International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868
Foundation of Computer Science FCS, New York, USA
Volume 8– No.6, April 2015 – www.ijais.org

LR Rotation Rule for Creating Minimal NFA

Himanshu Pandey
M.Tech
Department of Computer Science, BBD University, Lucknow, India.

V. K Singh, Ph.D.
HOD
Department of Information Technology, BBDNITM, Lucknow, India.

Neeraj Kumar Verma
Department of Computer Science, SVNIET, Lucknow, India.

ABSTRACT

The problem of creating a minimal NFA is a primal (fundamental) problem. Reducing the size of NFA by using LR rotation rule has been shown to reduce importantly the search time. In [1] Ilie and Yu describe a construction of a right invariant equivalence relation on the states of a non-deterministic finite state automaton. We give a more efficient LR Rotation rule for constructing the minimal NFA. In this paper we represent new LR Rotation rule for the state minimization of NFA. The description of the proposed methods is given and we also shown the results of the numerical experiments.

We conceive the problem of reducing the number of state and transition of Non Deterministic Finite Automata. Numerical experiments show that NFA reduction algorithm produces a minimal automation in all most cases. NFA reduction algorithm also reduces the complexity of Kameda-Weiner algorithm. We have shown empirically that these algorithms are effective in largely reducing the memory requirement of NFA minimization algorithm and algorithm minimization of the number of rules for NFA grows each year.

Keywords

NFA, Algorithm

1. INTRODUCTION

A refined phenomenon offer rules to efficiently solve typical problems by mapping them to regular expression, then getting NFA that recognize them and finally constructing Deterministic Finite Automata. LR Rotation Rule exploits unique features of the minimized NFA to achieve high throughput. By merging states many complex problem has been efficiently solved. A more challenging alternative is directly minimizing the NFA before changing it into a DFA. This has the advantage of working over a much smaller structure (of size polynomial in the length of the regular expression) and of building the smaller DFA without the need to go through a larger one first. However, the NFA state minimization problem is hard. In this paper, we inquire pertinence a formal LR Rotation rule to reduce the number of states and transitions in NFA. The paper presents a LR Rotation rule, with a different reduction power and time complexity. The idea of reducing the size of NFAs by merging states was first introduced by Ilie and Yu [12] who used left and right equivalence relations. Later, Champarnaud and Coulon [2] modified the idea to work for preorders. An algorithm to compute the equivalences in O(m log n) time on an NFA with n states and m transitions and an O(mn) algorithm. Our focus and main contribution is a study of LR rotation techniques for creating minimal NFA. In particular, the state minimization of deterministic finite automata (DFAs) is well-known but the state minimization of nondeterministic finite automata (NFAs) is more complicated. Finite automata (FA) are widely used in various fields and peculiarly reorganization of formal Languages. We provide some necessary definitions.

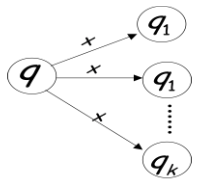
A nondeterministic finite automaton (or NFA) can be formally defined as a 5-tuple (Q, Σ, T, q_D, F) Where,

-Q is a finite set of states.

-Σ is the alphabet (defining what set of input strings the automaton operates on).

-T: $Q\times(\Sigma \cup \lambda) \rightarrow Q$ is the transition function.

$$\delta(q, x) = \{q_1, q_2, \ldots, q_k\}.$$



(Resulting states with following one transition with symbol x)

-$q_D \in Q$ is the starting state.

-$F \subset Q$ is a set of final (or accepting states).

Finite automata may be used to recognize and define the regular languages. Two automata are called equivalent if they recognize one and the same language. For each NFA the equivalent DFA may be constructed using the powerset construction process (each state of such DFA is a subset of states of the original NFA).

NFAs represent regular languages, and can be used to test whether any string is in the language it represents.

The above mentioned algorithms identify sets of states that could be merged without modifying the language accepted by the automaton. The number of states of the resulting NFA depends on the order in which the states are merged.

Randomly choosing the order in which these mergings take place, as used so far, does not guarantee that the smallest NFAs that can be built with these techniques are produced.

