

State Minimization Algorithm to Remove Useless State in Deterministic Finite Automata

Mr. Gaurav Ojha¹, Akash Niras², Mr. Uday Singh^{3*}

^{1,2}Research scholar, Department of computer science and engineering, Azad Institute of Engineering and Technology, Lucknow, U.P India

³Assistant Professor Department of computer science and engineering, Azad Institute of Engineering and Technology, Lucknow, U.P India

ABSTRACT

This report contemplated a new technique for efficiently calculating and removing useless states in finite-state automata. Some state is not useful in deterministic finite automata means these type of state doesn't participant for generating useful string. And these types of state are called dead state, unreachable state or indistinguishable state. And those state increase the complexity but nothing, our work is primarily about the removal of those state to make DFA more efficient in some way. It is observed that neither unreachable states nor dead states can participate in scanning any valid token. We have therefore eliminated all such states as in order to optimization process. In deterministic finite automata, it is not easy to determine dead state, unreachable state or inaccessible state and it is necessary for removing unreachable state and dead state from DFA (deterministic finite automata) and hence removing unreachable state and dead state from deterministic finite automata is very necessary for generating useful string. We can generate minimized deterministic finite automata after removing unreachable state, dead state and indistinguishable state. But it is very difficult to remove this type of state from DFA. Therefore first we chosen useful state. Apart from the primary goal our work also concerned about how useful automata package simulator and java formal languages could be for new technique. In new deterministic finite automata (after generating of useful state), unreachable state will not available. Generated useful state by new technique or new approaches with taking less input symbol. Also in this report proposed algorithm developed for removing unreachable state in deterministic finite automata. And, in this report compares different running approaches with input symbol.

Keyword: Token, Deterministic Finite Automata, Non-Deterministic Finite Automata, Lexime.

INTRODUCTION

A local automaton is a DFA for which all edges with the same label lead to a single vertex. Local automata 'accept' the class of local languages, those for which connected of a word in the language is finding by a "sliding window" of length two on the word [1].

A **Myhillgraph** [1] over an alphabet 'A' is a directed graph with vertex set A and subsets of vertices labeled "start" and "finish". The language accepted by a Myhill graph is the set of directed paths from a start vertex to a finish vertex: the graph thus acts as an automaton. The class of languages accepted by Myhill graphs is the class of local languages [4].

DFAs were invented to model 'real world' finite state machines in contrast to the concept of a Turing machine, which was too general to study properties of real world machines [3].

DFAs are one of the most practical models of computation, since there is a trivial sequential time, constant-space, online algorithm to simulate a DFA on a maximum of input. Also, there are efficient algorithms to find a DFA recognizing [7]:

- The complement of the natural language recognized by a given DFA.
- The union/intersection of the natural languages recognized by two given DFAs [5].

Because DFAs can be minimizing to a 'consequence form' (minimal DFAs), there are also effective algorithms to determine [1]:

- Any DFA accepts any strings

- Any DFA accepts all strings
- Any two DFAs recognize the same language
- The DFA with a lowest number of states for a particular regular language

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

A finite automaton (FA) M as the quintuple $M = (Q, \Sigma, \delta, q_s, F)$ where Q is a finite set of states [1] $\{q_i \mid i \text{ is a nonnegative integer}\}$

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 In the conversion of deterministic finite automata to non-deterministic finite automata, deterministic finite automata should be minimized because if deterministic finite automata is not minimized then conversion not execute well. Minimization of automata refers to that minimal state should be present in the deterministic finite automata [1]. In automata theory that is a branch of theoretical computer science, a **deterministic finite automaton (DFA)** also known as **deterministic finite state machine** is a finite state machine that accepts/rejects finite strings of symbols and only produces a unique computation (or run) of the automaton for each input string [10]. ‘Deterministic’ means to the uniqueness of the computation. In words, the first condition says that the machine starts in the start state q_0 . The second condition says that the given each character of string ‘w’, the machine will transition from state to state according to the transition function ‘ δ ’. The last condition says that the machine accepts ‘w’ if the last input of ‘w’ causes the machine to halt in one of the accepting states [9]. It is said that the automaton rejects the string. The set of strings M accepts is the language recognized by M and this language is denoted by $L(M)$ [9].

