Journal of King Saud University – Computer and Information Sciences (2015) 27, 238–247

King Saud University

Journal of King Saud University – Computer and Information Sciences

www.ksu.edu.sa
www.sciencedirect.com

QEFSM model and Markov Algorithm for translating Quran reciting rules into Braille code

Abdallah M. Abualkishik [a,*], Khairuddin Omar [b], Ghadeer A. Odiebat [b]

[a] Sohar University, Oman
[b] University Kebangsaan Malaysia, Malaysia

Received 22 May 2013; revised 9 December 2014; accepted 21 January 2015
Available online 26 June 2015

KEYWORDS

Quran reciting rules;
Braille code;
Extended Finite State Machine;
Markov Algorithm;
Decision table

Abstract The Holy Quran is the central religious verbal text of Islam. Muslims are expected to read, understand, and apply the teachings of the Holy Quran. The Holy Quran was translated to Braille code as a normal Arabic text without having its reciting rules included. It is obvious that the users of this transliteration will not be able to recite the Quran the right way. Through this work, Quran Braille Translator (QBT) presents a specific translator to translate Quran verses and their reciting rules into the Braille code. Quran Extended Finite State Machine (QEFSM) model is proposed through this study as it is able to detect the Quran reciting rules (QRR) from the Quran text. Basis path testing was used to evaluate the inner work for the model by checking all the test cases for the model. Markov Algorithm (MA) was used for translating the detected QRR and Quran text into the matched Braille code. The data entries for QBT are Arabic letters and diacritics. The outputs of this study are seen in the double lines of Braille symbols; the first line is the proposed Quran reciting rules and the second line is for the Quran scripts.

1. Introduction

With the rapid growth of the technology over recent years, the way people interact with the outside world has changed and the gap between disabled and normal people has reduced. Lately, according to the World Health Organization (WHO), there are around 314 million people who live with visual impairment and 90% of them live in low level countries (WHO, 2014). Braille is a system of writing that uses patterns of raised dots to inscribe characters on paper. The original military code was twelve dot cells that were represented at six rows and two columns of dots called the night writing code, and it was used after dark as a communication way between the soldiers where the dot or relational dots represent a character, sound or a specific sign (Yamuna and Vora, 2013). Louis Braille improved the code and made the code easy and fast to read, where the soldiers were facing the difficulty reading the letters and signs that were represented by the twelve dots (Mellor, 2006). The six Louis dots were easier than the twelve military dots where it could be felt by the fingertips, and the new code had been accepted as a universal

* Corresponding author.
E-mail addresses: abdallah@uniten.edu.my (A.M. Abualkishik), ko@ftsm.ukm.my (K. Omar), g.odibat@yahoo.com (G.A. Odiebat).
Peer review under responsibility of King Saud University.

ELSEVIER Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.jksuci.2015.01.001
1319-1578

communication way between the blinds and the world around them, and as it has been translated for most languages, it becomes the World's blind language. As shown in Fig. 1; the Braille cell consists of six (6) dots arranged in the formation of a rectangle; three (3) down, two (2) across. Each Braille cell or symbol represents scientific characters, mathematical symbols, punctuations in music, and computer notations.

Previously in Abualkishik and Omar (2009a), we propose Braille symbols for Quran reciting rules in order to give the visual impaired people a chance to recite the Holy Quran correctly. The current work continues the previous work in Abualkishik and Omar (2009a) by proposing a Quran Extended Finite State Machine (QEFSM) model to detect all the reciting rules as found in the Quran text. Look-up table indexes QEFSM model states and decision table control the transaction function for the model. The outputs of the QEFSM model are sent to the Markov Algorithm for the translating process. Also, a Quran Braille Translator (QBT) is proposed to evaluate the QEFSM model work. The translated Braille code is printed out into a new proposed printed method. Each Quran line is translated into two Braille lines, the first Braille line represents Quran reciting rules' Braille symbols and the second line represents letters and diacritics Braille symbols.

2. Related works

Recently, technology explosion has had a positive effect on the increase of the human perception, especially for the third and second world countries. The Braille system is one of the systems that are important to the humanity. Braille gives a lot of young blind people the opportunity to participate in the scientific growth and automatically making them effective members in this world; but the blindness limited their ability. The Arabic language was translated to Braille code in Roy (2000) and Sensus, 1999. However, there is no specific system or research done with regard to the area of the Quran in Braille, and usually Braille Quran is printed out by unsystematic methods such as Perkins Braillewriter and Slate and Stylus. The learning of Quran reciting rules is an obligation for each Quran reader, where the need to learn and apply Quran reciting rules tends to affect the meaning of Holy Quran verses and causes change in the purpose of the verses. Al-Quran is GOD's Holy Book, and the correct understanding of the Quran text is imperative for all Muslims, including those who are disabled.

In King (2000), Alksander King concerns with translating the text to and from Braille (not handle the Arabic language), by using the matching of left and right contexts of the translation windows, with the finite state machine. The Finite state machine technique was used to handle the grades of Braille within the same language and to allow a single set of rules to assume the role as translation to and from Braille for a language. The decision table controls the operation of the finite state machine, and makes a simple list of character translation rules that can be edited directly by non-technical users. The UMIST translation system (Blenkhorn, 1997) is one of the few published works on text and Braille translation in recent years. Where the engine state is controlled by a finite state machine, the contents of the decision table are used, and which subset of the language translation rules to be used is regulated. The translation engine can use any language rule table, so any language can be translated to or from the Braille code if the language rule table is constructed.

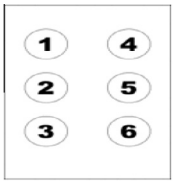

Figure 1 Braille cell (symbol).

