IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 2 Issue 12, December 2015.
www.ijiset.com
ISSN 2348 – 7968

Acceptor Automata of Middle Hamzah Rules Using Pushdown Automata

Bayan Bagasi[1], Afnan Aqal [2] , Ola Al-Beshry [3], Ablah Al-Ameri [4]
Dr. Zineb Eid [5]
[1] Computer science, King Abdul-Aziz University,
Jeddah, Saudi Arabia , bbagasi@kau.edu.sa

[2] Computer science, King Abdul-Aziz University,
Jeddah, Saudi Arabia , AAgal0001@stu.kau.edu.sa

[3] Computer science, King Abdul-Aziz University,
Jeddah, Saudi Arabia , oalbeshri0001@stu.kau.edu.sa

[4] Computer science, King Abdul-Aziz University,
Jeddah, Saudi Arabia , asmalamri1@kau.edu.sa

[5] Computer science, King Abdul-Aziz University,
Jeddah, Saudi Arabia , zmahmoud@kau.edu.sa

Abstract
In Arabic language, the middle hamzah rules of writing are a complicated rule. Because, there are different ways with various cases, many peoples make a mistake. These rules depending on the Harkah of Hamzah and previous letter of hamzah to detect the way of writing middle hamzah. To achieve the correctly writing of word that has middle hamzah, we proposed the Acceptor automata. These automata concerns on middle hamzah rules. While middle hamzah rule include checking of the previous, we need the automata has storage. For this reason, we used pushdown automata to build the acceptor.
Keywords: *Pushdown automata, Context Free grammar, Middle Hamzah rules.*

1. Introduction

The Arabic language is the sixth of the most popular languages in the world. The wide arc stretching across the Middle East, North Africa, and the Horn of Africa are spoken Arabic languages.[7] The written language of Arabic divides into two types the modern written Modern Standard Arabic and Classical Arabic. However, Modern Standard Arabic is derived from the language of Classical Arabic which is language of Quran. Modern Standard is widely used in poplar's life such as schools, universities, government and the media. [8]
Otherwise, there are features of Modern Standard Arabic writing such as words are written from right to left, numbers are written from left to right, depending on where they are placed in a word (beginning, middle, end, or standalone), some Arabic letters will change form, the Arabic alphabet is derived from Aramaic script and contains 28 letters , and letters are always joined together in Arabic writing (both written and typed) except in very rare cases. [9]
Also, there are many rules of writing specific word. These rules help people to write word correctly. For example, Al Alshamsyia and Al Alqamryia rules, Almad rules, and Hamzah's rules. The middle Hamzah rule has a common mistake and complicated. In next section, we explain the rules and cases of middle hamzah

ISSN 2348 – 7968

1.1. Middle hamzah rules in Arabic writing :

There is four way of writing middle Hamzah depending on different cases:

On Nabreah (ـئـ)

- If the hamzha has Kasrah as Harkah
- If the previous letter has Kasrah as Harkah
- If the previous letter is Yah Saknah

On Alef (أ)

- If Hamzah has Fatha as Harkah and the previous letter has a Fatha too
- If Hamzah has Fatha as Harkah and the previous letter has a Sokoon
- If Hamzah has Sokoon as Harkah and the prvious letter has a Fatha

On Waw (ؤ)

- If Hamzah has Dammah as Harkah and the previous letter doesn't has Kasrah
- if Hamzh doesn't has Kasrah and the previous letter has Damah

On the line (ء)

- If Hamzah has Fatha as harkah and the previous letter is Allaf with Sokoon as Harkah
- If Hamzah ha Fatha or Damah and the previous letter is Wow with Sokoon as Harkah [10]

Table 1: Example of Middle Hamzah rules

Example	***Rules***	Case
سُئِل	On Naberah	If the hamzha has Kasrah as Harkah (سُئِل)
سَأَل	On Alf	If Hamzah has Fatha(سَأَل)as Harkah and the previous letter has a Fatha too(سَأَل)
سُؤَال	On Waw	if Hamzh doesn't has Kasrah (سُؤَال) and the previous letter has Damah(سُؤَال)
جاءَت	On line	If Hamzah has Fatha (جاءَت) as harkah and the previous letter is Allaf with Sokoon as Harkah (جاءَ ت)

The writer is must be remember this rule during writing, while the meaning and sound of word mostly change. To make theses easier for writer, it can be checking this rule during writer writes the sentence. Using Acceptor automata, can be check the word writing correctly way. While the rule of middle Hamzah depending on two factors Harkah of the Hamzah and the previous letter , we need a storage to store the Harkah of the previous letter.