Proposed Technique: In this technique, first of all we have generated useful state before minimization. If we generate useful state then useless state (unreachable state) will remove simultaneously. In proposed approach initially we proposed initial state as a final state and obtain only one input symbol from initial state and move simultaneously with changing accepting input symbol state. In the fig3.1 input symbol is ‘a’, ‘b’ because outgoing symbol of state q_0 is ‘a’, ‘b’. Input symbol ‘a’ and ‘b’ both is rejecting so we can see View Trace by JFLAP simulator and finalized all state including accepting input symbol [6].

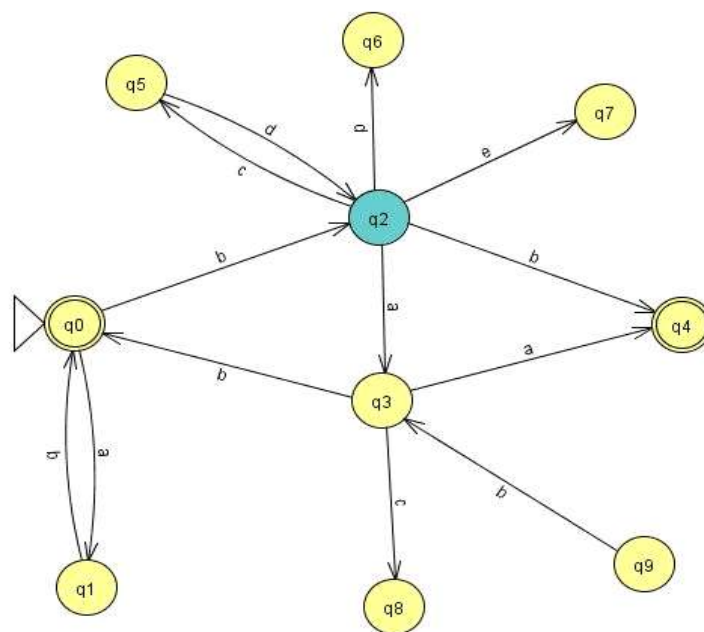
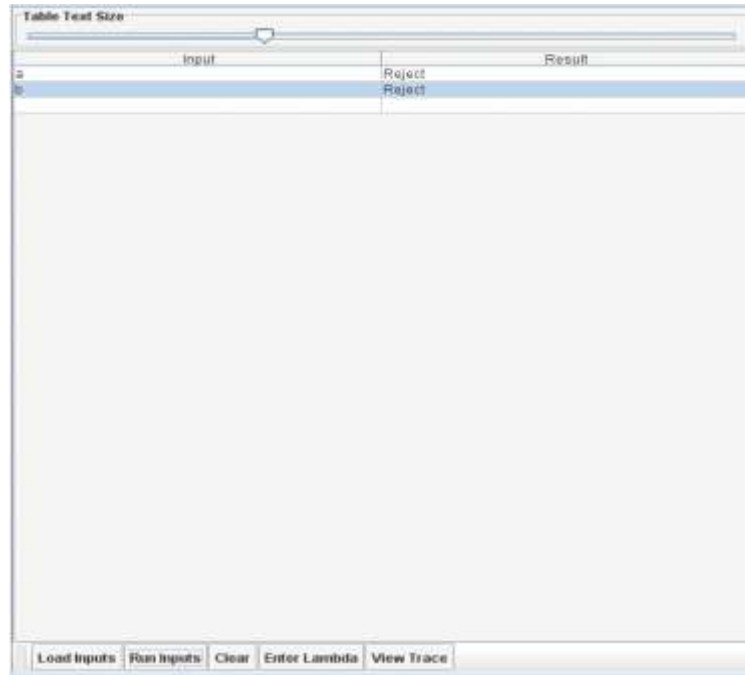


