Technique to Remove Indistinguishable State with Unreachable State and Dead State from Deterministic Finite Automata

Dipanshu Rastogi
Gov. Engineering College of Ajmer,
Rajasthan, India

Ravinder Singh
Department of Computer Science
Engineering and Information Technology
Gov. Engineering College of Ajmer,
Rajasthan, India

ABSTRACT
This paper presents a new technique for efficiently calculating and remove indistinguishable states in finite-state automata. A central problem in automata theory is to minimize a given Deterministic Finite Automaton (DFA). DFA minimization is an important topic because it can be applied both theoretically and practically, in for instance compilers. Minimizing a DFA increases its efficiency by reducing its amount of states and it also enables us to determine if two DFAs are equivalent. A DFA(deterministic finite automata) have some redundant state that means this type of state doesn't participant for generating useful strings. And these types of state are called dead state, unreachable state or indistinguishable state. 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 removing unreachable state and dead state from deterministic finite automata is very necessary to generating useful string. We can generate minimize deterministic finite automata after removing unreachable state, dead state and indistinguishable state. But it is very difficult to removing these type of state from DFA. Then first we will choose useful state. This paper also explaining about how useful automata package simulator and java formal languages for new technique.

Keywords
Automata, Deterministic finite automata, Unreachable State, Dead State.

1. INTRODUCTION
Automata: It is define as a system where some information, material or energy is transmitted, transformed or used to perform actions without the actual participation of man. And in other words we can describe as 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 can be represented by a 5-tuple (Q, Σ, δ, q_o, F), where [3, 4, 5]

1. Q is a finite nonempty set of states.
2. Σ is a finite nonempty set of input called the input alphabet.
3. δ is the next state function, $\delta : D \rightarrow 2^Q$ where D is a finite subset of $Q \times \Sigma^*$
4. q_o : initial state ; $q_o \subseteq Q$
5. F :set of final states ; $F \subseteq Q$

Note that, above definition is valid for both DFAs (deterministic finite automata), and NFAs (nondeterministic finite automata)[6,7].

We will discussing shortly about deterministic finite automata (DFAs), and we will discuss on later about nondeterministic finite automata.

Deterministic Finite Automata: DFA's are called deterministic because following any input string, we know exactly which state it's in and the path it took to get there.[8]

Deterministic finite automata (DFA) can be described by 5-tuples (Q, Σ, δ, q0, F), where

Q is a finite non-empty set of states

Σ is a finite non-empty set of symbols

δ is the next state function, that is, $\delta: Q \times \Sigma \rightarrow Q$.

q0 is the initial state; $q_o \subseteq Q$

F is a set of final states of Q (i.e. $F \subseteq Q$) called accept states [9].

Transition functions can also be represented by transition table as shown in table 1.1. A finite automata is represented by ({0, 1, 2}, {0}, δ, {0}, {2}) where, δ is shown in the following table [10].

Table 1.1: Transition Table representing transition function of DFA

State (Q)	Next State δ(q,0)
0	1
1	2
2	2

Transition function can also be represented by transition diagram as shown below in figure 1.1.

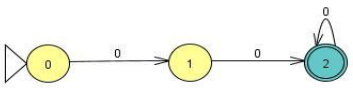


Figure 1.1: Deterministic finite automata corresponding to table 1.1.

2. PROBLEM STATEMENT

The problem of finding the redundant state in deterministic finite automata. Different types of technique and approaches are available for generating useful state in DFA. Indistinguishable state is the one of the major issue for DFA(deterministic finite automata). For removing the dead state, unreachable state, indistinguishable state some approaches are available. First of all we know about what is the unreachable state, dead state and indistinguishable state.

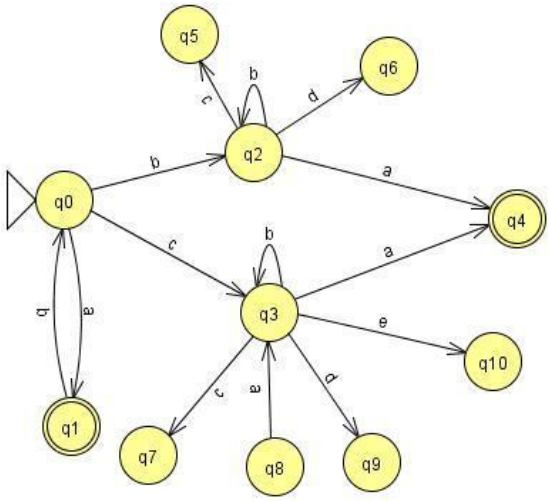
Unreachable state: All those states which can never be reached from initial state are called inaccessible states or unreachable state.

Dead state: All those non final state which transit to itself for all input symbol in Σ are called Dead state.

