A Rule-Based Extensible Stemmer for Information Retrieval with Application to Arabic

Haidar Harmanani[1], Walid Keirouz[2], and Saeed Raheel[1]
[1]Computer Science and Mathematics Division, Lebanese American University, Lebanon
[2]Department of Computer Science, American University of Beirut, Lebanon

Abstract: *This paper presents a new and extensible method for information retrieval and content analysis in Natural Languages (NL). The proposed method is stem-based; stems are extracted based on a set of language dependent rules that are interpreted by a rule engine. The rule engine allows the system to be adapted to any natural language by modifying the NL semantic rules and grammar. The system has been fully tested using Arabic, and partially using English, Hebrew, and Persian. We have validated our approach using a database-based prototype.*

Keywords: *Natural language processing, information retrieval, stemming*.

Received February 21, 2005; accepted July 13, 2005

1. Introduction

Automatic Information Retrieval (IR) is a process that informs on the existence and whereabouts of information relating to a specific request. Queries supplied by users are composed of a set of words interrelated by boolean operators; the system responds by locating the documents containing combinations of these query words. The retrieval process is influenced by the indexing process as well as by the natural language that is being indexed. Indexing may be based on meta-data in the documents headers. Other techniques are based on full-text indexing where the result is a set of keywords that constitute the content of the document. Indexing can also be thesaurus-based where indexing in this case relies on the synonyms of words in addition to the words themselves. The retrieval effectiveness is evaluated using a pair of measures, recall and precision. Recall is the proportion of relevant material actually retrieved from the file while precision is the proportion of the retrieved material that is found to be relevant to the user's information needs [13]. Various techniques have been used to enhance the recall level such as the use of term phrases, the use of truncated terms or stems, and the use of synonyms. A high precision level is achieved by assigning weights to terms as well as the use of indexing phrases [13, 16].

In Natural Language Processing (NLP), stem extraction or lemmatization relates a given item to the lexical or grammatical morpheme it goes back to [3]. Stemming is language specific and requires linguistic expertise and an understanding of the needs of information retrieval [13]. Stemmers have been developed for a range of languages including English [13], Latin [6], French [12], Turkish [5], and German [11].

Stemming is applied in order to improve search performance by stemming documents and query terms. Kraaij *et al*. [8] noticed that stemmers yield a significant improvement over non-stemmers and that derivational stemmers generally give the best results. Krovetz [9] noted that improvements from stemming increase at higher levels of *recall* and that derivational morphology is responsible for improvement at high levels of *precision*. In [15], Salton *et al*. propose a phrase generation method that helps in generating texts' content identification.

Within the context of Arabic, several morphological analyzers have been developed [1, 2, 7]. However, very few of these stemmers have received a standard IR evaluation. Recently, Larkey *et al*. [10] proposed a light stemmer for Arabic retrieval. This stemmer is based on a simple repartitioning process followed by clustering using co-occurrence analysis. The process produces stem classes that are better than no stemming or very light stemming, but is still inferior to good light stemming or morphological analysis. Furthermore, statistical term co-occurrence generates a large number of potential phrases, many of which are semantically improper.

This paper presents an extensible method for natural languages indexing and search. The method is based on a rule engine that allows the system to be adapted to a variety of natural languages without the need to develop a specialized IR system. The proposed method uses full-text indexing and ranks terms according to their degree of relevance. The term's occurrence, its stem's occurrences in addition to the term and stem positions are used in order to calculate the term's

degree of relevance. Furthermore, the system uses a predefined thesaurus that have a minimal set of keywords and their types. The thesaurus is built incrementally by adding key terms and term phrases that are encountered in target documents.

The remainder of the paper is organized as follows. Section 2 briefly describes the Arabic language while section 3 describes our information retrieval system. Section 4 describes the system's extensibility to other languages. Experimental results are presented in section 5. We conclude with some remarks and observations in section 6.