3. Quran reciting rules

The Holy Quran was translated in many languages, but the original text was revealed to the Prophet in the Arabic language, (Malik, 2007). At Babker (1983) & Al-kari' (1998); There are around 21 reciting styles for the Holy Quran verses, but the most widespread reciting style in the Islamic world is Huff's reciting style. Quran reciting rules (QRR) are a set of rules that must be applied through Quran reciting. These rules change the pronunciation sound for Arabic language letters (Abualkishik & Omar, 2009b). This work is concerned with five main types of QRR: that are Noon Sakenah, Meem Sakenah, Lam Sakenah, Scoon (Kalkala) and Mudd (Prolong). These five main types are divided into thirteen subtypes.

As shown above in Fig. 2, there is a need for thirteen Braille symbols to represent the previous QRR but in order to prevent any conflict and to decrease the number of the new adopted symbols, six Braille symbols had been adopted to represent the thirteen Quran reciting rules. (The numbers inside the circles represent the number of rules set for each QRR, and there are one hundred twenty rules.) The reciting rules denote the pronouncing state for the letters which means that the same state (reciting rule) could apply to different letters, for instance see these two examples in Table 1.

The thirteen (13) QRRs' pronunciations could be classified into six (6) QRRs that are Edhar (Adhere), Edgham (Diphthong), Ekhfa' (Conceal), three types of Mudd (Prolong) and Kalkala. Through this study; five reciting rules are represented by one Braille symbol and one reciting rule (Mudd) is represented by two Braille symbols; first one for Mudd and the second for the prolong space. These symbols were proposed in Abualkishik and Omar (2009a) (see Table 2).

4. Quran Extended Finite State Machine (QEFSM) model

In Beesley and Karttunen (2002); the finite state machine (FSM) is the most suitable technique for linguistic and Natural Language Processing (NLP) applications. FSM is modularity (FSM is supporting and combining a variety of operations and relations gather), clear representation (FSM is implementing the rules directly and in a straightforward manner, so it is easy to understand and modify), efficiency (FSM is a deterministic technique that is almost non mysterious and gives a positive effect to improve time efficiency) and compactness (FSM can be minimized at the same time improving the storage requirement, time efficiency and the probability

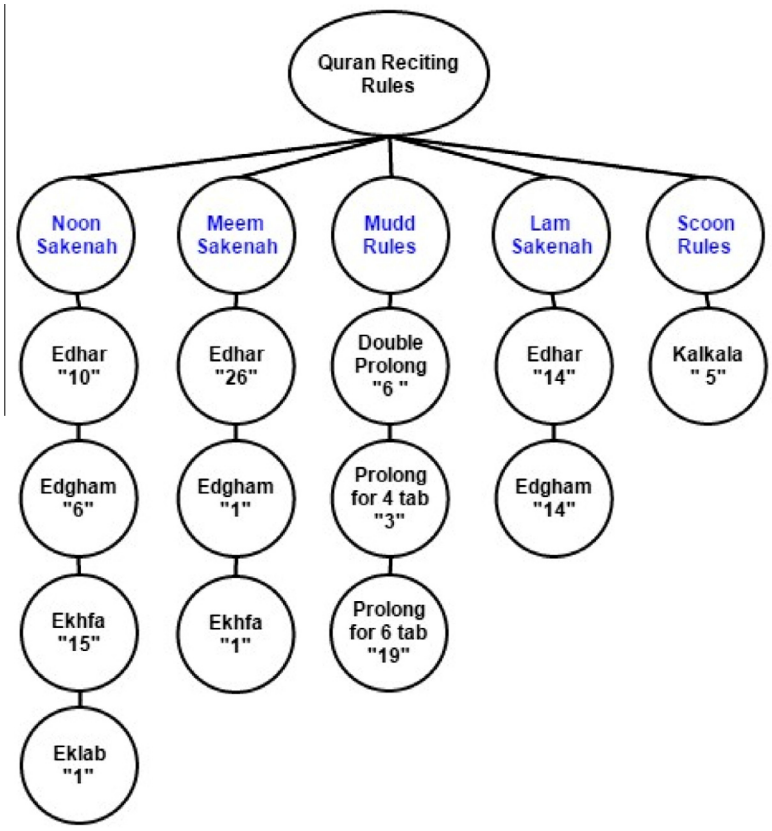

Figure 2 Quran reciting rules.

Table 1 Edhar reciting rules for different letters.

Noon group	Edhar	Noon + Scoon + Ain	ن + + ع	مِنْ عِندِ اللَّهِ
Meem group	Edhar	Noon + Scoon + Yaa'	ن + + ع	هُمْ يَحْزَنُونَ

Table 2 Quran pronunciation classify.

		Noon Sakenah	Meem Sakenah	Lam Sakenah
1	0	Edhar	Edhar	Edhar
2	8	Edgham	Edgham	
3	7	Ekhfa	Ekhfa	
4	6	Eklab		
Mudd (Prolong)				
5	_B	Mudd 2 tab		
	_D	Mudd 4 tab		
	_F	Mudd 6 tab		
Scoon				
6	p	Kalkala		

of discovered faults and error early through minimum test cases); (Karttunen, 1995; Kaplan and Kay, 1994; Johnson, 1997).