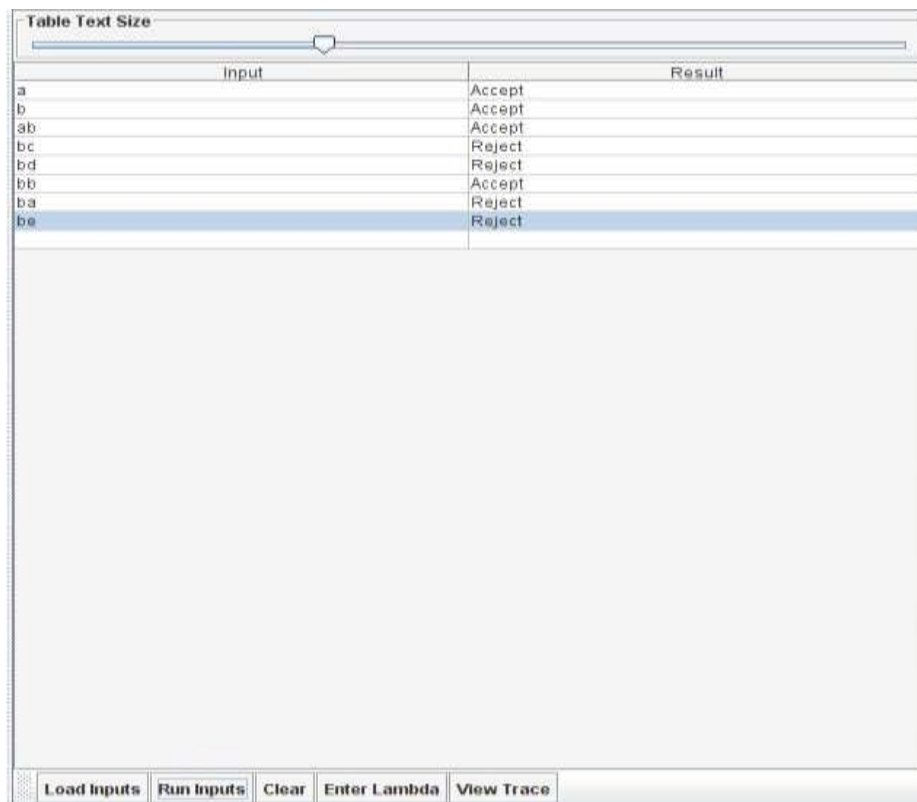
Fig 1: Deterministic Finite Automata with Useless State.



Input	Result
a	Reject
b	Reject

Fig 2: Multiple Run with 'a' and 'b' Input Symbols.

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 included at a time. As shown in side fig.



Input	Result
a	Accept
b	Accept
ab	Accept
bc	Reject
bd	Reject
bb	Accept
ba	Reject
be	Reject

Fig 3: Multiple Run with 'One Length' And 'Two Length' Input Symbols.

Proposed Algorithm :

INPUT: $A = (Q, \Sigma, \delta, q_0, q_f)$ – Deterministic finite automata.

OUTPUT: $A' = (Q', \Sigma, \delta', q_0', q_f')$ – Deterministic finite automata without unreachable state.

