapest, Hungary, March–April 2008