Extended Finite State Machine (EFSM) was used with the decision table method and look-up table in order to detect Quran reciting rules. EFSM is an enhancement technique for the FSM technique; it is applied before in computer science, reactive systems, philosophy, linguistics, logic and mathematics (Androutsopoulos et al., 2009). Huang (2001) declares that the EFSM technique is a powerful model for verification and test derivation; it is composed of states, transitions and actions. The FSM and EFSM both are sequential methods that move from state to state based at the transaction function outcome, and the main difference at the transaction function. The reason behind using EFSM is that FSM outputs have one bit binary code (true and false) which makes FSM problematic, where implementing a great number of state and rules are concerned, otherwise; the EFSM output is not Boolean function and could be represented by triggers condition rules, which is more suitable for implementation through this research where it has many rules and states that need to be handled. The state diagram that represents the reciting rule detection for the EFSM technique is defined as 6-tuple; $\langle I, O, S, F, U, T\rangle$. The transaction for the QEFSM model occurs through the following representation transmission (Eq. (1)):

$$(T : s \times F \times I \rightarrow S \times U \times O) \tag{1}$$

where T represents the transaction between the states; s represents the current state; F represents the enable function from the transaction; I represents the input for the state; S represents the next state; U represents the update for function and O represents the output for each transaction. The proposed

QEFSM model in Fig. 3 shows twenty seven states distributed at three levels of states.

The QEFSM engine checks four letters at each cycle, and it is shifting one letter until the end of the verses. The initial state for the model is S0; the window is working along the word letter by letter until a match rule is found and fired in the related decision table. For each state in QEFSM, there are transaction functions that control the movements between the states. Table 3 declares an explanation for all model transaction functions.

As shown in Fig. 3 and Table 3; the QEFSM model has eight output states that represent different Quran reciting rules.

The first six states (main stats) are managed by using the look-up table (Index table) that is responsible for executing the related decision table. Fig. 4 demonstrates the integration processes between the EFSM technique, decision table and look-up table (see Table 4) in the QEFSM model.

Table 3 Transaction function description.

Transaction function	Description
0	No reciting rules
1	Noon and Scoon
2	Kalkala letters and Scoon
3	Meem and Scoon
4	Lam and Scoon
5	Mudd letters
6	Mudd for special words
7	Eklab (Noon Sakenah) letters
8	Edhar (Noon Sakenah) letters
9	Edgham (Noon Sakenah) letters
10	Ekhfa' (Noon Sakenah) letters
11	Edhar (Meem Sakenah) letters
12	Ekhfa' (Meem Sakenah) letters
13	Edgham(Meem Sakenah) letters
14	Edhar (Lam Sakenah) letters
15	Edgham (Lam Sakenah) letters
16	Mudd 2 tab rules
17	Mudd 6 tab rules
18	Mudd 4 tab rules
19	Eklab reciting rules (Output)
20	Edhar reciting rules (Output)
21	Ekhfa' reciting rules (Output)
22	Kalkala reciting rules (Output)
23	Edgham reciting rules (Output)
24	Mudd 2 tab reciting rules (Output)
25	Mudd 4 tab reciting rules (Output)
26	Mudd 6 tab reciting rules (Output)
27	Mudd for special words (Output)

5. Evaluate the inner structure of QEFSM model

White box testing (Structure testing) is a verification technique that examines the code working and the internal mechanism of the system or component (Beizer, 1995). Through a structure test, several techniques and strategies were developed (as path test, branch testing, and data flow testing) to pick the possible test cases for the software structure, to ensure that enough tests are done in the testing process. In Hunt (2002); the main advantages in determining the test cases set are: to eliminate redundant testing, to provide appropriate test coverage, to give more effective testing and make limited testing. Among the white box testing methods, the path testing is considered as the best capacity method for detecting the errors through the unit testing phase, (Beiter, 1999; Frankl and Weyuker, 1995; Zhu et al., 1997).

The basis path testing is one of the testing strategies that are proposed by McCabe (1982) to test each linearly independent path through the software, as the number of the independent paths represents the number of test cases that should be tested for the software and the Cyclomatic complexity of the software, (Guangmmei et al., 2005). In Watson and McCabe (1996) there is a declaration about the strong connection between complexity and testing. Additionally, the structured testing methodology makes this connection explicit. Four steps should be done to satisfy the basis path testing strategy: which are

Step I: Drawing the program graph (Control flow graph) for the model.

The control flow graph is drawing in the first step for describing the logic structure of the software, where there are

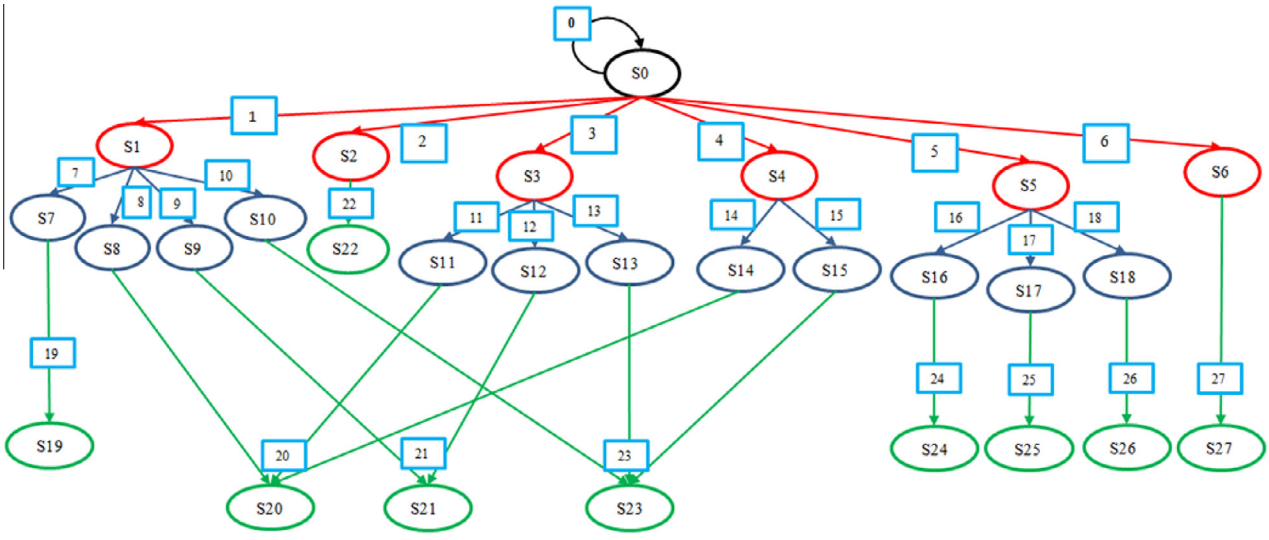

Figure 3 QEFSM model diagram.

