A Computational Morphology System for Arabic

Riyad Al-Shalabi
Martha Evens
Department of Computer Science and Applied Mathematics
Illinois Institute of Technology
10 West 31st Street
Chicago, IL 60616
alshriy@minna.cns.iit.edu, mwe@math.nwu.edu
312-567-5153

Abstract

This paper describes a new algorithm for morphological analysis of Arabic words, which has been tested on a corpus of 242 abstracts from the Saudi Arabian National Computer Conference. It runs an order of magnitude faster than other algorithms in the literature.

1. Introduction

This paper describes a computer system for Arabic morphology that employs a new, faster algorithm to find roots and patterns for verb forms and for nouns and adjectives derived from verbs. The program has been tested on a corpus of 242 abstracts from the Saudi Arabian National Conferences and we are in the process of extending the list of roots to handle a newspaper corpus as well.

To represent the Arabic character set, we used the Nafitha software developed by 01 system, Manama, Bahrain (Nafitha 1988).

The morphology system was written with the goal of supporting natural language processing programs such as parsers and information retrieval systems. It is coordinated with a large Arabic lexicon (AlSamara, 1996). It can, however, be used to display whole paradigms for Arabic verbs. It can also display a single form, if the user chooses to specify not just the root but the mood, gender, number, and person. It can also analyze any verb form given to it. In addition to 1,116 roots for regular verbs, the system stores forms for the thirty-nine most common irregular verbs. The Arabic word for morphology is "t(a)Sryf" based on the root "Srf", which has a basic idea of changing direction, averting, and flowing freely. "t(a)Sryf" is the total range of morphological patterns used with a given root (Owens, 1988). Here the "S" in the word "T(a)Sryf" stands for the letter "ص" since there is no corresponding letter in English for this letter.

We became involved in problems of morphology because we need to find stems and roots for purposes of information retrieval (Al- Kharashi and Evens, 1994; Abu-Salem, 1992; Hmeidi, 1995) and parsing (Abu-Arafah, 1995). The morphology system is coordinated with a large lexicon for Arabic (Hammouri, 1994; AlSamara, 1996).

The organization of this paper is very straightforward. The next section contains an overview of other approaches to computational morphology. Then we describe our approach to Arabic morphology and its extension to four and five letter roots as well as the three letter roots that are much more common. Finally, we show examples of the output that the program produces when it is used interactively and conclude with plans for future research.

2. Review of Some Other Morphology Systems.

Systematic attempts at computational morphology in the West were successful enough by 1992 to lead to the almost simultaneous publication of two major books, Sproat (1992) and Ritchie et al. (1992). At about the same time the PC-Kimmo program became widely available (Antworth, 1992). It had been obvious from the very beginning of Arabic language processing that morphology systems were an absolute necessity, because of the extremely complex morphology that Arabic shares with other Semitic languages. Hegazi and ElSharkawi (1986) designed a system to detect the root of any Arabic word along with morphological patterns and word categories. Their system has also been used for detection and correction of mistakes in spelling and vowelization.

Saliba and Al-Dannan (1989) developed a Comprehensive Arabic Morphological Analysis and Generation System at the IBM Scientific Center in Kuwait. Their analyzer examines the input word for different word types and attempts to find all possible analyses. In the analysis process the longest valid prefix and suffix are stripped from the word and the remaining part of the word, which is called the stem, is used to identify a valid Arabic word. If the stem is accepted as a content word (noun or verb) then further analysis processes will be carried out.

El-Sadany and Hashish (1989) developed an Arabic morphological system also designed to carry out both analysis and generation, capable of dealing with vowelized, semivowelized, and nonvowelized Arabic words. This system was developed at the IBM Cairo Scientific Center. The system has the ability to vowelize nonvowelized words. The system was implemented in Prolog on the IBM PS/2 Model 60.

Al-Fedaghi and Al-Anzi (1989) present an algorithm to generate the root and the pattern of a given Arabic word. The main concept in the algorithm is to locate the position of the three letters of a possible triliteral root in the pattern and check to see whether the candidate trigram appears in a list of known roots.

When we began to work on the morphology problem ourselves, our first reaction was to start with PC-Kimmo, which we had used in some experiments with much simpler problems in English morphology. But when we communicated with Evan Antworth of the Summer Institute of Linguistics, he discouraged us:

"The basic two level mechanism as it is implemented in PC-KIMMO can't easily handle (if at all) the distinctive semitic patterns of consonantal root and intercalated vowels".

When we received this message we abandoned our plans to use PC KIMMO and resolved to first extend the El-Anzi and Al-Fedaghi Algorithm to handle quadriliteral roots and then to look for ways to improve on it.

3. Algorithm to Find Quadriliteral Roots.

The first author designed and implemented an algorithm to find quadriliteral roots and their patterns. This algorithm follows the same strategy as the algorithm of Al-Fedaghi and Al-Anzi (1989).