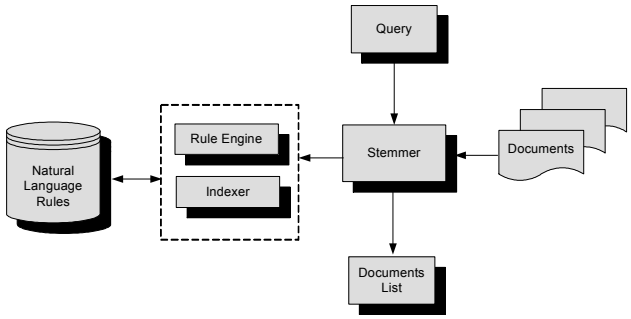

Figure 1. Extensible information retrieval system.

2. Arabic Language

Natural languages rely on grammatical, syntactical, and morphological rules. The level of difficulty and sophistication depends on the language itself. Arabic is a highly inflected language and has a particularly acute need for effective normalization and stemming. Both orthography and morphology give rise to a wide range of lexical variation. Vocalized text includes diacritics for short vowels and other details, conveying a nearly phonetic representation of a word. Non-vocalized orthography is more ambiguous and can cause a mismatch with texts, dictionaries, or queries that are vocalized [10]. Additional variability arises from the derivational and inflectional productiveness of Arabic. A given word can be found in many different forms that should possibly be conflated for information retrieval. Arabic has three genders, masculine, feminine, and neutral. It also has three persons, one to describe the speaker, one to describe the person being addressed and one to describe the person that is not present.

At a deeper level, most Arabic words are built up from and can be analyzed down to roots. There are some exceptions to this rule especially common nouns and articles. Roots are usually composed of three consonants and there are roots consisting of two, four, and five consonants [4]. Words are derived from these roots following a fixed pattern that adds prefixes, suffixes and infixes to the word. Prefixes and suffixes can be added to words that have been built up from roots to add number or gender (among others). Derivational morphology is the science of studying and describing roots. For example, the pattern *k t b* (ك ت ب) forms the root of the verb *katab*[1] (كتب) which means "to write." This stem can derive the noun *kutub* (books) while by adding the suffix "*y"*the term becomes *kutuby* (my books). Other derivations change the tense of a verb by adding a prefix such as *ya'* in *yaktob* (he is writing). Other words that are derived from the same root are *maktab* (office), *kitaab* (book), *kataba* (he wrote), *katabat* (she wrote), *naktubu* (we write), and *katabna* (we wrote). Additional derivations denote the active participle or the analogous adjective.

On the other hand, the three letters paradigm fâ' *c*ayn lâm (ف ع ل) is the basis for all stems in Arabic. Rhyming is performed by comparing all the additions performed on the three basic letters to the additions performed on the three basic letters that form the stem of the word that is being processed. If they match then the word and the pattern "rhyme" and thus we extract the stem of that word by extracting only the three letters that fall exactly in the same positions as the three letters fâ', *c*ayn, and lâm in the pattern. What is common among all derivations is that they all preserve the existence of those three letters. For example, the word *yaktob* rhymes with the pattern *yaf*^{c}*al* and thus the stem is *k t b*.

3. Arabic Stem Extraction and Indexing System

Our work aims at embedding morphological rules that can be used for stemming in the system through a set of rules. The rules use morphological pattern-matching operators and are processed by a rule engine that enables the system to be easily adapted to other NLs. The system, shown in Figure 1, performs stemming on target documents. The stems are next indexed and stored in the database with the relevant statistics and information. Phrases are constructed and stored as well. Finally, the system applies the reverse process for information retrieval.

3.1. Rule Engine and Natural Language Rules

The rule engine allows the system's extensibility through dynamic configuration of the natural language morphological rules needed during indexing. The rule engine parses the rules and checks for consistency as well as for syntactic errors. The rules are next transformed into database records that are stored in the "rules" table. The indexer uses the rules table to perform the proper stemming operation.

A rule may imply an action as simple as discarding a noise word or as complex as a morphological rule that leads to a stem extraction. A rule in our IR system is represented in the following EBNF format:

1 In this paper we will use, for convenience, Latin characters to represent Arabic characters based on the transliteration in Appendix A.

Rule := <TYPE-PRIORITY> IF <Condition>
<Value> (OR <Value>)? (AND <Condition>
<Value> (OR <Value>)?)? THEN <Action>
<Value>+ (AND <Action> <Value>+)?

