Regular Expression to Deterministic Finite Automata

Shravani Bahirat[1] , Vaishnavi Dalvi[2], Sakshi Darawade[3], Shravani Jagtap[4], Sakshi Shakhawar[5]

[1,2,3,4,5]Student Dept. of Artificial Intelligence & Data Science , Vishwakarma Institue of Technology, Maharashtra, India

---***---

Abstract - *Regular expression is a sequence of characters(strings) which is also called as language that is accepted by a finite automata. Regular Expression consists of characters which are useful for defining filter. Practical applications of regular expressions are it is used in pattern matching for text editors, compiler design, etc. Expression in regular expression corresponds to a language and are well defined. Deterministic finite automata or the (DFA) are finite set of states and transitions that decides whether to accepts or rejects set of characters mainly called as strings or languages. Applications of DFA are it is included in protocol analysis, text parsing vending machine, video game character behaving, security analysis, NLP, speech recognition, elevators etc. DFA represents any system's which require an internal definition state. DFA represents a maching that accepts the specific RE. In DFA, after a particular input is provided to first state, it will pass on to one state only. To conver regular expression to DFA methods like transitive closure, state elimination and brzozowski algebraic method using arden's theorem are introduced. DFA is represented by digraphs called state diagram*

Key Words **- Theory of Computation, Deterministic Finite Automata, String passing**

1. INTRODUCTION

Regular expression is a shortened way of representing a language, which is a condition or filter to be accepted by Deterministic finite Automata. In order to speed up the process of converting regular expressions to finite state machines and vice versa, various options should be explored. Thus, we have tried to facilitate the assignment of Re-to-DFA conversion by enabling a step-by-step conversion from RE to DFA. In this review, various approaches to converting deterministic finite automata into regular expressions are briefly compared. The most common and least burdensome to the environment is the earth elimination method. Our interactions with generation are simple: In most cases, the scanners produced are the right size. The larger RE length allows a smaller scanner to be specified, which is useful for symbolic operations such as inductive statistics types and pattern machines.

2. LITERATURE REVIEW

[1]The editorial "Visual Designing and Debugging of Deterministic FiniteState Machines" in FSM This article shows how the DFA visualisation tool was created to create a trustworthy DFA. The user can also create the state transitions while the execution is taking place using the visualisation tool. Users can edit the states of their finite state machines (FSMs), which can then be rendered as executable code. Additionally, it states that the state-based machine visualisation component of the software is exclusive to code generation. It suggested a method to visualise machine verification and debugging for formal machines in accordance with design.

[2] A Review Paper on String Identification Using Finite Automata It demonstrates how the finite automata model keeps track of the user-supplied input strings for transition stages. It determines whether the user-provided input strings contain letters, numbers, or valid symbols. Computation is the process of receiving information and carrying out operations in accordance with a set of rules. This system uses lexical analysis technologies to reduce errors and increase performance levels for completing activities.

[3] On Implementing Deterministic Finite Automata in Parallel A system known as the deterministic finite automat is utilised to solve numerous procedures and jobs in the field of computer technology. DFA is compatible with two different parallel computer architectures. This work presents the computer-based parallel simulation of the general DFA, which employs automata.

[4] On Implementing Deterministic Finite Automata in Parallel A system known as the deterministic finite automat is utilised to solve numerous procedures and jobs in the field of computer technology. DFA is compatible with two different parallel computer architectures. This work presents the computer-based parallel simulation of the general DFA, which employs automata.

[5] . Ho Ngoc Vinh and Nguyen Thi Thu Ha together compiled a paper in which they have depicted the conversion of NFA to DFA using the concept of bounded words

[6]In this paper,concepts of bounded words on an alphabet A,languages and monoid morphism are introduced.

[7] A work in CSESpring18 that uses the Gallier's technique for NFA to DFA conversion was coauthored by Ghafoor[3]. It makes use of the different C++ data structures, including vectors, parser classes, etc. The DFA tuple, which is actually represented as strings, is the output that is ultimately produced.

[8]To test the system Dynamic system through automata learning it presented a dynamic testing method.. It generate and analyze the pages generated by the application to execute test by creating a web application. It also presents a case study that establishes dynamic testing procedure work.