Indistinguishable state: State p and q are indistinguishable if, staring in p and q, every string leads to the same state of "finality" (i.e., the string fail or succeed together.)

- $\delta^*(p, w) \in F \Rightarrow \delta^*(q, w) \in F$, and
- $\delta^*(p, w) \notin F \Rightarrow \delta^*(q, w) \notin F$,
- for all string $w \in \sum^*$

As shown in below figure:



unreachable state, dead state and indistinguishable state.

In the above figure state q8 is unreachable state because if we will take any string from initial state q0 to q8 then it is not possible. And state q5, q6, q7, q9 and q10 are dead states because no any transition from another state than itself. State q2 and q3 are indistinguishable states because q2 and q3 use same input symbol for reach the final state. 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 q3 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 q3. 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 this state is an unreachable state. Problem of this approach is it will taking more and more time for removing unreachable state.

3. PROPOSED TECHNIQUE

In the proposed technique, for generate useful state before minimization we have to remove all redundant state from deterministic finite automata. This technique is based on when indistinguishable state have unreachable state and dead state. And in the pair of indistinguishable we have to remove one state that is equivalent to each other state. One of the merit of this technique is we remove which state that have move unreachable state and dead state. Then unreachable and dead state are automatically remove from DFA.

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.

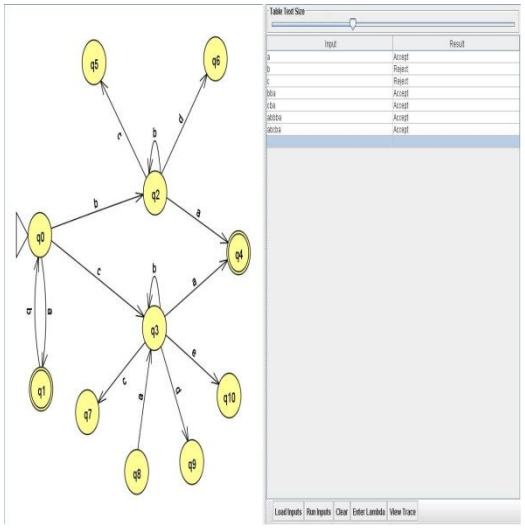


Figure 3.1 Deterministic finite Automata.

Step 1. Find Indistinguishable State:

In the above figure, taking outgoing input symbol from q0 are a , b , c , bba , cba , abbba , abcba and in these input symbol a , bba , cba , abbba , abcba accepting symbol and b , c are rejecting symbol. We can see view trace by JFLAP simulator [11]. In the accepting symbol (bba , cba , abbba , abcba) ba string is common. We reach the final state from these states {qo,q2,q4}, {q0,q3,q4},{q0,q1,q2,q4},{q0,q1,q3,q}and q2 ,q3 states are satisfy the condion for indistinguishable state.

i.e., $\delta (q2 , ba) \in F \Rightarrow \delta (q3 , ba) \in F$

$ba \in \sum^*$

So q2 and q3 are equivalent state then we have to remove one state for generating useful state.

Step 2. Check which indistinguishable state have more unreachable and Dead state:

Check for q2 :

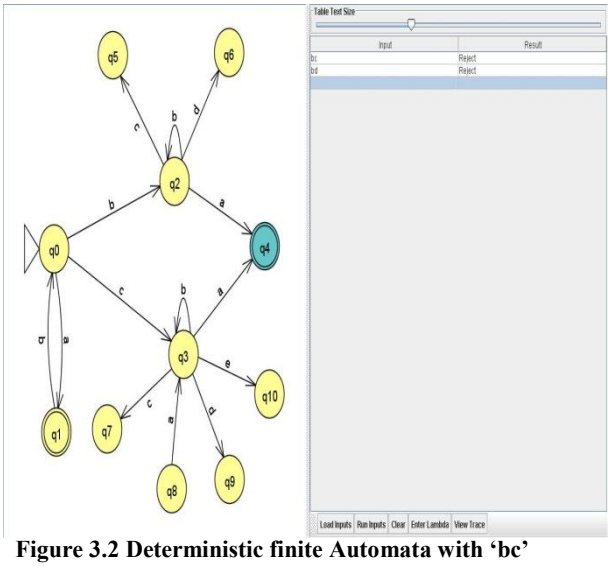


Figure 3.2 Deterministic finite Automata with 'bc' and 'bd' input symbol.

In above figure, no any state generating accepting input symbol. 'bc' and 'bd' are rejecting input symbol. So we can see view trace by JFLAP simulator.

So both q5 and q6 are dead state.