In this paper we investigate optimal ways to use the information in equivalences and preorders to reduce NFAs. We first give an efficient algorithm for optimally combining the left and right equivalences for achieving the maximum reduction in the size of an NFA. We show that the same problem for preorders, however, is NP-hard. Since, potentially, preorders could produce a better reduction, a number of open problems remain, such as looking for alternative ways, e.g., approximation algorithms, to reduce NFAs using preorders.

2. RELATED WORK

The paper by Michael Albert and Steve Linton [6] on "A Practical Algorithm for Reducing Non-Deterministic Finite State Automata" give a more efficient algorithm for constructing the same equivalence, together with results from a computer implementation". We are inspired by the work of Guangming Xing [1], highlights that no auxiliary states can be eliminated without violating the defining properties of Thompson NFA in his paper "Minimized Thompson NFA". In this paper 'Reducing the Size of NFAs by Using Equivalences and Preorders' Lucian lie, Roberto Solis-Oba, Sheng yu clubbed the concept of Equivalences and Preorders for minimization of NFA. Hermann Gruber and Markus Holzer [8] proposed the investigation of computational complexity of the nondeterministic finite automaton (NFA) minimization problem for finite and unary regular languages; they show this on his paper "Computational Complexity of NFA Minimization for Finite and Unary Languages". The paper 'Local Search Heuristics for NFA State Minimization Problem* by Andrey V. Tsyganov [9] introduce new heuristic methods for the state minimization of nondeterministic finite automata. These methods are based on the classical Kameda-Weiner algorithm joined with local search heuristics. We have used this concept for minimization of NFA. The concept of using Hash Table for minimization of DFA is very useful concept for creating a minimal DFA. This concept is given by Vishal Garg, Anu in 2013. Yi Liu, Taghi M. Khoshgoftaar [12] in DFA Minimizing State Machines Using Hash- Tables. Henrik Bj orklunda, Wim Martens [13] have shown that no such significant extensions exist, under the assumption that PTIME 6= NP. and also proved that minimization is NP-hard for all finite automata classes that contain the NFAs that accept strings of length three. In this paper 'NFA Reduction for Regular Expressions Matching Using FPGA' Vlastimil Koˇsaˇr, Martin ˇZ ´adn´ık, Jan Koˇrenek [11] proposed to accelerate regular expression matching via mapping of a nondeterministic finite automaton into a circuit implemented in an FPGA. These algorithms exploit unique features of the FPGA to achieve high throughput. Paper by Manuel Vázquez de Parga, Pedro García, Damián López [14] proposed a polynomial-time deterministic finite automaton minimization algorithm directly derived from Brzozowski's double reversal algorithm. We take into account the framework by Brzozowski and Tamm, to propose an atomization algorithm that allows us to achieve polynomial time complexity. We are inspired by the work of Jean Vuillemin, Nicolas Gamaon present a cubic time algorithm to reduce a xor-automaton A 2 NXA to a minimal form M = MXA(A) which accepts ¤©(M) = ¤©(A), within the least possible number of states. It is a finite strong normal form SNF: ¤© (A) = ¤© (A0), MXA (A) = MXA (A0) and automata equivalence is efficiently decided through reduction to the SNF. Wojciech Wieczorek clubbed a Supercomputers with NFA, treated the induction of NFAs based on finite languages. That is constituted by the following task: given two disjoint finite sets S+; S of words and an integer k > 0, build a k-state NFA that accepts the language S+ and does not accept any word from the set S□. We are inspired by the work of C. Hsiang Chan, R. Paigeb [10] overcome drawbacks of both methods with a O(r) time O(s) space algorithm to construct an O(s) space representation of McNaughton and Yamada's NFA. Given any set V of NFA states, our representation can be used to compute the set U of states one transition away from the states in V in optimal time 0(1 V I+ 1 U I). McNaughton and Yamada's NFA requires O (1 VI x I UI) time in the worst case.

3. OUR ALGORITHM

By the help of L-R Rule we have to minimize the non Deterministic finite automata. If we have a NFA with following condition:

'p' and 'q' are two states of any NFA. Such that p, q Є Q (non empty finite set of states).