Quadriliteral roots are usually formed as extensions of triliteral roots by reduplicating the final consonant. Thus, the standard triliteral pattern "f9l" becomes the quadriliteral pattern "f9ll." Here 9 stands for the letter "ayn" since there is no corresponding letter in English for this letter. The other forms of quadriliteral verbs are then obtained by adding affixes to the root.

The first step of the algorithm for quadriliteral roots is to search the input form

for a correct pattern. We take a candidate pattern and look for the four letters in the input word (corresponding to f, 9, l, and l). If the letters are found we label their positions, pos1, pos2, pos3, and pos4. Otherwise, we choose the next candidate pattern and try again. Once we have a match in all four positions we go to the second step.

The second step is to extract the root from the input word in the positions pos1, pos2, pos3, and pos4.

4. New Approach to Finding the Root and the Pattern. The algorithm for quadriliteral roots shown in Figure 1 is an extension of the triliteral algorithm of Al-Fedaghi and Al-Anzi (1989). Once we had implemented it successfully, we were concerned that it was somewhat slow, so we searched for a new approach that would give us the same result. This new approach was then implemented for both triliteral and quadriliteral roots.

We describe how our approach works for triliteral roots. The first step is to remove the longest possible prefix. Then we look at the remainder. The three letters of the root must lie somewhere in the first four or five characters of the remainder. What is more, the first letter of the remainder is the first letter of the root since we have removed the longest possible prefix.

We check all possible trigrams within the first five letters of the remainder. That is, we check the following six possible trigrams:
first, second, and third letters
first, second, and fourth
first, second, and fifth
first, third, and fourth
first, third, and fifth
first, fourth, and fifth

In order to test the algorithm, we prepared two files: a file of roots and a file of prefixes. The program outputs the root and the pattern for each word in each of the 242 abstracts. Our colleagues in the Arabic Language Processing Laboratory checked all the results for correctness.

In the abstracts there are 19,167 running words, 16,775 with triliteral roots, and 1,124 with quadriliteral roots, none with quintiliteral roots. The program handles all these correctly. The other 1,268 words are nouns not derived from verbal roots (solid nouns) or borrowings from foreign languages.

The algorithm requires less space and much less time than the Al-Fedaghi and Al-Anzi algorithm. The average time to search for the roots for all words in an abstract is 2.2 seconds and the average time to search for roots with the Al-Fedaghi and Al-Anzi algorithm is 17.2 seconds. The average length of an abstract is 35 words.

5. The Morphology System. The main system menu contains the following options. First, get the various paradigms of the word. This is most often needed by human users and perhaps tutoring programs. Second, get a specific form after passing in a word and mood, person, number, and gender. This is most often needed by text generation systems. Third, analyze the input word to get back the part of speech, person, number, and gender. This is most often needed by a parser. First, get the root of the input word. This is most often needed by information retrieval systems. The main menu of the system is shown in Figure 2. From the main menu the user can select one of the four options. In case the user selects the first option, he/she will get all the information about the input word as seen in Figure 3. When the user selects the second option, the menus in Figures 4, 5, 6, and 7 appear in sequence to select the appropriate codes. Examples of the output in these cases is shown below.

```
begin
get word
for all patterns that have the same length as the input word
do begin
        let pat = pattern
        locate the positions of the letters f, 9, l, and l in pat
        let pos1, pos2, pos3, and pos4 be the positions respectively
          replace the letters in the given word at the positions
          pos1, pos2, pos3, and pos4 with the letters f, 9, l and l respectively
        let  new-word  be the word formed in the previous step
        if (new-word == pat)
              then exit the loop
    end;
        let root = word[pos1]+word[pos2]+word[pos3]+word[pos4]
        return root and the pattern pat
end;
```

Figure 1. Algorithm to find quadriliteral roots

```
WELCOME TO THE MORPHOLOGY SYSTEM
MAIN MENU
[1] GetParadigm (display all information)
[2] GetForm (get specific tense form)
[3] Analyze (get mood/person/number/gender)
[4] GetRoot
```

Figure 2. System Main Menu

Given the word يقدم after passing the mood = imperfect, number = plural, gender = masculine, and person = 3rd person

the form is يقدمون

When the user selects the third option he/she will get the following output:

the input word يقدم

(يقدم verb 3rd sing masc)

When the user selects option four he/she will get the following output:

the input word يقدم

the root of يقدم is قدم

word	معلومات
root	علم
pattern	مفعولات
stem	يعلم

The Paradigm in the Third Person

Present	Past	Gender	Pronoun
يعلم	علم	(mas)	هو
يعلمان	علما	(mas)	هما
يعلمون	علموا	(mas)	هم
تعلم	علمت	(fem)	هي
تعلمان	علمتا	(fem)	هما
يعلمن	علمن	(fem)	هن