Due to, we proposed build an Automata has a storage such as pushdown . In the next section , we present the background of context free grammar and pushdown automata that used on proposed work. Then, we will present the relate Then, we present proposed context free grammar and the pushdown automata for middle Hamzah rules. Finally, we will test automata .

2. Background

2.1. Context Free Grammar

To generate patterns of strings, used a set of recursive rewriting rules (or productions) called A context-free grammar (CFG). The component of defined CFG are :

- The characters of the alphabet that appear in the strings generated by the grammar called *terminal symbols*.
- The placeholders for patterns of terminal symbols that can be generated by it called *nonterminal symbols*
- Rules for replacing (or rewriting) nonterminal symbols (on the left side of the production) in a string with other nonterminal or terminal symbols (on the right side of the production) called *productions*.
- A special nonterminal symbol that appears in the initial string generated by the grammar called a *start symbol*.

To generate the string from CFG followed the following steps :

- Begin with a string which contain the start symbol
- Using the production has the start symbol on the left hand size, replacing the start symbol with the right hand side of the production
- Repeat the replacing of non-terminal with the right hand side of some corresponding production, until all non-terminals have been replaced by terminal symbols. [11]

2.2. Pushdown automata

Pushdown Automata (PDA) is type of automata that use a stack to allow unbounded storage. PDA is NFA with stack and it is equivalent to Context free grammar (CFG). Non-deterministic PDA can recognize all context free languages (CFLs) but deterministic version only recognize deterministic context free languages.

A PDA is described by finite set of internal states Q, finite set of input alphabet Σ, finite set of stack alphabet Γ, transition function δ where δ: Q x (Σ U {λ}) x $\Gamma \rightarrow$ finite subsets of Q x Γ*, start/initial state q0 $\in$ Q, stack start

ISSN 2348 – 7968

symbol Z ∈ Γ, and finally a set of final/accepting states F ⊆ Q .
Each move in PDA determined by the current state and current input symbol and the current symbol on the top of the stack then the result of the move is a new state and changing the top of the stack by push and pop operations. [12]

3. Related work

There are many researchers that produced different approach for Arabic or English parsing, some of these are listed below:
McCord build parsing system using bottom up chart parser .[1] Tounsi produced a parser based on Treebank-based parsers and automatic LFG f-structure annotation methodologies. [2] Shihadeh produced a simple parser to check the syntax of an Arabic sentences from the grammatical point view, they construct a new efficient Context-Free Grammar.[3] Also a framework called A'reb produced by Essam to automate the parsing of Arabic language sentence, this system is divide into two phases : lexical analysis which find words original and roots, then separate the words from its prefixes and suffixes, and assign the separated words to special tokens, and syntax analysis which receive all the tokens then use context free grammar to find the best grammar for the tokens. [4] Madiha design recognizer for English Language by using Nondeterministic Pushdown Automata [5].
This field is very rich, more complicated because it's based on natural language processing (NLP), and need more research to get accurate result especially in Arabic language.

