Analysis of Bangla-2-Braille Machine Translator

Syed Akhter Hossain
Department of Computer Science and Engineering
Daffodil International University
Dhaka, Bangladesh
aktarhossain@{daffodilvarsity.edu.bd, yahoo.com}

Lora Annanya Biswas and Md Iqbal Hossain
Department of Computer Science and Engineering
Daffodil International University
Dhaka, Bangladesh
lora_iqbal15-1327@diu.edu.bd

Abstract— **Bangla-2-Braille machine translator is implemented using rule based Discrete Finite Automata (DFA) [1]. Rule based DFA directs the necessity of generating regular expression from the DFA in order to validate the language model used for Braille conversion and extend the translator for better usability by the visually impaired people. In this work, we conducted experiments using structured and state elimination method to generate, and validate regular expressions from the DFA designed for Bangla-2-Braille machine translator. The generated expressions were tested for Braille language rules and the results were satisfactory.**

Keywords— DFA, Regular Expression, Braille, Bangla, Machine Translator

I. Introduction

A deterministic finite automaton (DFA)—also known as deterministic finite state machine—is a finite state machine that accepts or rejects finite strings of symbols and only produces a unique computation of the automaton for each input string [2]. The word 'Deterministic' refers to the uniqueness of the computation. In general the theory of computation is the branch of computer science and mathematics that deals with efficient problem solving based on a model of computation using an algorithm. In theoretical computer science, automata theory represents abstract machines for the computational problems that can be solved using different techniques [2].

On the other hand, automata theory [2, 3] is closely related to formal language theory as the automata are often classified by the class of formal languages they are able to recognize. The study of automata is an important part of core of Computer Science. Automata is used in designing and checking the behavior of digital circuits, in lexical analysis phase of compiler construction, in software for scanning large bodies of text and in software for verifying systems of all types that have finite number of distinct states, such as communications protocols [2].

In the same context, regular expressions [2, 3] also denote regular languages, which consist of strings of particular type. The patterns of strings described by regular expression are exactly same as what can be described by finite automata. It means every formal language defined by any finite automata is also defined by a regular expression.

Bangla is one of the most spoken languages (ranking sixth) in the world, with nearly 300 million total speakers.

According to Titumir[1] about 0.75 million people are blind in Bangladesh. Blind person cannot read text written on a plain paper and use Braille system in their reading and writing. Braille is a universal code for mapping character sets of various languages to Braille cells. Series of Braille cells are embossed on paper so that visually impaired persons can read them by touching the raised dots of the cells using their fingers. Like other languages Bangla has its own representation for Braille system [1].

The Braille system is a method that is widely used by blind people to read and write. Braille was devised in 1821 by Louis Braille, a blind Frenchman. Each Braille character or cell is made up of six dot positions, arranged in a rectangle containing two columns of three dots each. A dot may be raised at any of the six positions to form sixty-four (26) permutations, including the arrangement in which no dots are raised. For reference purposes, a particular permutation may be described by naming the positions where dots are raised, the positions being universally numbered 1 to 3, from top to bottom, on the left, and 4 to 6, from top to bottom, on the right [1].

An example can be given by naming the raised dot positions. For example if it is said like dots 1-2-3-5, it describes dots 1, 2 and 3 are raised in first column and dot 5 is raised in second column. This particular permutation indicates English letter R as shown in Fig. 1.

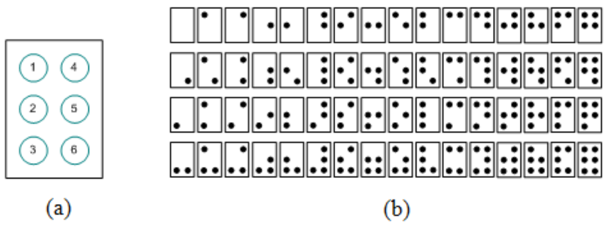


Figure 1. (a) A Braille cell, (b) 64 permutations

In this work, the DFA designed for Bangla-2-Braille machine translator is analyzed and regular expression is generated using state elimination method. This paper is organized as follows: Section 2 explains related works about regular expression and DFA of Bangla-2-Braille. Experimental method is described in section 3. In section 4 regular expression of Bangla-2-Braille is elaborated from the analysis and findings with the testing of the regular expression is elaborated. Finally the conclusion and discussion is outlined in Section 5.