There must be no edges between two adjacent (p, q) states.

For that type of condition we will use L-R rule for creating minimal NFA.

LR rule is based on merging two states 'p' and 'q' (having no edges between them) on the basis of following rule.

$L_R(p) \subseteq L_R(q)$ or $L_R(q) \subseteq L_R(p)$

L (p, p) and L (q, q) = Ǿ or same string

p and q can be merged.

$L_L(p) \subseteq L_L(q)$ or $L_L(q) \subseteq L_L(p)$,

L (p, p) and L (q, q) = Ǿ or same string

p and q can be merged.

$L_R(p) \not\subset L_R(q)$ and $L_L(q) \not\subset L_L(p)$

L (p, p) and L (q, q) ≠ Ǿ or same string

p and q can be merged.

L_R= right side input of incoming or outgoing transition of any state.

L_L= left side input of incoming or outgoing transition of any state.

We can merge two states p and q as soon as any of the above condition is met.

If any one of these conditions will applicable on any given NFA, states (p, q) can be merged but there must be no edge between 'p' and 'q'.

(1). for condition 1, find the right side input of incoming or outgoing transition of state 'p' and 'q' and then check for-

$L_R(p) \subseteq L_R(q)$ or $L_R(q) \subseteq L_R(p)$

If these condition exist in given NFA, then

Check for, L (p, p) and L (q, q) = Ǿ or

L (p, p) and L (q, q) = {string}

If condition 1 satisfied then 'p' and 'q' can be merged for given NFA.

If condition 1 will not satisfy, we will check condition 2.

For condition 2, find the left side input of incoming or outgoing transition of state of 'p' and 'q' and check for-

$L_L(p) \subseteq L_L(q)$ or $L_L(q) \subseteq L_L(p)$, if its exist, then

Check for, L (p, p) and L (q, q) = Ǿ or

L (p, p) and L (q, q) = {string}

If condition 2 is satisfied then ‘p’ and ‘q’ can be merged.

If condition 2 is not satisfied, check for condition 3.

For condition 3, find both right and left side input of incoming or outgoing transition of state ‘p’ and ‘q’ and then check for-

L_R (p) ⊄ L_R (q) and L_L (q) ⊄ L_L (p), if its exist then check for,

L (p, p) and L (q, q) ≠ Ǿ or

L (p, p) and L (q, q) = {string}

If condition 3 is satisfied then ‘p’ and ‘q’ can be merged.

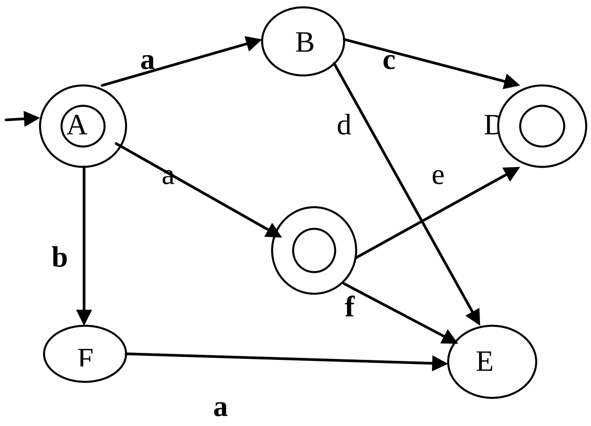


Figure 1: Transition Graph of NFA

Reduction of given NFA using LR rule-

By LR rule, we try to reduce NFA by merging the states. In this rule we have taken two states that do not have any edge between them.

In given NFA ‘B’ and ‘C’ are two states which have not consist any edge between them. Now we will check all the condition for the states B and C,

Condition1: Firstly we will find the right side input of incoming or outgoing transition of state ‘ B’ and ’C’ as L_R (B) and L_R (C).

L_R (B) ={c} and L_R (C) = {e}

But, L_R (B) ⊊ L_R (c)

{c} ⊊ {e}

Condition is not satisfied so state B and C cannot be merged. Now we will go for condition 2.