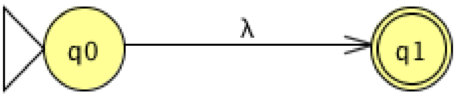
[9]This paper introduced model od Finite state automata to monitor whether the words are included in the text are sensitive or not in paper names as "Application research of Finite Automata in Distance Education". The method described the existence of large numbers of keywords in the text string which is well suited for the application and this method is further used to improve the system efficiency. For ignoring mistakes in strings or in word error recognitions exist in this method which is used to reduce the error rate and improve the performance of the system.

[2] addressed challenges to compile large set of patterns it may be in different data structure. It helps runtime to match input string to appropriate patterns effectively. Some time input string matches more than one pattern. This is main challenge in efficient classification of strings. To tackle with above problem sequential approach can be used but main problem with this approach is it will take more time as pattern increases. This paper also suggested the refinements so that decision tree compilation algorithm will be time effective. Sailesh Kumar et al.

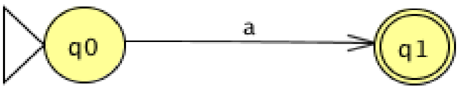
3. METHODOLOGY

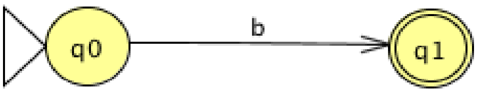
For every language there is atleast one regular expression and multiple DFA's that represent the language. The algorithm below shows conversion of Regular Expression to Deterministic Finite Automata(DFA). It initially converts RE to NFA that is non deterministic finite automata and then to equivalent DFA. In this process of conversion every possible case is considered

- NFA for two-state transition which signifies a language makes a empty string

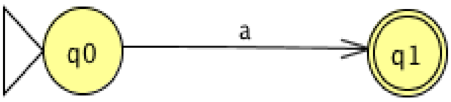


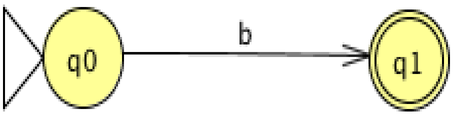
- NFA comprised for α transition to represent literal characters.



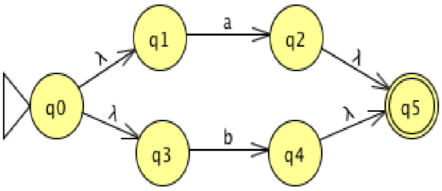


- Union of two languages forms from the NFA used to represent languages.By α transition initial state is linked to the R nd S, and the final state is connected to the Final states of R and S by ε-transitions.



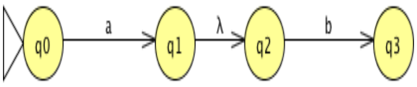


- **a | b**



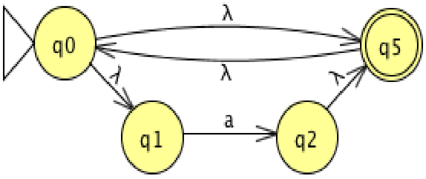
- Concatenation of two lnguages results into the NFA of language, which have the linking of transition final state R to initial state S.

- **ab**



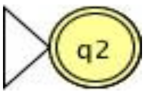
- An NFA with Kleen star language is the ε-conversion from its initial state to the initial state of R, a ε-transition from its final state to the accepting state of R, and ε-transitions between its initial and accepting states.

- a*

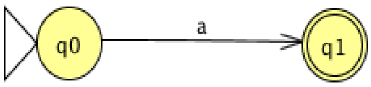


- Here each can be a valid DFA so it can be endangered towards the NFA to DFA algorithm. Equivalent DFA for the previously developed NFA shown below.

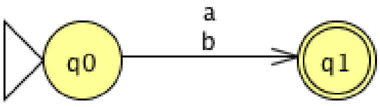
ε



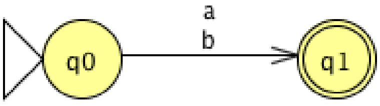
- **a**



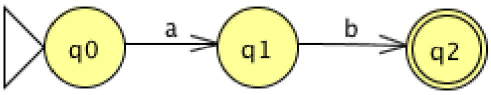
- **b**



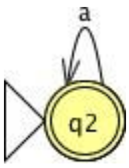
- **a|b**



- **ab**



- **a***

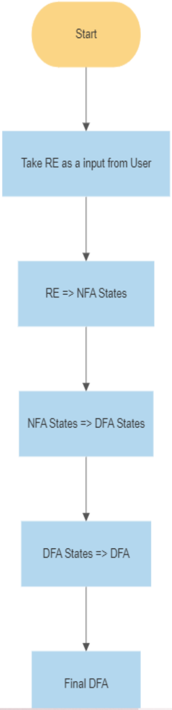


3.3 Algorithm

• Step 1 – Convert RE to NFA using standard methods such as Thompsons algorithm.