Where
<Type-Priority>:=
['SL'|'ND'|'PF'|'SF'|'RH'|'ST']D+
<Condition>:= 'TERM_IS_EQUAL_TO' |
'STARTS_WITH' |
'ENDS_WITH' |
'RHYMES_WITH' |
'STEM_ENDS_WITH'|
'STEM_IS_EQUAL_TO'
<Value>:= ['أ-'ي'] +
<Action>:= 'DISCARD' |
'NEXT_WORD_IS_A_NOUN' |
'NEXT_WORD_IS_A_VERB' |
'REMOVE_LETTERS' |
'SAVE_AS_IS' |
'RHYME_IS_EQUAL_TO' |
'TERM_IS_A_NOUN' |
'TERM_IS_A_VERB' |
'EXTRACT_STEM' |
'REPLACE_WITH' |
'DONT_REMOVE_LETTERS'

The Type-priority is used to enforce rules precedence relationships that exist among various rules. In fact, the rules are classified by the system based on their types and priorities. Condition represents an action that will be applied by the indexer when the term's value is encountered. Finally, action is the resulting action that will be applied as a result of the indexing process. In what follows, we will use two simple examples to illustrate the use of these rules.

3.1.1. Example Rhyming Rule

In order to apply a Rhyming rule (RH) to check for the pattern *yaf*^{c}*aloun* (يفعلون), one may write the following rule:

RH6 IF RHYMES_WITH يفعلون THEN EXTRACT_STEM AND TERM_IS_A_VERB

RH6 indicates that the rule deals with *rhymes* and its priority is six. The RHYMES_WITH indicates that an action "rhyme" will be applied in the rhyme's table. The EXTRACT_STEM action instructs the compiler to add the *yaf*^{c}*aloun* (يفعلون) pattern to the rhymes table while the TERM_IS_A_VERB indicates that this pattern is used with verbs.

3.1.2. Example Noise Word Rule

In order to apply a Stop List (SL) or a noise word rule, one may write the following rule:

SL2 IF TERM_IS_EQUAL_TO أن THEN DISCARD AND NEXT_WORD_IS_A_VERB

SL2 indicates that the rule deals with noise or stop list terms and has a priority equal to two. The TERM_IS_EQUAL_TO indicates that the term *an* (أن) should be ignored since it is a *Stop List* term. Finally, this type of stop word usually precedes verbs and thus we use this fact to predict that the type of the next word is a verb using the NEXT_WORD_IS_A_VERB action.

3.2. Stemmer

In order to derive the appropriate stem, the stemmer analyzes terms in order to determine whether they are verbs or nouns as the stemming technique differs for either type. The stemmer, shown in Figure 2, examines initially the preceding term in order to check for trivial cases. For example, some noise words precede verbs (*adawat al naseb waljazem*) only while some other noise words precede nouns only (al asma' al mawsoula and ahrof al jar). If this fails, then rhyming is used. For example, if a term matches with *yaf*^{c}*al* then it is a verb; however, if on the other hand, it matches with *fa*^{c}*ela* then it is a noun. If none of the above approaches were able to determine the type of a term, then the system will treat this term as *unknown* and will process it twice, as a *noun* as well as a *verb*.

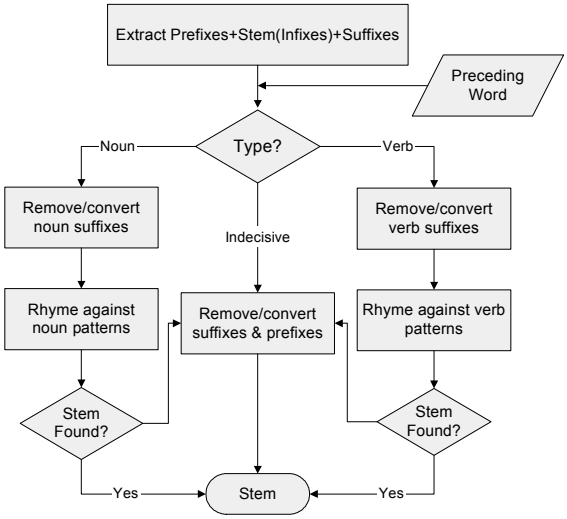

Figure 2. Arabic stemming process.