II. Regular Expression and DFA of Bangla-2-Braille

Deterministic finite automaton (DFA) is a finite state machine accepting finite strings of symbols. For each state, there is a transition arrow leading out to a next state for each symbol [4]. A DFA has a finite set of states and a finite set of input symbols. One state is designated the start state, and zero or more states are accepting states. A transition function determines how the state change each time an input symbol is processed. The term 'deterministic' refers to the fact that on each input there is one and only one state to which the automaton can transition from its current state [4, 5].

It is convenient to represent automata by a graph in which the nodes are the states and arcs are labeled by input symbols indicating the transitions of that automaton. The start state is designated by an arrow and the accepting states by double circles as shown in Fig.2 [2].

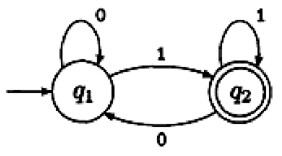


Figure 2. A discrete finite automata

A DFA represents a finite state machine that recognizes a RE. A deterministic finite automaton is a 5-tuple, (Q, Σ, δ, q0, F) [6], consisting of

1. a finite set of states (Q)
2. a finite set of input symbols called the alphabet (Σ)
3. a transition function (δ : Q × Σ → Q)
4. a start state (q0 ∈ Q)
5. a set of accept states (F ⊆ Q)

A regular expression (RE) is a pattern that describes some set of strings [6]. Regular expressions are used when you want to search for specify lines of text containing a particular pattern. The DFA shown in Fig. 2 is recognized by (0*1*)*. Asterisk (*) repeats the previous item zero or more times. Greedy, so as many items as possible will be matched before trying permutations with less matches of the preceding item, up to the point where the preceding item is not matched at all [6, 7].

The DFA of Bangla-2-Braille is constructed according to Bangla text to Braille conversion grammar for tokenizing input Bangla text. Each accepting states indicates that a token has been found with applicable rules. The constructed DFA in "five tuple" notation is DFA, A = (Q, Σ, δ, q0, F).

- Here A is the name of the DFA.
- Q is the set of states.

Q = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24,
25}

- Σ is input symbols. We have seven input symbols. Each of them represents set of Bangla characters.
- q0 is the start state of the DFA. q0 = state 0
- F is the set of accepting states.

F = {2, 4, 10, 11, 12, 13, 14, 15, 19, 22, 24, 25}

- δ is the transition function that takes as arguments a state and an input symbol and returns a state. In a transition diagram each transition from one state to another or same state for a specific input symbol also represents a transition function. It means transition function δ(a,x) = b is a transition from state a to state b for input symbol x in transition diagram where state a and b can be same or different state. In the proposed DFA, we have 182 (7 input symbols X 26 states) transition functions. All these transition functions are represented in the transition diagram of Fig. 3.

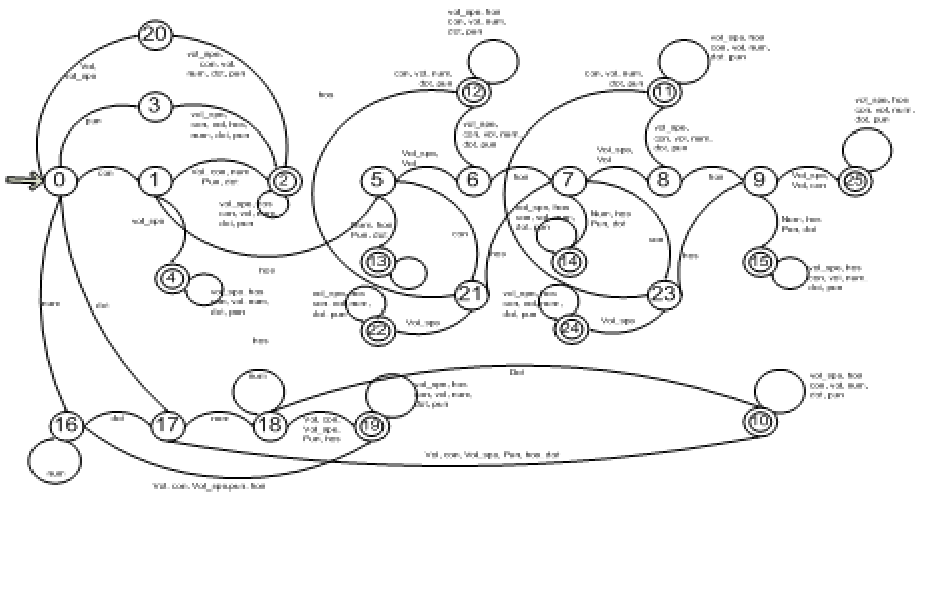


Figure 3. Computational Model for Bangla-2-Braille converter [1, 4]

III. Experimental Methodology

In this research work, the DFA of Bangla-2-Braille is analyzed for the generation of the regular expression using a step-wise refinement methodology shown in the Fig. 4.