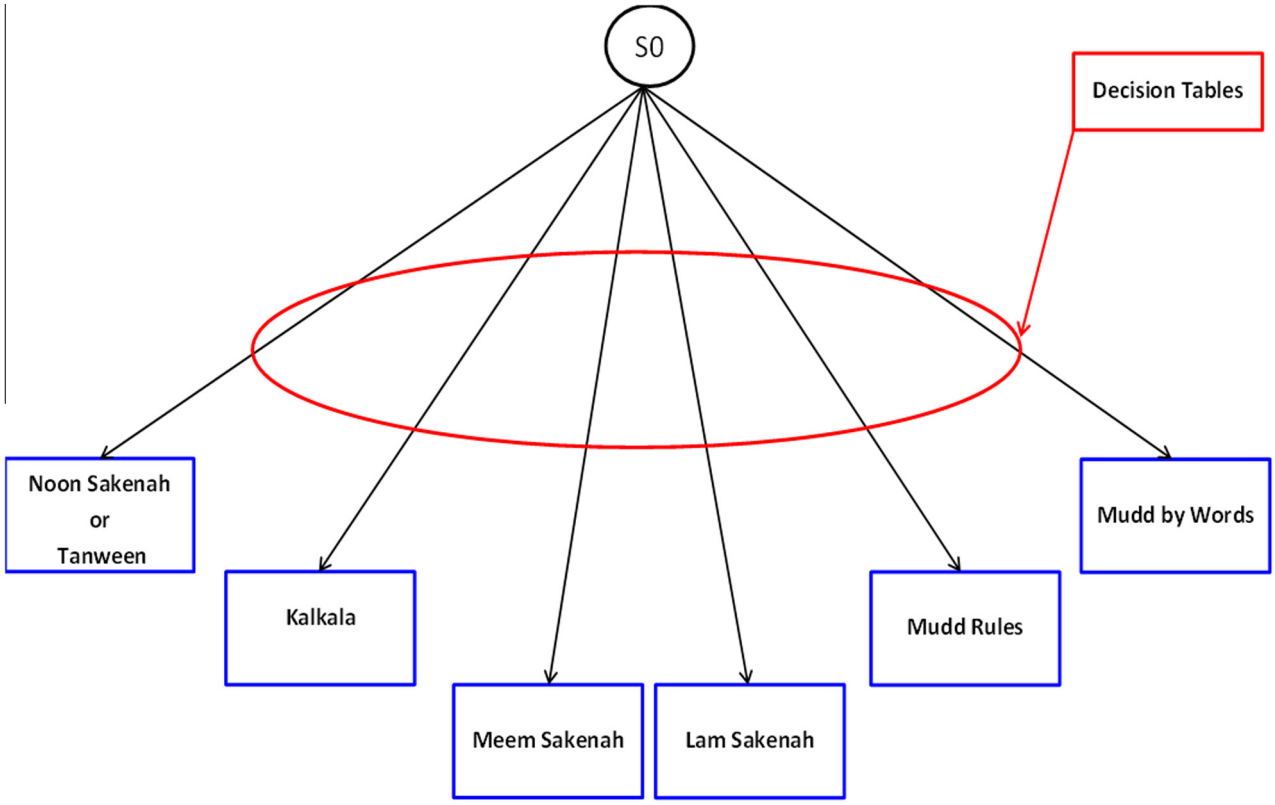

Figure 4 QEFSM integration diagram.

Table 4 Lookup table.

Description	Input Class			States	Decision Table
	1st	2nd	3rd		
نْ	Noon	Scoon		S1	D.T.1
ًor ٍor ٌ	Tanween			S1	D.T.1
قْ	Gaf	Scoon		S2	D.T.2
طْ	Ta'	Scoon		S2	D.T.2
بْ	Ba	Scoon		S2	D.T.2
جْ	jeem	Scoon		S2	D.T.2
دْ	Dal	Scoon		S2	D.T.2
مْ	Meem	Scoon		S3	D.T.3
لْ	Lam	Scoon		S4	D.T.4
يْ+ِ	Kasrah diacritic	ya'	Scoon	S5	D.T.5
وْ+ُ	Dumah diacritic	Wow	Scoon	S5	D.T.5
اْ+َ	Futt'ha diacritic	Alf	Scoon	S5	D.T.5
Mudd words				S6	D.T.6

lines (edge), circles (nodes), Areas boundary (region) and predicate node (conditional node). Fig. 3 shows the source code as a control flow graph for the QEFSM model. Through the previous flow graph, the state, edge, region and Predicted node numbers are determined in order to be the inputs for the Cyclomatic complexity (Cc) equation in Step 2 below.

Step II: Calculating Cyclomatic complexity.

Cyclomatic complexity (Cc) determines the basis paths (all the unique paths in the software without any iterations), and generates the test cases to execute each path at least once. Calculating the Cc used in the white box testing is usually done to specify the minimum number of tests and to measure the number of the independent paths through the unit. Cc is calculated by subtracting the nodes number from the edges number and adding two times the number of graph parts. Eq. (2) establishes that Cc is calculated from the flow graph of the QEFSM model. n is the number of nodes, e is the number of edges and p is the number of unconnected parts of the graph.