Check for q3:

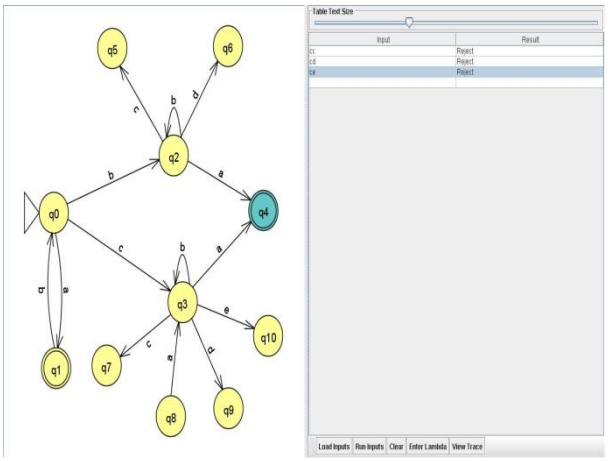


Figure 3.3 Deterministic finite Automata with 'cc', 'cd', 'ce' input symbol.

In above figure, no any state generating accepting input symbol. 'cc', 'cd' and 'ce' are rejecting input symbol. So we can see view trace by JFLAP simulator.

So both q7, q9 and q10 are dead state. And finally q3 have more dead state than q2 then we have to remove q3 and connecting string from q0 to q3 is 'c' that is merge with q2 state.

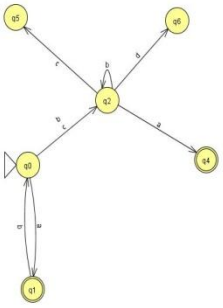


Figure3.4 Deterministic finite Automata with no any indistinguishable state.

Step 3. Remove Dead state and unreachable state:

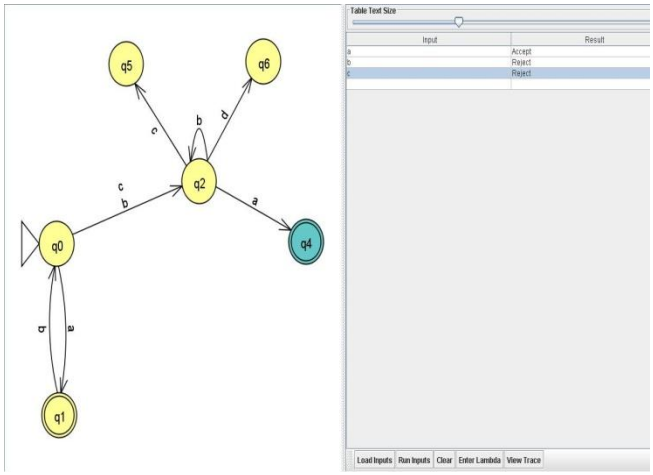


Figure 3.5 Deterministic finite Automata with 'a', 'b' and 'c' input symbol.

In this figure input symbol is 'a', 'b' and 'c' because outgoing symbol of state q0 are 'a', 'b' and 'c'. Input symbol 'a' is accepting but input symbol 'b' and 'c' are rejecting so we can see View Trace by JFLAP simulator and finalized all state including accepting input symbol. As shown in below figure.

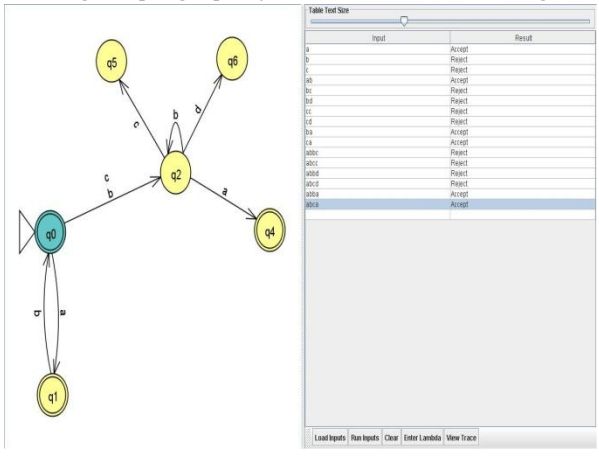


Figure 3.6 Deterministic finite Automata with all possible input symbol.

In above figure we finalize state q0 because this state generating accepting input symbol with final state and take next input symbol. In this figure we are taking all possible input symbol and we can see view trace of JFLAP simulator that shown input symbol 'ab', 'ba', 'ca', 'abba', 'abca' are accepting and others are rejecting. And then finalized all the state including in accepting input symbols. As shown in the figure.

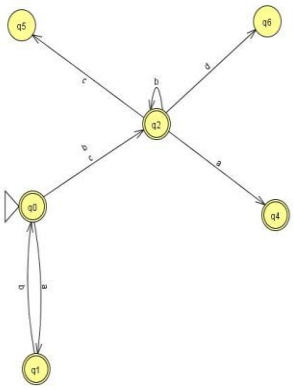


Figure 3.6 Deterministic finite Automata with after finalized all the state including in accepting input symbol.