• Step 2 – Now convert NFA to DFA using standard algorithms that is a subset construction in which every one state have one subset of states of the NFA

3.4 Flowchart



4. RESULTS AND DISCUSSIONS

Various techniques such as transitive closure, Brzozowski algebra, and state elimination have been invented to convert deterministic finite automata to regular expressions. However, these techniques are efficient work giving minimal regular expressions. Our main motive is to find the minimal RE corresponding to a given DFA. The most commonly used and efficient approach to convert DFA to RE is state removal approach. This article explores and equates different techniques used to transform regular expressions into deterministic finite automata.

4.1 Results

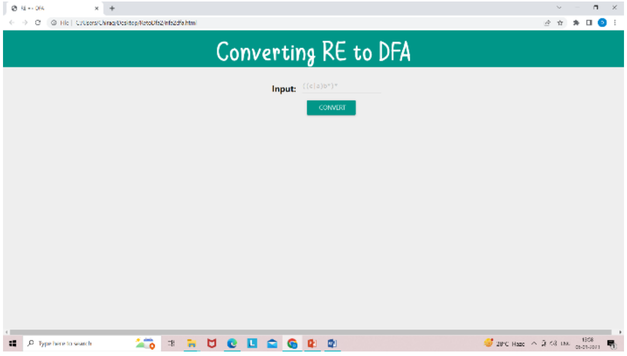


Fig.1 Input screen for conversion

In the given snapshot, there is the input screen where we accept the Regular expression from the user to convert it into the Deterministic Finite Aotomata (DFA)

4.2 LIMITATIONS

- Limited algorithms are implemented.
- RE gives the NFA Table states but not the proper NFA diagram.

4.3 CONCLUSION

This study gives readers an understanding of the many methods used to translate Regular Expressions into deterministic finite automata. There are comparisons made between several methods for converting RE to DFA. Research in this project shows that the procedure for converting regular expressions to DFAs and vice versa is well understood and easy to implement. Coding the parser for the regular expression took the most time in the project. Due to the fact that regular expressions create regular languages but are not themselves regular, they require context-free grammars to be defined.

4.4 FUTURE SCOPE

- In future we can upgrade this project by making it more user friendly and adding graphics interface.
- As this project gives output after compiling the whole result,In the future,we can calculate the output of NFA also.
- we can add DFA to RE

7. REFERENCES

[1] A Review Paper On Finite Automata Application in String Identification (PDF) EasyChair Preprint A Review Paper on Finite Automata Application in String Identification A Review Paper On Finite Automata Application in String Identification (researchgate.net)

[2] On Parallel Implementations of Deterministic Finite Automata On Parallel Implementations of Deterministic Finite Automata | SpringerLink

[3] Kuldeep Vayadande, Aditya Bodhankar, Ajinkya Mahajan, Diksha Prasad, Shivani Mahajan, Aishwarya Pujari and Riya Dhakalkar, "Classification of Depression on social media using Distant Supervision", ITM Web Conf. Volume 50, 2022.

[4] Kuldeep Vayadande, Rahebar Shaikh, Suraj Rothe, Sangam Patil, Tanuj Barware and Sameer Naik," Blockchain-Based Land Record System", ITM Web Conf. Volume 50, 2022.

[5] Kuldeep Vayadande, Kirti Agarwal, Aadesh Kabra, Ketan Gangwal and Atharv Kinage," Cryptography using Automata Theory", ITM Web Conf. Volume 50, 2022

[6] Samruddhi Mumbare, Kunal Shivam, Priyanka Lokhande, Samruddhi Zaware, Varad Deshpande and Kuldeep Vayadande," Software Controller using Hand Gestures", ITM Web Conf. Volume 50, 2022.

[7] Preetham, H. D., and Kuldeep Baban Vayadande. "Online Crime Reporting System Using Python Django."