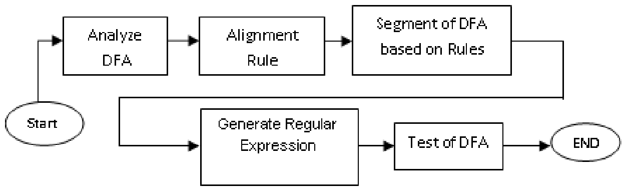


Figure 4. Experimental Methodology

The different blocks shown in the methodology performing different task. In the analysis of DFA stage, the DFA of Bangla-2-Braille shown in Fig. 3 are analyzed according to Bangla text to Braille conversion grammar [4, 5]. This analysis lead to understanding of the components for regular expressions.

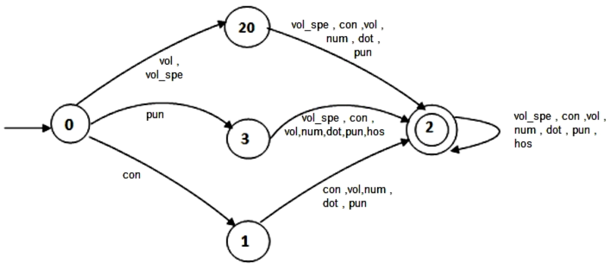
In the next alignment rule section, different grammatical rules used in the machine translation of Bangla-2-Braille is mapped to different part of the DFA in order to facilitate segmentation of DFA for later stage processing. Once identified, the DFA is segmented in the next stage based on each specific rule. In this stage, a computation validation is used to do necessary corrections. In the next stage, the segmented DFA is further processed for the generation of regular expression which is elaborated in the preceding sections.

IV. Regular Expression of DFA for Bangla-2-Braile

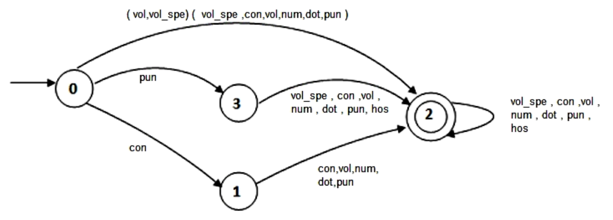
The state removal approach is used for converting Bangla-2-Braille DFA into a regular expression. In this approach, states of DFA are removed one by one until it is left with only starting and the final state. For each removed state regular expression is generated. The advantage of this technique over the transitive closure [6] method is that it is easier to visualize. In this approach, (a) if there are multiple edges from one node to other node, then there make a single edge by union [6, 7, 8], (b) if there are parallel edges from one node to other node, then there make a single edge by concatenation [6, 7, 8], (c) if there are many strings with repetition its mean same string can be selected more than once ,then use Kleene Closure (*) [7-10].

As an example, for final state 2 using the state removal method to convert the DFA to RE as follows:

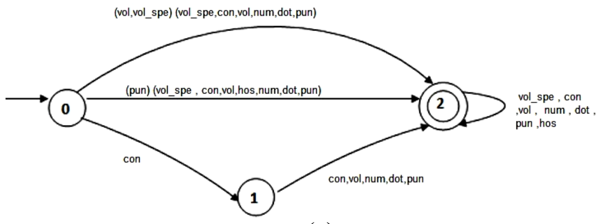
In order to convert the DFA shown in Fig. 5(a) into its corresponding regular expression (RE), this is required to eliminate the nodes step by step and in this case the removal sequence is 20-3-1. After removing the state 20, the corresponding DFA is shown in Fig. 5(b).


0
20
3
1
2
vol , vol_spe
vol_spe , con ,vol , num , dot , pun
pun
vol_spe , con , vol,num,dot,pun,hos
con
con ,vol,num , dot , pun
vol_spe , con ,vol , num , dot , pun , hos

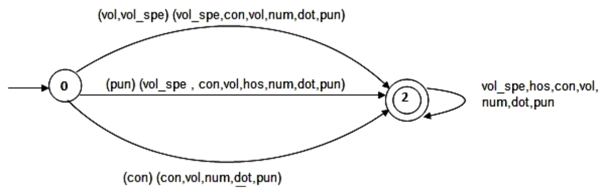
(a)