In above figure state q1 is finalizing because this state including accepting input symbol 'ba', 'ca' 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. When all possible input 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.

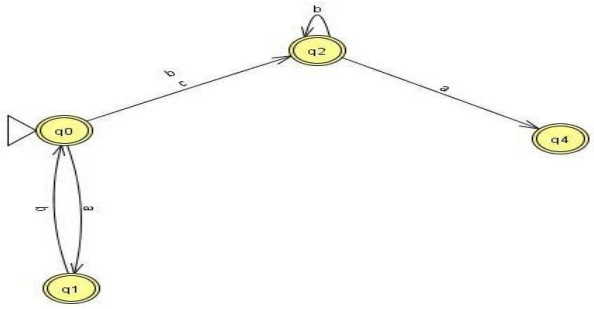


Figure 3.7 deterministic finite Automata with unreachable state and dead state.

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 and q4 so these state are not same as a previous form and all updated label should be remove. As shown in below figure.

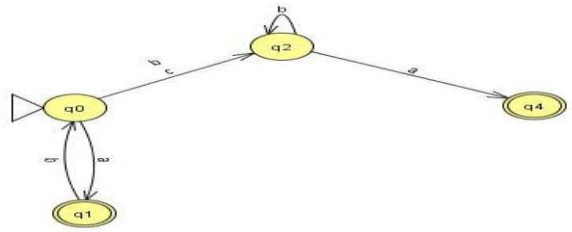


Figure 3.8 Deterministic finite Automata with useful state.

In above figure all useful state available means only whose state is available those are including in accepting string and all Dead state remove in this process. So this is a proposed technique for removing unreachable state, Dead state and indistinguishable state of deterministic finite automata.

4. 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.

1. For ($q_{\epsilon Q}$) /*all state belong to given set of deterministic finite automata *\
2. Go to (q_i <- Initial State) /* go to initial state *\
3. If ($\delta_{i\text{-}Fi}$ ->q_f) /* this show transition function if any state reach to final state that means input string is accepted *\
4. If (δ (q_m , a) $\in$ F = = (δ (q_n , a) $\in$ F || δ (q_m , a) $\notin$ F = = (δ (q_n , a) $\notin$ F /* this show these state are indistinguishable. And n= 1,2,3,……; m= 1,2,3……; m≠n; a (input symbol) $\in \sum$*(non-empty finite set of input symbols) *\
5. If (q_m -> $q_{m\ next} \notin$ F > q_n -> $q_{n\ next} \notin$ F) /* this show q_n have more dead state.
6. q_n<-Remove, /* remove indistinguishable state and connecting input symbol from q_n to its privious state is merge with q_m state.*\
7. Else

q_m<-Remove,/* remove indistinguishable state and connecting input symbol from q_m to its previous state is merge with q_n state.*\

8. Go to (q_i <- Initial State) /* go to initial state after removing the indistinguishable state for dead state *\
9. Else

Go to (q_i <- Initial State) /* go to initial state *\

10. END Else
11. END if
12. MAKE $q_{i\text{-}Fi}$ <- Final State /* if any iteration for accepting state then all the state including in this iteration should be final state*\
13. Else

$q_{i\ ->}\ q_{next}$ /*if input string is not accepted then move to another state q_{next} *\

14. END Else
15. END if
16. END For
17. For (q' € q'_f) /*after checking all possible input string according to proposed technique all useful state should be final state*\
18. If (q' €! q'_f) Then /* unreachable state*\
19. q' <- Remove /* remove unreachable state*\
20. END if
21. END For
22. For (q'_i € q'_f) /* after removing unreachable state all useful state present and all state shuld be final state *\
23. if($q_{initial}$ € q'_f) Then /* checking proposed initial state is final or not *\
24. $q_{initial\ <-}$ previous position /* if proposed initial state is final the it will form previous state*\
25. Else

$q'_{f\ \epsilon}\ q_f$ Then /* all proposed state is final state*\

26. q'_f <- Previous position /* all proposed state become previous state*\
27. END Else
28. END if
29. END For.

5. 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 indistinguishable state with 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 determine indistinguishable state and remove unuseful 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 minimize simply of deterministic finite automata.

6. FUTURE WORK

In this paper, we remove indistinguishable state when indistinguishable state have only unreachable and dead state of Deterministic finite automata. But in this paper is not for when indistinguishable state have reachable state that means any state reach to final state that means input string is accepted and this state is connected to indistinguishable state.

7. 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, 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 saddle point 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.

[11] M. Procopiuc, O. Procopiuc, and S. Rodger, Visualization and Interaction in the Computer Science Formal Languages Course with JFLAP, *1996 Frontiers in Education Conference*, Salt Lake City, Utah, p. 121-125, 1996