Condition 2: In this condition we will find the left side input of incoming or outgoing transition of state of ‘B’ and ‘C’ as L_L (B) and L_L (C).

L_L (B) = {a}

L_L (C) = {a}

Check for L_L (p) ⊆ L_L (q)

L_L (B) ⊆ L_L (C)

{a} ⊆ {a}

, condition exists then. Check for L (B, B) and L(C, C) =Ǿ

There is no self loop on the state B and C.

Condition satisfied. So ‘B’ and ’C’ can be merged. We need not to verify condition 3.

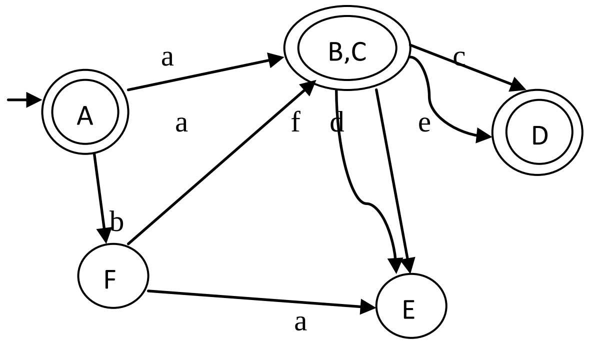


Figure 2: Reduced NFA having merge states B and C

In above transition graph of NFA states B and C have been merged.

In the following transition graph there are two transitions (c. e) between the states D and (B, C). By using LR rule of state transition, we can merge the both transition in single one.

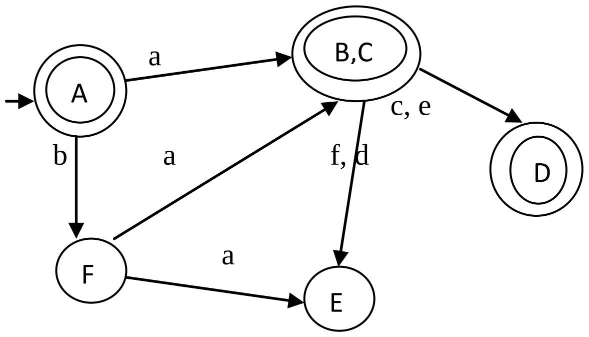


Figure 3: Reduced NFA having merged transition c and e

In the given NFA, there are another two states (D and E) having no edge between them. In States ‘D’ and ‘E’ has only left side input of incoming or outgoing transition. So condition one cannot be applied.

Check for condition 2.

Condition 2: left side input of state ‘D’ and ‘E’ will be,

L_L (D) = {c, e}, L_L (E) = {f, d}

Check for condition-

L_L (p) ⊆ L_L (q)

L_L (D) ⊄ LL (E) {D is not subset of E}

{c, e} ⊄ {f, d} so condition is not satisfied.

Check for condition 3.

Condition 3: Left and right side input of state ‘D’ and ‘E’

L_L (D) = {c, e} L_L (E) = {d, f}

Right side input is not available for the state D and E. so,

L_L (D) ⊄ L_L (E)

Now, check for L (D, D) and L (E, E) =Ǿ

There is no self loop on state D and E

So ‘D’ and ‘E’ cannot be merged.

4. CONCLUSION

In the present paper we have represent the LR Rotation rule for NFA state minimization problem which is known to be computationally hard. We wish to point out that our algorithm for computing the reduced NFA only finds the best way of merging states with respect to the LR Rotation rule. It is possible to reduce the size of an NFA by first merging some equivalent states. That type of algorithm is widely used in combinatorial optimization. The essential feature of the proposed algorithm is that the most time consuming part of the exact algorithm is replaced with LR Rotation rule. Numerical experiments have shown that such type of concept is much less time consuming and allows obtaining acceptable results. In the future we plan to concentrate on the other time consuming part of preorder concept. Therefore, reductions with LR Rotation Rule, potentially more powerful than those with equivalences, might be too expensive to compute. This opens a new research topic: designing efficient approximation algorithms for using LR Rotation Rule in reducing NFAs, and testing their performance in practice.