(vol,vol_spe) (vol_spe ,con,vol,num,dot,pun)
0
3
1
2
pun
vol_spe , con ,vol , num , dot , pun, hos
con
con,vol,num, dot,pun
vol_spe , con ,vol , num , dot , pun , hos

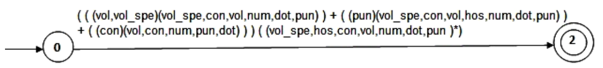
(b)


(vol,vol_spe) (vol_spe,con,vol,num,dot,pun)
(pun) (vol_spe , con,vol,hos,num,dot,pun)
0
2
1
con
con,vol,num,dot,pun
vol_spe , con ,vol , num , dot , pun ,hos

(c)


(vol,vol_spe) (vol_spe,con,vol,num,dot,pun)
(pun) (vol_spe , con,vol,hos,num,dot,pun)
0
2
vol_spe,hos,con,vol, num,dot,pun
(con) (con,vol,num,dot,pun)

(d)


(((vol,vol_spe)(vol_spe,con,vol,num,dot,pun)) + ((pun)(vol_spe,con,vol,hos,num,dot,pun)) + ((con)(vol,con,num,pun,dot))) ((vol_spe,hos,con,vol,num,dot,pun)*)
0
2

(e)

Figure 5. (a) An example of final state 2 (from Bangla-2-Braille DFA) (b)After removal of state 20 (from Bangla-2-Braille DFA)(c) after removal of 3 (d) after removal of 1 (e) final states

Applying the same technique, after removal of the state 3 and state 1, the final state diagram is shown in the Fig. 5(c), Fig. 5(d) and Fig. (e).

The final regular expression generated through this process of state elimination is shown in the (1).

RE= (((vol , vol_spe)(vol , vol_spe , con , num , dot , pun))+ ((pun)(vol , vol_spe , con , num , dot , pun)) + ((con)(vol , con , num , dot , pun))) ((vol , vol_spe , con , num , dot , pun)*) (1)

The different symbols used in (1) is based on the Table 1 shown below.

TABLE I. INPUT SYMBOLS AND CHARACTERS USED FOR BANGLA-2-BRAILLE TRANSLATION [1, 4]

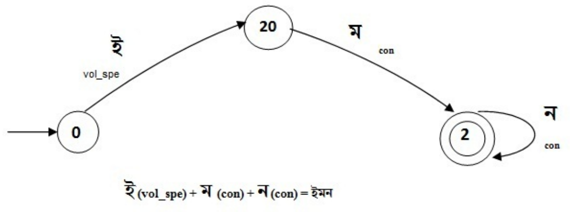
Input Symbols	Description	Characters
Vol_spe	Set of three independent vowels ই,উ,ও	ই, উ, ও
Vol	Set of independent vowels except ই,উ,ও and all dependent vowels	অ, আ, ঈ, ঊ, ঋ, এ, ঐ, ঔ, া, ি, ী, ু, ূ, ৃ, ে, ৈ, ো, ৌ
Con	Set of all Bangla consonants	ক, খ, গ, ঘ, ঙ, চ, ছ, জ, ঝ, ঞ, ট, ঠ, ড, ঢ, ণ, ত, থ, দ, ধ, ন, প, ফ, ব, ভ, ম, য, র, ল, শ, ষ, স, হ, ড়, ঢ়, য়, [illegible], ং, ঃ, ঁ
Num	Set of all Bangla digits	০, ১, ২, ৩, ৪, ৫, ৬, ৭, ৮, ৯
Pun	Set of all punctuation signs	, । ? ; : ‘ ’ “ ” - ! ( ) [ ]
Hos	Represents Bangla hasanta symbol	্
Dot	Represents decimal sign (.)	.

The regular expression is generated and evaluated for rules used in the DFA of Bangla-2-Braille as shown in the Table 2 below.

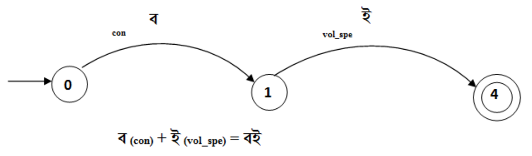
TABLE II. INPUT SYMBOLS AND CHARACTERS USED FOR BANGLA-2-BRAILLE TRANSLATION [1, 4]