4. Proposed work :

4.1. Context Free Grammar:

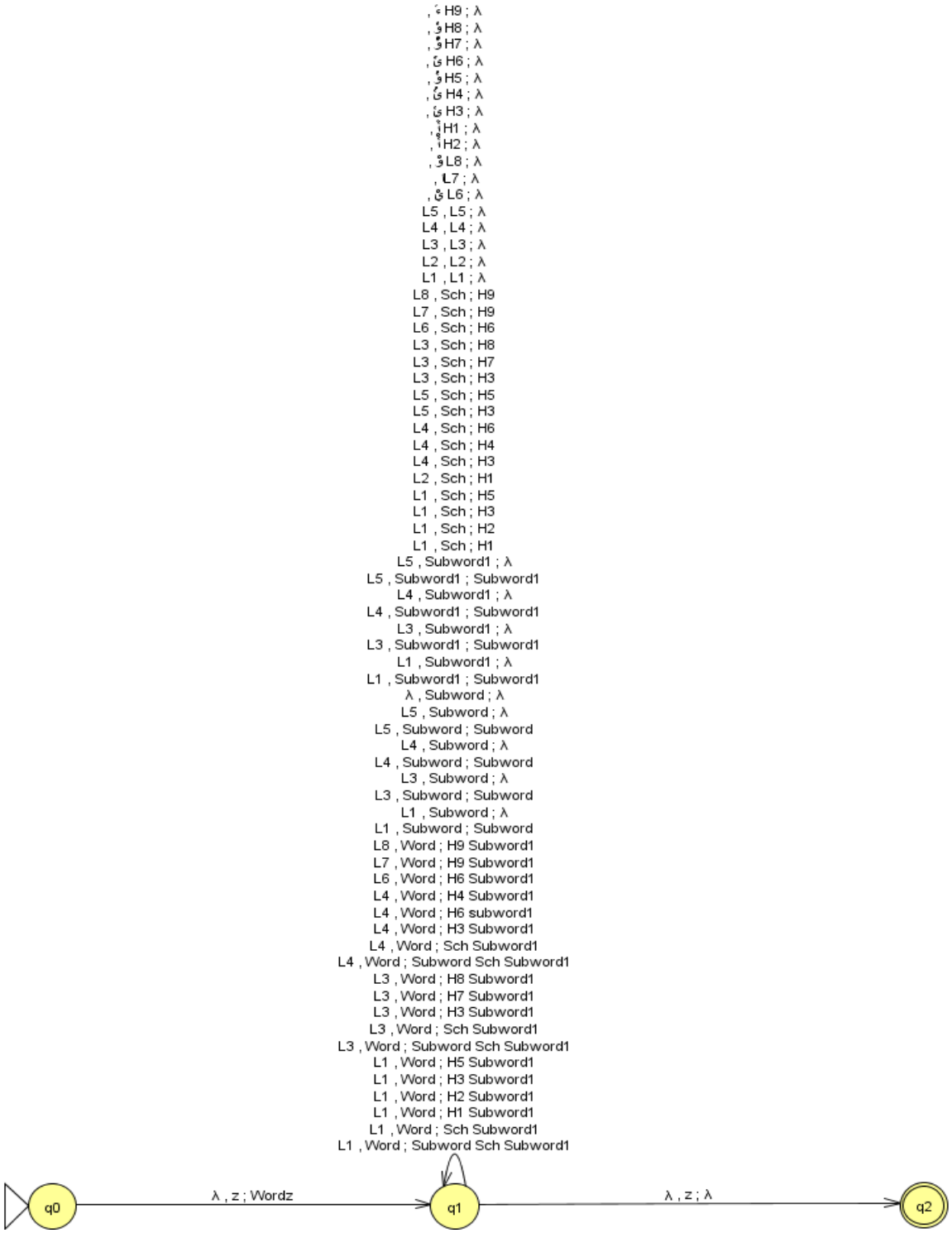
The grammar should be into Greibach normal form:
Word → L1 Subword Sch Subword1| L3 Subword Sch Subword1| L4 Subword Sch Subword1| L1 Sch Subword1| L3 Sch Subword1| L4 Sch Subword1|L1 H1 Subword1|L1 H2 Subword1| L1 H3 Subword1| L3 H3 Subword1|L4 H3 Subword1| L4 H6 Subword1| L4 H4 Subword1| L6 H6 Subword1 | L1 H5 Subword1| L3 H7 Subword1| L3 H8 Subword1| L7 H9 Subword1| L8 H9 Subword1
Subword → L1 Subword |L3 Subword | L4 Subword | L5 Subword | L1|L3|L4|L5| λ
Subword1 → L1 Subword1|L3 Subword1 |L4 Subword1 |L5 Subword1 | L1|L3|L4|L5
Sch → L1 H1|L1 H2 |L2 H1 | L1 H3|L3 H3 |L4 H3|L5 H3|L4 H6 |L4 H4 | L6 H6| L1 H5 |L5 H5|L3 H7 |L3 H8 | L7 H9| L8 H9
L1→اَ |بَ|يَ...
L2→ اْ|بْ|وْ.....
L3 →اُ|بُ|يُ.....
L4 → اِ|بِ|يِ.....
L5→اْ|بْ|يْ |.....
L6 → يْ
L7→ا
L8→وْ
H1→أ
H2→ اْ
H3→ئ
H4→ئُ
H5→ؤُ
H6→ئَ
H7→ؤْ
H8→ؤَ
H9→ء