The Paradigm in the Second Person

Imperative	Present	Past	Gender	Pronoun
اعلم	تعلم	علمت	(mas)	انت
اعلما	تعلمان	علمتما	(mas)	انتما
اعلموا	تعلمون	علمتم	(mas)	انتم
اعلمي	تعلمين	علمت	(fem)	انت
اعلما	تعلمان	علمتما	(fem)	انتما
اعلمن	تعلمن	علمتن	(fem)	انتن

The Paradigm in the First Person

Present	Past	Gender	Pronoun
اعلم	علمت	(mas)	انا
نعلم	علمنا	(mas)	نحن

Figure 3. The Complete Analysis of the Word "m(a)9lwmat"

Enter the mood Code

[1] For Perfect
[2] For Imperfect
[3] For Imperative

Figure 4. Mood Code Menu

Enter the Gender Code

If the mood is imperative enter [1]
[1] For Masculine
[2] For Feminine

Figure 5. Gender Code Menu

Enter the Person Code

If the mood is imperative enter [2]
[1] For 1st Person
[2] For 2nd Person
[3] For 3rd Person

Figure 6. Person Code Menu

Enter the Number Code

If the mood is imperative enter [1] or [3]
[1] For Singular
[2] For Dual
[3] For Plural

Figure 7. Number Code Menu

6. Conclusion. Morphological analysis is the first step of most natural language processing applications. We have developed a new algorithm that runs an order of magnitude faster than other algorithms in the literature. We plan to make efforts to extend our system to generate adjectives and generate different types of derived nouns.

The area of vowelization deserves further research. It is very important in resolving ambiguity in the meaning of the words and the correct pronunciation of the words. Vowelizing Arabic text is the process of placing the short vowels above and below Arabic consonants. Our concentration in this project has been on the analysis of non-vowelized text. The next step is to investigate more about this area, in order to build a morphological system that can analyze the vowelized text.

References

Abu-Arafah, A. 1995. A Grammar for the Arabic Language Suitable for Machine Parsing and Automatic Text Generation. Ph.D. Dissertation, Computer Science Department, Illinois Institute of Technology, Chicago, IL.

Abu-Salem, H. 1992. A Microcomputer Based Arabic Bibliographic Information Retrieval System with Relational Thesauri (Arabic IRS). h.D i ssertation, Computer Science Department, Illinois Institute of Technology, Chicago, IL.

Al-Fedaghi, S. S., and Al-Anzi, F. S. 1989. A New Algorithm to Generate Arabic Root-Pattern Forms, Proceedings of the 11th National Computer Conference and Exhibition, March, Dharan, Saudi Arabia, 391-400.

Al-Kharashi, I., and Evens, M. 1994. Comparing Words, Stems, and Roots as Index Terms in an Arabic Information Retrieval System. Journal of the American Society for Information Science. Vol.45, No.8, September 1994, 548-560.

AlSamara, K. 1996. An Arabic Lexicon to Support Information Retrieval, Parsing and text Generation. Ph.D. Dissertation, Computer Science Department, Illinois Institute of Technology, Chicago, IL.

Antworth, E.,1992. Glossing Text with the PC-KIMMO Morphological Parser. Computers and the Humanities. Vol-26. Nos. 5-6, December 1992, 389-398.

El-Sadany, T. A., and Hashish, M. A. 1989. An Arabic Morphological System. IBM Systems Journal. Vol.28, No.4, 600-612.

Hammouri, A. 1994. An Arabic Lexicon Database to Support Natural Language Processing. Ph.D. Dissertation, Computer Science Department, Illinois Institute of Technology.Chicago, IL.

Hegazi, N., and ElSharkawi, A. A. 1986. Natural Arabic Language Processing, Proceedings of the 9th National Computer Conference and Exhibition, Riyadh, Saudi Arabia, 1-17.

Hmeidi, I. I. 1995. Design and Implementation of Automatic
Word and Phrase Indexing for Information Retrieval with Arabic Documents. Ph.D. Dissertation, Computer Science Department, Illinois Institute of Technology, Chicago, IL.

Nafitha 1988. User's Guide and Advanced Programmers Reference, release 3.0, Manama, Bahrain: 01 system.

Owens, J. 1988. An Introduction to Medieval Arabic Gramatical Theory. The Foundations of Grammar. John Benjamins, Amsterdam.

Ritchie, G., Black, A., Russell, G., and Pulman, S. 1992. Computational Morphology. MIT Press, Cambridge, MA.

Saliba, B., and Al-Dannan, A. 1989. Automatic Morphological Analysis of Arabic: A Study of Content Word Analysis. Proceedings of the First Kuwait Computer Conference, Kuwait, March, 3-5.

Sproat, R. 1992. Morphology and Computation. MIT Press, Cam- bridge, MA.