As declared in Fig. 3, let $e = 41$, $n = 28$ and $p = 1$ then

$$\begin{aligned} Cc &= e - n + 2(p); \\ Cc &= 41 - 28 + 2(1); \\ Cc &= 13 + 2; \\ Cc &= 15. \end{aligned} \tag{2}$$

The number of the independent paths for the QEFSM model is equal to Cc that is calculated from Eq. (2) above which is fifteen. Also Cc is calculated from the Predicted node (p) number, as in Eq. (3) below.

$$\begin{aligned} Cc &= p + 1; \\ Cc &= 14 + 1; \\ Cc &= 15. \end{aligned} \tag{3}$$

Since the independent paths for the QEFSM Model equal the value of Cc which is fifteen (15) paths, then; QEFSM have fifteen independent paths from the state (0) till the output states.

Step III: Determining basis paths for model graph

The finding of Cc for the software gives the developer an attention about the appropriate level of testing that should be done for the model. Table 6 below represents sample of the independent paths for the QEFSM model.

Step IV: Executing test cases for each independent path

To further evaluate the inner structure of the QEFSM model, test case sets were tested for all the independent paths of the model, where the testing processes for the test cases determined the number of the errors and defects that the model had. Each independent path in the model should have at least one test case to ascertain the structure work of the model for each output. The test cases that were done for the QEFSM totaled fifteen test cases.

Table 5 reveals that sample of test cases comprises of: test case number, independent path, test case number, input, expected output.

Table 5 Independent paths for QEFSM model.

	Path output
0-1-7-19	Eklab (Noon Sakenah)
0-1-8-20	Edhar (Noon Sakenah)
0-1-9-21	Ekfa' (Noon Sakenah)
0-1-10-23	Edgham (Noon Sakenah)
0-2-22	Kalkala (scoon)
0-3-11-20	Eklab (Meem Sakenah)
0-3-12-21	Ekfa' (Meem Sakenah)
0-3-13-23	Edgham (Meem Sakenah)
0-4-14-20	Edhar (Lam Sakenah)
0-4-15-23	Edgham (Lam Sakenah)
0-5-16-21	Mudd 2 tab
0-5-17-21	Mudd 4 tab
0-5-18-21	Mudd 6 tab
0-6-27	Mudd special words
0	No reciting rules found

Table 6 Sample of test cases.

Test Cases	Independent Paths	Input	Expected Output
1	S0-S1-S7-S19	مِنْ بعد	Eklab (Noon Sakenah)
2	S0-S1-S8-S20	مِنْ حميم	Edhar (Noon Sakenah)
3	S0-S1-S9-S21	صواباً ذلك	Ekhfa' (Noon Sakenah)
4	S0-S1-S10-S23	مِنْ يخشاها	Edgham (Noon Sakenah)
5	S0-S2-S22	اللهُ أحدٌ	Kalkala (Scoon)
6	S0-S3-S11-S20	همْ يحزنون	Edhar (Meem Sakenah)
7	S0-S3-S12-S21	ان ربهمْ بهم	Ekhfa' (Meem Sakenah)
8	S0-S3-S13-S23	وآتاكمْ من كل	Edgham (Meem Sakenah)
9	S0-S4-S14-S20	قلْ أعوذ	Edhar (LamSakenah)
10	S0-S4-S15-S23	الرحمن الرحيم	Edgham (Lam Sakenah)

6. Transliteration algorithm

In order to translate the proposed Quran reciting rules and Quran scripts into Braille code, Markov Algorithm (MA) was involved with the QEFSM model in order to introduce accurate Quran Braille translation. MA is a string rewriting system that transforms a set of strings (or symbols) to another set of strings (or symbols) according to specific rewriting rules, (Manning and Schütze, 1999). MA is a first order algorithm which means the first applicable production rule must be used and the rightmost/leftmost of the sub-string must be replaced. MA is used as a substitution system for the detected reciting rule into Latin script and Arabic scripts into Latin script, then it transforms the Latin script to a set of Braille symbols. In this work, three alphabets are involved:

1- The alphabet of the Arabic letters and diacritics.
$\Sigma 1 = \{..., \ً, \ٌ, \ٍ \ldots أ, ب, ت, ث\}$.
2- The alphabet of the Latin letters, Arabic numerals, punctuation marks and other special symbols.
$\Sigma 2 = \{A, B, C, D \ldots 1, 2, 3, 4 \ldots @, :, \$, \ldots\}$.
3- The alphabet of Braille symbols, (consisting of 63 symbols).
$\Sigma 3 =$ {⠁, ⠃, ⠉, … , ⠿}

As shown below in Eq. (4), the rules are a sequence of two strings as a one to one transformation. Let P be an element over $\Sigma 1$ and R an element over $\Sigma 2$.

$$T = (P \rightarrow R), \tag{4}$$

where T is Transformation, P is Pattern and R is Replacement. Table 7 shows example of the transformation rule set that is applied at the entry text.

In order to apply MA at the QEFSM model to translate Quran reciting rules and scripts to Braille code; in Orallo (2010); five steps should be taken into consideration, when the MA is applied:

- Check the rules in order, from top to bottom to see whether any of the patterns in the left of the arrow can be found in the symbol string.
- The algorithm stops executing when no string is found.
- If one or more is found, we replace the rightmost matching text in the input string with its replacement at the right of the arrow at the first corresponding rule.

Table 7 Transformation rule set.

∑1	→	∑2
ا	→	a
ب	→	b
ت	→	t
ث	→	?
ج	→	j
ح	→	:
خ	→	x
د	→	d
ذ	→	!
ر	→	r
ز	→	z
س	→	s
ش	→	%
ص	→	&
ض	→	$
ط	→	)
ظ	→	=
ع	→	(
غ	→	<
ف	→	f
ق	→	q
ك	→	k
ل	→	l
م	→	m
ن	→	n
ه	→	h
و	→	w
ي	→	i
ى	→	o
ة	→	*
لا	→	v
آ	→	/
أ	→	.
إ	→	>
ء	→	‘
ؤ	→	|
ئ	→	y
ً	→	1
ٌ	→	2
ٍ	→	u
َ	→	5
ُ	→	e
ِ	→	9
ّ	→	3
ْ	→	,

- If the applied rule is a terminating one, the algorithm executing is stopped.
- Return to Step one and proceed.