4.2. Pushdown Automata for Context Free languages

Each move in PDA determined by the current state and current input symbol and the current symbol on the top of the stack then the result of the move is a new state and changing the top of the stack by push and pop operations [6].
To construct a PDA that accepts language that generated by Context Free grammar in previous section. First, transform the grammar into Greibach normal form as you see in previous section. Then, the automaton will have three states {q0, q1, q2}, initial state q0 and final state q2 and the start symbol of grammar (Word) is putting on the stack by following transition:
δ(q0, λ, z) = {(q1 , Word z)}.
Then, represent other production in our grammar by following transitions:
δ(q1, L1, Word) = {(q1, Subword Sch Subword1), (q1, Sch Subword1), (q1, H1 Subword1), (q1, H2 Subword1), (q1, H3 Subword1), (q1, H5 Subword1)}
δ(q1, L3, Word) = {(q1, Subword Sch Subword1), (q1, Sch Subword1), (q1, H3 Subword1), (q1, H7 Subword1), (q1, H8 Subword1)}
δ(q1, L4, Word) = {(q1, Subword Sch Subword1), (q1, Sch Subword1), (q1, H3 Subword1), (q1, H6 Subword1), (q1, H4 Subword1)}
δ(q1, L6, Word) = {(q1 , H6 Subword1)}
δ(q1, L7, Word) = {(q1 , H9 Subword1)}
δ(q1, L8, Word) = {(q1 , H9 Subword1)}

ISSN 2348 – 7968

$\delta(q1, L1, Subword\ ) = \{(q1,\ Subword), (q1,\ \lambda)\}$



$\delta(q1, L3, Subword\ ) = \{(q1,\ \ Subword), (q1,\ \lambda)\}$
$\delta(q1, L4, Subword\ ) = \{(q1,\ Subword), (q1,\ \lambda)\}$
$\delta(q1, L5, Subword\ ) = \{(q1,\ Subword), (q1,\ \lambda)\}$

Fig. 1 Transition graph of pushdown automaton for Arabic wordscontain middle hamzah

ISSN 2348 – 7968

δ(q1, λ, Subword) = {(q1 , λ)}
δ(q1, L1, Subword1) = {(q1, Subword1), (q1, λ)}
δ(q1, L3, Subword1) = {(q1, Subword1), (q1, λ)}
δ(q1, L4, Subword1) = {(q1, Subword1), (q1, λ)}
δ(q1, L5, Subword1) = {(q1, Subword1), (q1, λ)}
δ(q1, L1, Sch) = {(q1, H1), (q1, H2), (q1, H3), (q1, H5)}
δ(q1, L2, Sch) = {(q1 , H1)}
δ(q1, L4, Sch) = {(q1, H3), (q1, H6), (q1, H4)}
δ(q1, L5, Sch) = {(q1, H3), (q1, H5)}
δ(q1, L3, Sch) = {(q1 , H3),(q1 , H7), (q1 , H8)}
δ(q1, L6, Sch) = {(q1 , H6)}
δ(q1, L7, Sch) = {(q1 , H9)}
δ(q1, L8, Sch) = {(q1 , H9)}
δ(q1, L1, L1) = {(q1 , λ)}
δ(q1, L2, L2) = {(q1 , λ)}
δ(q1, L3, L3) = {(q1 , λ)}
δ(q1, L4, L4) = {(q1 , λ)}
δ(q1, L5, L5) = {(q1 , λ)}
δ(q1,ْي , L6) = {(q1 , λ)}
δ(q1,ا , L7) = {(q1 , λ)}
δ(q1,ْو , L8) = {(q1 , λ)}
δ(q1,أَ , H1) = {(q1 , λ)}
δ(q1,أُ , H2) = {(q1 , λ)}
δ(q1,ئِ , H3) = {(q1 , λ)}
δ(q1,ئُ , H4) = {(q1 , λ)}
δ(q1,وُ , H5) = {(q1 , λ)}
δ(q1,ئَ , H6) = {(q1 , λ)}
δ(q1,وْ , H7) = {(q1 , λ)}
δ(q1,وَ , H8) = {(q1 , λ)}
δ(q1,ءَ , H9) = {(q1 , λ)}