- For $(q \in Q)$ /*all state belong to given set of deterministic finite automata */

- Go to ($q_i \leftarrow$ Initial State) /* go to initial state */
- And MAKE $q_i \leftarrow$ Final State
- If ($\delta_{i-F_i} \rightarrow q_f$) /* this show transition function if any state reach to final state that means input string is accepted */
- MAKE $q_{i+1-F_i} \leftarrow$ Final State /* if any iteration for accepting state then all the state including in this iteration should be final state */
- Else
 $q_i \rightarrow q_{next}$ /*if input string is not accepted then move to another state q_{next} */
- END Else
- END if
- END For
- For ($q' \in q'_f$) /*after checking all possible input string according to proposed technique all useful state should be final state */
- If ($q' \notin q'_f$) Then /* unreachable state */
- $q' \leftarrow$ Remove /* remove unreachable state */
- END if
- END For
- For ($q'_i \in q'_f$) /* after removing unreachable state all useful state present and all state should be final state */
- if ($q_{initial} \in q'_f$) Then /* checking proposed initial state is final or not */
- $q_{initial} \leftarrow$ previous position /* if proposed initial state is final the it will form previous state */
- Else
 $q'_f \in q_f$ Then /* all proposed state is final state */
- $q'_f \leftarrow$ Previous position /* all proposed state become previous state */
- END Else
- END if
- END For.

Comparison Based on Iteration :This part of our work is very important for understanding which is better approach either running approaches or proposed approach. As shown below table (PART-1 and PART-2). In this table we can see for removing iteration if we will follow proposed technique. In this table mainly comparison between running approaches and proposed approached by taking iteration. We can see in running approaches there are more iteration than proposed technique. Primarily, proposed technique will applicable if more state is connected in deterministic finite automata and this is well form in removing unreachable state as well or we can apply this technique in generating useful state in deterministic finite automata.

Table 1: Comparison between DFS Approach and Proposed Approach for Removing Unreachable State In Deterministic Finite Automata

Running approaches for removing unreachable state from deterministic finite automata (DFA).		Proposed technique (approach) for removing unreachable state in deterministic finite automata (DFA).		
INPUT SYMBOLE(TAKING STRING)	NUMBER OF ITERATION MOVE ONE STATE TO ANOTHER FOR GIVEN STRING	INPUT SYMBLE (TAKING-STRING)	REDUSED INPUT SYMBOLE (TAKING STRING)	NUMBER OF ITERATION MOVE ONE STATE TO ANOTHER FOR GIVEN STRING
'bb'	2	'bb'		2
'baa'	3	'baa'	'baa'	3
'abbb'	4	'abbb'	'ab'	2
'bcdb'	4	'bcd'	'bcd'	3
'babbb'	5	'babbb'	'b'	1
'abbcdb'	6	'abbcdb'	'ab'	2
'abbaa'	5	'abbaa'	'ab'	2
'abbabbb'	7	'abbabbb'	'ab'	2
'bcdaa'	5	'bcdaa'	'bc'	2
'bcdabbb'	7	'bcdabbb'	'bc'	2
'acdabbb'	7	'acdabbb'	'acd'	3
'bcdaa'	5	'bcdaa'	'b'	1
'abbcdaa'	7	'abbcdaa'	'ab'	2

CONCLUSION

While choosing the useful state of deterministic finite state automaton is one of the most challenging concepts for students at an introductory level to understand and learn. In this work we primarily aimed at the removal of unreachable and dead state of deterministic finite automata. We have followed simple approach for generating useful state by proposed approach in this work. We have chosen JFLAP simulation for removal of inefficient state of deterministic finite automata. If the proposed technique apply for generating useful state then both unreachable state and dead state would simply removed. If have gone through with contemplated technique or approach, then the number of steps for taking input is lesser than running technique. After selecting useful state we have minimized simply of deterministic finite automata.

FUTURE WORK

In this work only unreachable state and dead state of deterministic state is proposed to be removed, But indistinguishable state has not been removed of deterministic finite automata. So for future work we can work on this particular dimension that means for removing indistinguishable state of deterministic finite automata. We can work on minimization of deterministic finite automata in future time and on state label as well in the future work in order to extend our 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.cscolumbia.edu/~aho/cs3261/lectures>.
- [2]. S. H. Rodger. Jap web site, and tutorial, 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.