In order to gain more understanding about the integration processes between the QEFSM model and Markov Algorithm, see Fig. 5 below. The graph is going through five (5) main steps: that are started by segmenting the verses into characters and then saving them in two arrays; reversing the saved text at both arrays; replacing the Quran reciting rules that were detected by using the QEFSM Model at one of both arrays by matching roman characters and replacing the rest of the characters by Space; Transforming process for Arabic scripts into roman scripts; replacing the roman characters with compatible Braille code; and finally arranging the two arrays in double parallel Braille lines, Quran reciting rule lines above the characters and diacritics lines.

7. Experiments & measurements

7.1. Experimental data

There are many ways to import Quran verses to QBT, such as typing, copy, and as a text file. Five hundred (500) random verses were translated by the QBT. Table 8 shows a sample of experiments that have been done by using the QBT.

7.2. Experimental procedure

Fig. 6 shows the inputs and the outputs for Edgham reciting rules after it was translated by the QBT.

7.3. Measurements and testing processes

Fenton and Pfleeger (1998), as strong supporters of the need for the measurement in software development, state this: “You cannot control what you cannot measure”, in other words, we must control our projects, not just running them. Quran as a Holy book must be flawless. Three types of measurements were done for the experimented Quran verses.

- Measuring the accuracy for Quran reciting rules detecting process.

One hundred Quran verses were measured. The accuracy for the Quran reciting rules detected from the Quran verses was calculated by dividing the number of the detected reciting rules by QBT with the number of the whole reciting rules in the Quran text entered. See Eq. (5). Let M (RA) be abbreviated for Reciting Accuracy, QBT (Q.R.R) abbreviated for Quran reciting rules that are detected per verse by QBT and Q(Q.R.R) is abbreviated for Quran reciting rules that exist per verse in the original Quran text.

Suppose z = M (R.A), x = QBT (Q.R.R) and y = Q (Q.R.R) then:

$$\mathbf{z} = \mathbf{x}/\mathbf{y} \times 100\%. \tag{5}$$

The results for the measurements process were successful and one hundred percent (100%) detection accuracy was obtained.

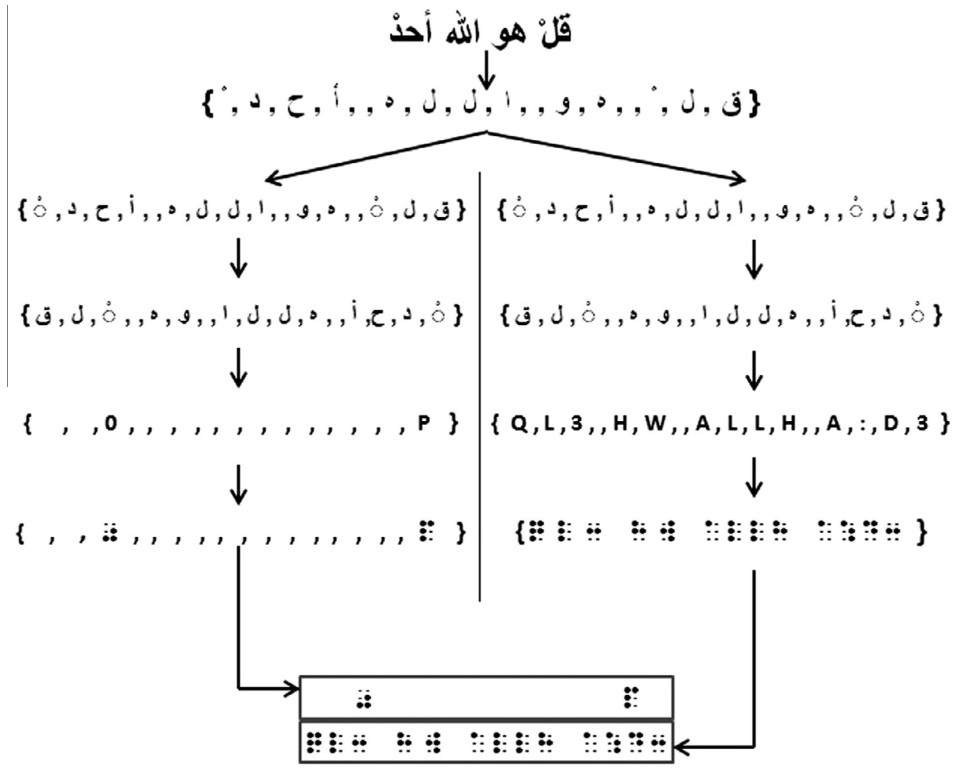

Figure 5 Markov Algorithm transformation flow graph.

Table 8 Sample of QBT experiments.

Braille translating verses Including the reciting rules	Quran verses
[illegible]	الرَّحْمَنِ الرَّحِيمِ
[illegible]	أَتُمِدّونَنِيمَالٍ
[illegible]	وُجُوهٌ يَوْمَئِذٍ
[illegible]	عَامِلَةٌ نَاصِبَةٌ
[illegible]	وَمَا هُم بِخَارِجِينَ

- Measuring the accuracy for Quran scripts translating process.

The accuracy for the translation process was calculated by dividing the number of the translated characters by the QBT at the number of characters for the entry text. See Eq. (6).

TA is an abbreviation for Translating Accuracy. QBT (T.L.V) is abbreviated for the number of letters and diacritics that are translated for each Quran verse using QBT, Q (Q.R.R) abbreviated for Quran letters and diacritics that exist per each verse in the entered Quran text.