The appearance of the stack start symbol on the top (z), the completion of the derivation and the PDA put into its final state by following transition:

δ(q1, λ, z) = {(q2 ,λ)}

Now we can draw the automaton by use the previous transition in figure 1.

5. Testing PDA :

Word	Result
سأل	δ(q0, λ, z) =(q1 , Word z). δ(q1, L1, Word) = (q1, H1 Subword1). δ(q1,أَ , H1) = (q1 , λ). δ(q1, L1, Subword1) = (q1, λ). δ(q1, λ, z) = (q2 , λ).
سؤال	δ(q0, λ, z) = (q1 , Word z). δ(q1, L3, Word) = (q1, H5 Subword1). δ(q1,وُ , H5) = (q1 , λ). δ(q1, L5, Subword1) = (q1, Subword1). δ(q1, L3, Subword1) = (q1, λ). δ(q1, λ, z) = (q2 , λ).
مسؤول	δ(q0, λ, z) = (q1 , Word z}. δ(q1, L1, Word) = (q1, Sch Subword1). δ(q1, L5, Sch) = (q1, H5). δ(q1,وُ , H5) = (q1 , λ). δ(q1, L5, Subword1) = (q1, Subword1). δ(q1, L3, Subword1) = (q1, λ). δ(q1, λ, z) = (q2 , λ).
مسألة	δ(q0, λ, z) = (q1 , Word z). δ(q1, L1, Word) = (q1, Sch Subword1). δ(q1, L2, Sch) = (q1 , H1). δ(q1,أَ , H1) = (q1 , λ). δ(q1, L1, Subword1) = (q1, Subword1). δ(q1, L3, Subword1) = (q1, λ). δ(q1, λ, z) = (q2 , λ).
سئل	δ(q0, λ, z) = (q1 , Word z). δ(q1, L3, Word) = (q1, H3 Subword1). δ(q1,ئِ , H3) = (q1 , λ). δ(q1, L1, Subword1) =(q1 , λ). δ(q1, λ, z) = (q2 , λ).
فأس	δ(q0, λ, z) =(q1 , Word z). δ(q1, L1, Word) = (q1, H2 Subword1). δ(q1,أُ , H2) = (q1 , λ). δ(q1, L3, Subword1) = (q1, λ). δ(q1, λ, z) = (q2 , λ).
طائر	δ(q0, λ, z) =(q1 , Word z). δ(q1, L1, Word) = (q1, Sch Subword1). δ(q1, L5, Sch) = (q1, H3). δ(q1,ئِ , H3) = (q1 , λ). δ(q1, L3, Subword1) = (q1, λ). δ(q1, λ, z) = (q2 , λ).
ظمئت	δ(q0, λ, z) =(q1 , Word z). δ(q1, L1, Word) = (q1, Sch Subword1).

ISSN 2348 – 7968

	δ(q1, L1, Sch) = (q1, H3). δ(q1,ئ , H3) = (q1 , λ). δ(q1, L5, Subword1) = (q1, λ). δ(q1, λ, z) = (q2 , λ).
تفاءل	δ(q0, λ, z) =(q1 , Word z). δ(q1, L1, Word) = (q1, Subword Sch Subword1). δ(q1, L1, Subword) = (q1 , λ). δ(q1, L7, Sch) = (q1 , H9) δ(q1, ء , H9) = (q1 , λ). δ(q1, L1, Subword1) = (q1, λ). δ(q1, λ, z) = (q2 , λ).
ضوءها	δ(q0, λ, z) =(q1 , Word z). δ(q1, L1, Word) = (q1, Sch Subword1). δ(q1, L8, Sch) = (q1 , H9). δ(q1, ء , H9) = (q1 , λ). δ(q1, L1, Subword1) = (q1, Subword1). δ(q1, L5, Subword1) = (q1, λ). δ(q1, λ, z) = (q2 , λ).