Accepting States	Regular Expressions
2	(((vol , vol_spe)(vol , vol_spe , con , num , dot , pun))+ ((pun)(vol , vol_spe , con , num , dot , pun)) + ((con)(vol , con , num , dot , pun))) ((vol , vol_spe , con , num , dot , pun)*)
4	((con) (vol_spe) ((vol_spe ,vol , con , num , dot , pun)*)
11	(((con)(hos)) (((vol_spe,vol)(hos))+((con)(hos)))) ((vol_spe,vol)(vol_spe,vol,con,num,dot,pun)) + ((con)(con,vol,num,dot,pun)) ((vol_spe,vol,con,hos,num,dot,pun)*)
12	((con) (hos))(((vol_spe,vol) (vol_spe,vol,con,num,dot,pun)) + ((con)(vol_spe,vol,con,num,dot,pun,hos)))((vol_spe,vol,con,num,dot,pun,hos)*)
13	(con) (hos) (num,hos,pun,dot)
14	((con)(hos))(((vol_spe,vol)(hos))+((con)(hos)))(num,hos, pun,dot) ((vol_spe,vol,hos,con,pun,num,dot)*)
15	((((con)(hos)) (((vol_spe,vol)(hos)) +((con)(hos)))) (((vol_spe,vol)(hos)) + ((con)(hos)))) (num,hos,pun,dot)((vol_spe,vol,con,num,hos,pun,dot)*)
22	((con)(hos)(con)(vol_spe)) ((vol_spe,hos,con,vol,num,dot,pun)*)
24	(con)(hos)(((vol_spe,vol)(hos) + ((con)(hos))) ((con)(vol_spe)) ((vol_spe,vol,con,hos,num,dot,pun)*)
25	((((con)(hos)) (((vol_spe,vol)(hos)) +((con)(hos)))) (((vol_spe,vol)(hos)) + ((con)(hos)))) (vol_spe,vol,con) ((vol_spe,vol,con,num,hos,pun,dot)*)

The testing of the regular expression is also performed through state traversal. For the test of the State 2, where the regular expression is shown in Equation 1, the following Fig. 6(a) shows reconstruction using the regular expression for State 2. The similar test of State 4 is shown in the Fig. 6(b).



(a)



(b)

Figure 6. (a) Test of State 2 (b) Test of State 4

V. DISCUSSION AND CONCLUSION

The state removal approach is used for determining regular expressions of DFA used for Bangla-2-Braille via manual inspection in this work. The generated regular expressions are tested for the correctness and required validation of the generated Braille using Bangla-2-Braille machine translator. The transitive closure approach used in this work gives a clear and simple implementation, but tends to create very long regular expressions. In the future this regular expressions generation will be automated using the intelligent computation.

REFERENCES

[1] M A Karim, M Kaykobad, M Murshed, “Technical Challenges and Design Issues in Bangla Language Processing”, IGI Global, July, USA, 2013

[2] Mishra K.L.P.& N. Chandrasekaran, “Theory of Computer Science (Automata Language and. Computation)”, PHI, Second edition, 1998

[3] Ullman, J., J. E. Hopcroft and R. Motwani, “Introduction to Automata Theory, Languages, and Computation”. Pearson Education Inc, ISBN 0-201-44124-1. Addison Wesley, 2001

[4] Fakhruddin Muhammad Mahbub-ul-Islam, Ahamad Imtiaz Khan, Md Osman Gani, Samiul Azam, Syed Akhter Hossain, “Computational Model for Bangla Text to Braille Translation”, 2nd International Conference on Computer Processing of Bangla, Independent University Bangladesh, February, 2011

[5] Samiul Azam, Md Osman Gani, Ahamad Imtiaz Khan, Fakhruddin Muhammad Mahbub-ul-Islam, Syed Akhter Hossain, “Architecture and Implementation of Bangla Text to Braille Translation”, 2nd International Conference on Computer Processing of Bangla, Independent University Bangladesh, February, 2011

[6] M Rajasenathipathi, M. Arthanari, & M. Sivakumar, “Conversion of English Text to Braille Code vibration signal for Visually Impaired People”, International Journal of Computer Science and Information Security Publication, Vol. 8 No. 5, Paper 27071032, pp. 59-63, August, 2010

[7] Kulwinder Singh, “Conversion of Deterministic Finite Automata to Regular Expression using bridge state”, Masters Thesis, Thapar University, June, 2011

[8] G. Berry and R. Sethi. From regular expressions to deterministic automata. TCS: Theoretical Computer Science, 48:117, 1987

[9] Janusz A. Brzozowski. Derivatives of regular expressions. J. ACM, 11(4):481{494, 1964

[10] Ding-Shu Du and Ker-I Ko. Problem Solving in Automata, Languages, and Complexity. John Wiley & Sons, New York, NY, 2001