Suppose; v = M (R.A), u = QBT (Q.R.R) and w = Q (Q.R.R) then:

$$\boldsymbol{v} = \boldsymbol{u}/\boldsymbol{w} \times 100\%. \quad (6)$$

The results for the measurements process had succeeded and one hundred percent (100%) translating accuracy obtained.

- Measure the accuracy of QBT results at the real life by users.

QBT was tested in Jordan at the Alsoula Blind institute. The accuracy in detecting Quran reciting rules from the Quran verses by users was calculated by using Eq. (7). Thirty users were tested in 100 experiments for three times.

R.A is abbreviated for User Reciting Accuracy. QBT (U.Q.R.R) is abbreviated for Quran reciting rules where the user successfully detects and recites each verse after the translation by using the Quran Braille Translator. Q (Q.R.R) is abbreviated for Quran reciting rules that exist for each verse in the original Quran text.

Suppose; a = M (R.A), b = QBT (Q.R.R) and c = Q (Q.R.R) then:

$$\boldsymbol{a} = \boldsymbol{b}/\boldsymbol{c} \times 100\%. \quad (7)$$

The best result on the experiments achieved 81% accuracy which was scored by user number one (1) & eight (8) who were Braille teachers. The worst result was sixty one (61%) by a student. Table 9 below declares the process of measuring the accuracy of detecting Quran reciting rules by user number one (1).

As shown in Table 8, three tests were done for the user. Each test (Edhare, Edgham, Ekhfa, Eklab and Mudd) was checked for accuracy in detecting Quran reciting rules respectively which are found in one hundred (100) Quran verses that

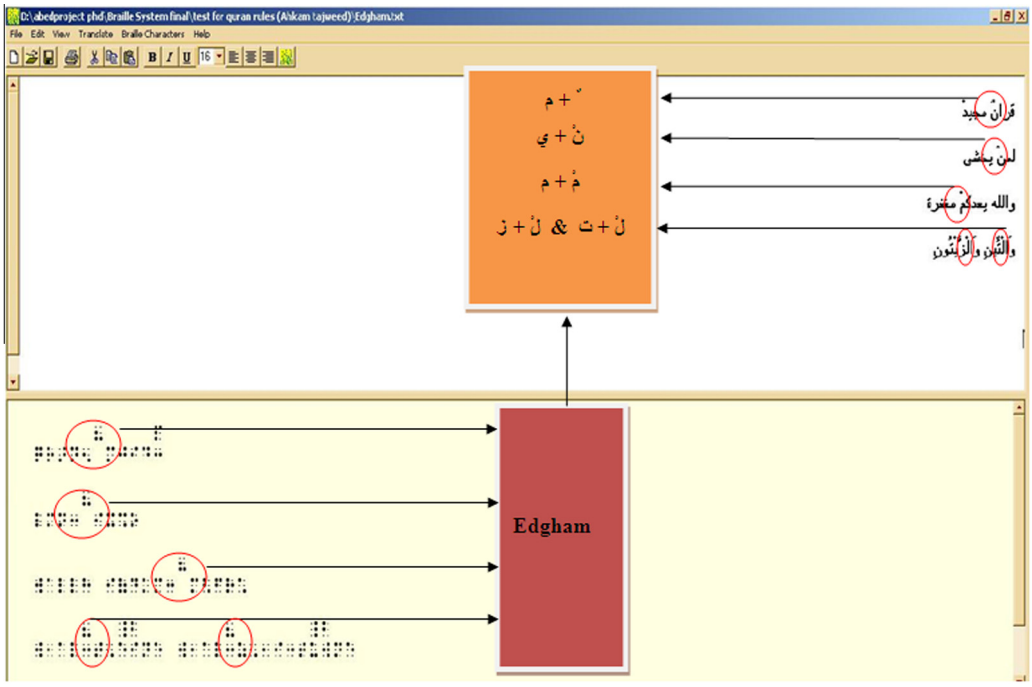

Figure 6 Output screen for Edgham.

Table 9 User 1-tests results for the experiments.

User 1	Edhar (Adhere)	Edgham (Diphthong)	Ekhfa' (Concealing)	Eklab (Labial Nasalization)	Kalkala	Mudd (Prolong)	Sum	Reciting Accuracy
	70	34	22	5	24	90	245	100%
Test 1	51	26	17	3	18	62	177	0.72
Test 2	58	29	19	4	21	68	199	0.81
Test 3	66	32	20	4	22	76	220	0.90

were chosen randomly. The existing Quran reciting rules in the one hundred (100) verses are seventeen (70) for Edhar, thirty four (34) for Edgham and so on. The user was able to detect fifty one (51) Edhars in the first test, fifty eight (58) Edhars in the second test and sixty six (66) Edhars in the third test. The M (RA) result graph for the first user is given in Fig. 7.

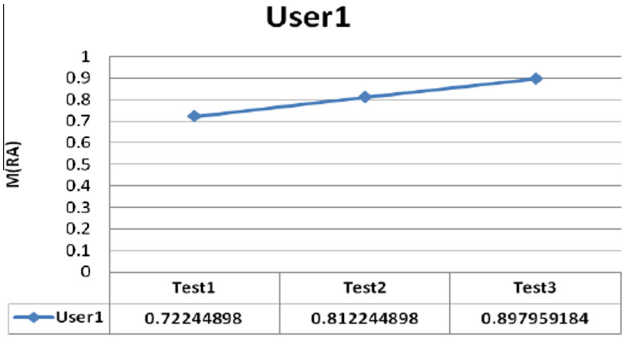

Figure 7 Reciting rules' measurement graph for the first user.

As proven from the graph, the M (RA) values had increased beginning from test 1 to test 3, which declares the continuous improvement for the user reciting and detecting ability through the repetition of tests, where through the first test, users are not familiar yet with the new Braille code as compared to how they fared in tests 2 and 3.