6. Conclusions

In this paper, using pushdown automata, we build acceptor automata for Arabic word which has a middle hamzah. Checking of common mistakes occur of word has a middle hamzha is the main goal of proposed automat. To build pushdown automat, we used a CFG in Greibach normal form. This Grammar covers all cases of rules by used the Alphabet of Arabic language with different harakah (ِ , ُ , َ) as a terminate symbols. To check the efficiency of automata , we test many cases of word by the automata manually. Finally , we will develop this automat to correct the mistake and cover all rules of Arabic writing.

References

1. *An Arabic Slot Grammar Parser.* **Michael C. McCord, Violetta Cavalli-Sforza.** s.l. : Workshop on Computational Approaches to Semitic Languages: Common Issues and Resources, 2007.
2. *PARSING ARABIC USING TREEBANK-BASED LFG RESOURCES.* **Lamia Tounsi, Mohammed Attia, Josef van Genabith.** s.l. : CSLI, 2009.
3. *Context-Free Grammar Analysis for Arabic Sentences.* **Shihadeh Alqrainy, Hasan Muaidi , Mahmud S. Alkoffash.** s.l. : International Journal of Computer Applications, 2012, Vol. 53.
4. *A Framework to Automate the Parsing of Arabic Language Sentences.* **Essam Al Daoud, Abdullah Basata.** 2, s.l. : The International Arab Journal of Information Technology, April 2009, Vol. 6.
5. *To Design a English Language Recognizer by using Nondeterministic Pushdown Automata.* **Madiha Khurram Pasha, M. Sadiq Ali Khan.** 1, November 2014 : International Journal of Computer Applications, Vol. 105.
6. **linz, Peter.** *An introduction to formal languages and automata.* s.l. : Jones & Bartlett Publishers, 2011. 978-1449615529.
7. The Arabic language. *AL-bab.* [Online] http://www.al-bab.com/arab/language/lang.htm.
8. Arabic. *Omniglot.* [Online] 1998-2015. http://www.omniglot.com/writing/arabic.htm.
9. The Arabic Writing System. *Globalization Partners International.* [Online] 2001-2015. [Cited:] http://www.globalizationpartners.com/resources/arabic-translation-quick-facts/the-arabic-writing-system.aspx.
10. **Jamil, Shariah Program graduate Mohtanick.** Ways in Which a Hamza can be Written. *Learning Arabic Online.* [Online] 2011-2013. [Cited:] http://www.learnarabiconline.com/writing-hamza.shtml.
11. Context Free Grammar. [Online] https://www.cs.rochester.edu/~nelson/courses/csc_173/grammars/cfg.html .
12. Ways in Which a Hamza can be Written. [Online] 2011-2013. http://www.learnarabiconline.com/writing-hamza.shtml.

Bayan T. Bagasi received B.S degree in Computer Science in 2012 from KAU. Currently, Teacher Assistant at KAU , and Master Student in Faculty of Computing and Information Technology, King Abdul-Aziz University Faculty of Computing and Information Technology, King Abdul-Aziz University

Ablah S. AlAmri received bachelor degree in computer science in 2012 from King Abdul-Aziz University. Currently, teaching assistant at King Abdul-Aziz University, and master student in Faculty of Computing and Information Technology, King Abdul-Aziz University

Ola A. Al--Beshry received bachelor degree in computer science in 2012 from King Abdul-Aziz University. Currently, master student in Faculty of Computing and Information Technology, King Abdul-Aziz University

Afnan T. Aqal received bachelor degree in computer science in 2012 from King Abdul-Aziz University. Currently, master student in Faculty of Computing and Information Technology, King Abdul-Aziz University