5. REFERENCES

[1] L. llie, Roberto Solis-Oba, Sheng yu: 2005,"Reducing the Size of NFAs by Using Equivalences and Preorders", Lecture Notes in Computer Science, Volume 3537, 2005, pp 310-321 in Springer.

[2] M. Albert and Steve Linton: July 2014, "A Practical Algorithm for Reducing Non-Deterministic Finite State Automata", Elsevier Science.

[3] A V. Tsyganov: September 2012, "Local Search Heuristics for NFA State Minimization Problem", *Int. J. Communications, Network and System Sciences*, 2012, 5, 638-643.

[4] H. Gruber and M. Holzer: "Computational Complexity of NFA Minimization for Finite and Unary Languages",Institut f¨ur Informatik, Ludwig-Maximilians-Universit¨at M¨unchen, Oettingenstraße 67, D-80538 M¨unchen, Germany.

[5] H. Bj¨orklunda, Wim Martens: April 2011, "The Tractability Frontier for NFA Minimization", International Colloquium on Automata, Languages and Programming 2008.

[6] Y. Zhou Yuliu Chen, "The QFD-based Decision-making Approach for Strategic BPR" Beijing 100084, P. R. China.

[7] G. Xing, August 20-22, 2007 "Minimized Thompson NFA", Western Kentucky University, Bowling Green, KY 42101.

[8] V. Garg, Anu: June 2013, "DFA Minimizing State Machines Using Hash- Tables", International Journal of Engineering Trends and Technology (IJETT) - Volume4 Issue6- June 2013.

[9] V. Koˇsaˇr, Martin ˇZ ´adn´ık, Jan Koˇrenek, 2007 "NFA Reduction for Regular Expressions Matching Using FPGA", MSM 0021630528, the IT4Innovations Centre of Excellence CZ.1.05/1.1.00/02.0070 and the grant BUT FIT-S-11-1.

[10] M. Vázquez de Parga, P. García, Damián López, 2013 "A polynomial double reversal minimization algorithm for deterministic finite automata", Theoretical Computer Science 487 (2013) 17–22.

[11] J. Vuillemin, N. Gama: Dec 2009, "Efficient Equivalence and Minimization for Non Deterministic Xor Automata", [Research Report] 2010, pp.25. <Inria-00487031>.

[12] W. Wieczorek: 2012, "Induction of Non-Deterministic Finite Automata on Supercomputers", JMLR: Workshop and Conference Proceedings 21:237{242, 2012. The 11th ICGI.

[13] M. Mohri, F. Pereira and M. Riley, "AT&T General-Purpose Finite-State Machine Software Tools," 1997. http://www.research.att.com/sw/tools/fsm

[14] S. Lombardy, R. Poss, Y. Régis-Gianas and J. Sakarovitch, "Introducing VAUCANSON," In: O. H. Ibarra and Z. Dang, Eds., *Implementation and Application of Automata*, CIAA 2003, Santa Barbara, 16-18 July 2003, pp. 96-107.

[15] S. H. Rodger, "JFLAP: An Interactive Formal Languages and Automata Package," Jones and Bartlett Publishers, Inc., USA, 2006.

[16] T. Kameda and P. Weiner, "On the State Minimization of Nondeterministic Finite Automata," *IEEE Transactions on Computers*, Vol. C-19, No. 7, 1970, pp. 617-627. doi:10.1109/T-C.1970.222994

[17] J. Hromkovic, "Algorithmics for Hard Problems—Introduction to Combinatorial Optimization, Randomization, Approximation, and Heuristics," Springer, Berlin, 2001.

[18] F. Glover and G. A. Kohenberger, "Handbook of Metaheuristics," Kluwer Academic Publishers, Boston, 2003.

[19] V. Kell, A. Maier, A. Potthoff, W. Thomas and U. Wermuth, "AMORE: A System for Computing Automata, Monoids and Regular Expressions," Proceedings of the 6th Annual Symposium on Theoretical Aspects of Computer Science on STACS 89, Springer-Verlag, New York,