[8] Vayadande, Kuldeep B., et al. "Simulation and Testing of Deterministic Finite Automata Machine." International Journal of Computer Sciences and Engineering 10.1 (2022): 13-17.

[9] Vayadande, Kuldeep, et al. "Modulo Calculator Using Tkinter Library." EasyChair Preprint 7578 (2022).

[10] VAYADANDE, KULDEEP. "Simulating Derivations of Context-Free Grammar." (2022).

[11] Vayadande, Kuldeep, Ram Mandhana, Kaustubh Paralkar, Dhananjay Pawal, Siddhant Deshpande, and Vishal Sonkusale. "Pattern Matching in File System." International Journal of Computer Applications 975: 8887.

[12] Vayadande, Kuldeep, Ritesh Pokarne, Mahalakshmi Phaldesai, Tanushri Bhuruk, Tanmay Patil, and Prachi Kumar. "Simulation Of Conway's Game Of Life Using Cellular Automata." SIMULATION 9, no. 01 (2022).

[13] Gurav, Rohit, Sakshi Suryawanshi, Parth Narkhede, Sankalp Patil, Sejal Hukare, and Kuldeep Vayadande. "Universal Turing machine simulator." International Journal of Advance Research, Ideas and Innovations in Technology, ISSN (2022).

[14] Vayadande, Kuldeep B., Parth Sheth, Arvind Shelke, Vaishnavi Patil, Srushti Shevate, and Chinmayee Sawakare. "Simulation and Testing of Deterministic Finite Automata 21 Machine." International Journal of Computer Sciences and Engineering 10, no. 1 (2022): 13-17.

[15] Vayadande, Kuldeep, Ram Mandhana, Kaustubh Paralkar, Dhananjay Pawal, Siddhant Deshpande, and Vishal Sonkusale. "Pattern Matching in File System." International Journal of Computer Applications 975: 8887.

[16] Vayadande, Kuldeep B., and Surendra Yadav. "A Review paper on Detection of Moving Object in Dynamic Background." International Journal of Computer Sciences and Engineering 6, no. 9 (2018): 877-880.

[17] Vayadande, Kuldeep, Neha Bhavar, Sayee Chauhan, Sushrut Kulkarni, Abhijit Thorat, and Yash Annapure. Spell Checker Model for String Comparison in Automata. No. 7375. EasyChair, 2022.

[18] Vayadande, Kuldeep, Harshwardhan More, Omkar More, Shubham Mulay, Atharva Pathak, and Vishwam Talnikar. "Pac Man: Game Development using PDA and OOP." (2022).

[19] Preetham, H. D., and Kuldeep Baban Vayadande. "Online Crime Reporting System Using Python Django."

[20] Vayadande, Kuldeep. "Harshwardhan More, Omkar More, Shubham Mulay, Atahrv Pathak, Vishwam Talanikar,"Pac Man: Game Development using PDA and OOP"." International Research Journal of Engineering and Technology (IRJET), eISSN (2022): 2395-0056.

[21] Ingale, Varad, Kuldeep Vayadande, Vivek Verma, Abhishek Yeole, Sahil Zawar, and Zoya Jamadar. "Lexical analyzer using DFA." International Journal of Advance Research, Ideas and Innovations in Technology, www. IJARIIT. com.

[22] Manjramkar, Devang, Adwait Gharpure, Aayush Gore, Ishan Gujarathi, and Dhananjay Deore. "A Review Paper on Document text search based on nondeterministic automata." (2022).

[23] Chandra, Arunav, Aashay Bongulwar, Aayush Jadhav, Rishikesh Ahire, Amogh Dumbre, Sumaan Ali, Anveshika Kamble, Rohit Arole, Bijin Jiby, and Sukhpreet Bhatti. Survey on Randomly Generating English Sentences. No. 7655. EasyChair, 2022

[24] F. Le Fessant and L. Maranget, "Optimizing pattern matching," ACM SIGPLAN Not., vol. 36, no. 10, pp. 26–37, Oct. 2001, Doi: 10.1145/507669.507641.

[25] R. M. Karp and M. O. Rabin, "EFFICIENT RANDOMIZED PATTERN-MATCHING ALGORITHMS.," IBM J. Res. Dev., vol. 31, no. 2, pp. 249–260, 1987, Doi: 10.1147/RD.312.0249.