8. Conclusions

The Quran Extended Finite State Machine (QEFSM) model was developed in order to detect all the reciting rules that are found in the Quran text. Basis path testing was used as white box testing to evaluate the QEFSM model. The number of the independent path for the model equals the number of the test cases that are done for the QEFSM model, thus reducing the test effort by discovering the errors and faults by doing fewer test cases. The look-up table was controlled and it indexed the state in the QEFSM model, and each state in the EFSM technique was represented by the decision table that managed the transaction functions belonging to it. Markov Algorithm was used in this work for translating Quran reciting rules, and script to Braille code. Markov Algorithm is used in

this research for the translating process because it is an ordered rule algorithm that could be used in future studies for other researches to translate the Braille code into Arabic or any other languages, just by inversing the rules and making some modifications. The outcome of these works is shown in the double lines of the Braille symbols; the first line is the proposed Quran reciting rules and the second line is for the Quran scripts. Holy Quran is a book of God which has specific page layout (as: the name of Surah, number of Juzu', special signs and symbols, and page numbers) and as the next step, one should consider to prepare a specific systematic Quran text that includes all the extended details to be handled and translated.

References

Abualkishik, A.M., Omar, K., 2009a. Quran vibrations in Braille code, ICEEI'09. In: International Conference on Electrical Engineering and Informatics, 1. IEEE, pp. 12–17. August 2009.

Abualkishik, J.A.M., Omar, K., 2009b. Quranic Braille System. International Journal of Humanities and Social Sciences, 313–319, Chicago.

Al-kari', A., 1998. Quran Reciting Rules-Haffes Style. Islamic University, KSA, vol. 1(1), pp. 24–44.

Androutsopoulos, K., Clark, D., Harman, M., Li, Z., Tratt, L., 2009. Control dependence for extended finite state machines. In: Fundamental Approaches to Software Engineering (FASE '09). Springer LNCS, York, UK, pp. 216–230.

Babker, A., 1983. Algra'at for Ibn Jareer Altabari (PhD thesis), Umm Algura University, Kingdom Saudi Arabia.

Beesley, K., Karttunen, L., 2002. Finite state morphology: xerox tools and techniques. Book. Stanford. Produced by Pstil. CSLI Publications, pp. 5–30.

Beiter, B., 1999. Software System Testing Book. Van Nostrand Reinhold Company, New York.

Beizer, B., 1995. Black Box Testing: Techniques for Functional Testing of Software and Systems. John Wiley & Sons Inc, New York, pp. 15–40, ISBN-10/ASIN 0471120944.

Blenkhorn, P., 1997. A system for converting print into Braille. IEEE Trans. Rehabil. Eng. 5 (2), 121–129.

Fenton, Norman E., Pfleeger, Shari Lawrence, 1998. Software Metrics: A Rigorous and Practical Approach. PWS Publishing Co..

Frankl, P., Weyuker, E., 1995. An applicable family of data flow testing criteria. IEEE Trans. Software Eng. 14 (10), 1483–1498.

Guangmmei, Z., Rui, C., Xiaowei, L., Congying, H., 2005. The automatic generation of basis set of path for path testing. In: Proceedings of the 14th Asian Test Symposium (ATS '05), pp. 1081–7735.

Huang, S., 2001. On speeding up extended finite state machines using catalyst circuitry. In: Proceedings Conference Design Automation. Yokohama, Japan, pp. 583–588.

Hunt, 2001. Basis path testing for structural and integration testing. In: Star East 2002 Conference. The Westfall Team.

Johnson, D., 1997. Formal Aspects of Phonological Description. Mouton. Hague. Walter de Gruyter. ISBN 978-90-279-2217-5.

Kaplan, M., Kay, M., 1994. Regular models of phonological rule systems. Comput. Linguist., 331–378

Karttunen, L., 1995. The replace operator. In: Proceeding of the Annual Meeting of the Association for Computational Linguistics, pp. 16–23.

King, A., 2000. Keio University Access Research Group, Keio University ARG website.

Malik, F., 2007. The Qur'an in English Translation Complete. MidEastWeb for Coexistence.

Manning, C., Schütze, H., 1999. Foundations of Statistical Natural Language Processing. MIT Press, Cambridge, MA.

McCabe, T., 1982. A Software Testing Methodology Using the Cyclomatic Complexity Metric. NIST Special Publication, Washington D.C., pp. 500–599.

Mellor, C. Michael, 2006. Louis Braille: A Touch of Genius. National Braille Press.

Orallo, J., 2010. A (hopefully) unbiased universal environment class for measuring intelligence of biological and artificial systems (Extended Version). In: Proceeding of the Third Conference on Artificial General Intelligence. Advances in Intelligent Systems Research. Atlantis Press, pp. 182–183.

Roy, W., 2000. Braille computer facilities for Kuwait special schools. The Kuwait Institute for Scientific Research. Printing system with Braille, User manual, Kuwait (In Arabic).

Sensus, 1999. Danish Braille system. Sensus ApS – Specialists accessibility.

Watson, A., McCabe, T., 1996. Structured Testing: A Testing Methodology Using the Cyclomatic Complexity Metric. National Institute of Standards and Technology, Gaithersburg.

WHO, 2014. Visual impairment and blindness. World Health Organization. <http://www.who.int/mediacentre/factsheets/fs282/en/>, August, 2014 [Retrieve in 11-Nov-2014].

Yamuna, M., Vora, Khyati R., 2013. Encryption using Braille alphabets and graph domination. Int. J. Comput. Sci. Appl., June 2013

Zhu, H., Hall, P., May, J., 1997. Software unit test coverage and adequacy. ACM Comput. Surv. 29 (4), 366–427.