3.2.1 Stemming Verbs

In order to determine the stem of a verb, the system checks and removes any attached pronouns (*al dama'er al mottasila*). Attached pronouns come either in the form of suffixes or in the form of prefixes. The attached pronouns are removed through pattern matching. The remainder of the term undergoes *rhyming* and is matched against various patterns. If there is a match, then all additional letters are removed. It should be noted that patterns in Arabic are categorized into various fixed categories. In what follows, we describe two main verb categories though our system handles all possible rhymes.

The first category is known as the *common five verbs*, referred to in Arabic as *al af^{c}al al xamsa*. These verbs constitute a finite set and the additions are known beforehand. Thus, the system does not need to perform pattern matching to derive the verb's stem but simply removes the known additions. For example, the main characteristic of these verbs is that they always end with an *n* (ن) unless they are preceded by articles (*adat jazem* or *adat nas**e**b*). In this case, the *n* is removed and substituted by either an *'alef* or *yâ'*.

The second category is known as the *ten additions* (*azziyadat al ^{c}as**a**ra)*. In these verbs, additions never come in the form of suffixes but rather as prefixes or in the middle of the word, which complicates the stemming process. As it is the case with the common five verbs, the word is rhymed against these patterns. If a match is found, then the additional letters are removed. Whenever a word has been determined to be a verb, it has to undergo the stemming process like in the previous section.

Table 1. A subset of T-derivations (مصدر تائي).

Stem	Example	T-Derivation
Xrj (خرج)	Taxrej (تَخريج)	Taf^{c}el (تفعيل)
Bdl (بدل)	Tabdol (تَبدُل)	Taf^{c}ol (تَفعُل)
Qsm (قسم)	Taqasom (تَقاسُم)	Tafa^{c}ol (تَفاعُل)

3.2.2. Stemming Nouns

Stemming techniques differ in the case of nouns as nouns introduce variability, irregularity, and inflection. For example, nouns can be singular, double, or plural. Variations and irregularities in these forms also exist and the stemming process is even more complicated when dealing with masculine or feminine terms. Nouns may also have prefixes that are attached to them such as the letters are *al*, *kal*, *fal*,*bal* , *lâm* (ل, بال, فال, كال, ال); these letters are removed through pattern matching. Nouns may belong to a special category called the common five nouns (*al asma' al khamsa*). Pronouns may be attached to nouns (*al dama'er al mouttas**i**la*) and are only variations for the attached pronouns (*al dama'er al mounfas**i**la*). These pronouns come in the form of suffixes. Nouns also have a state that is indicated either by diacritical marks or by letters in some other cases. This state depends on the term's position in the sentence or upon the preceding word. The system also has to detect whether a noun is in a singular, plural state, or *mou<u>t</u>anna*, which is the case when dealing with two entities. Furthermore, nouns introduce another difficulty, which is that the grapheme differs when addressing a male or a female. For example, *kitabouho* adds a singular masculine pronoun to the noun *kitab* in order to represent the singular masculine morpheme whereas it adds the *ha* to form *kitabouha in order* to represent the singular feminine morpheme. *kitabouhonna* is for the plural feminine state whereas *Kitabouhom* is for the plural masculine state. Finally, Arabic has an irregularity case that arises when forming the plural. These cases are known as the irregular plural (جمع تكسير or jame^{c} taksir). For example, the plural of *imraa* (امرأة) is *niswa* (نسوة) and the regular plural of *s**a**jara* (شجرة) is *s**a**jarat* (شجرات) while a*s**a**jar* (أشجار) is the irregular plural form.

When the system detects a three-letter noun, it performs pattern matching rather than rhyming. If the noun proves to belong to the five common nouns, then the first two letters are extracted as the stem for this word. It should be noted that among these nouns, only *<u>d</u>ou* (ذو) is considered a stop list term. Next, the system uses pattern matching to decide whether a noun is in its plural or singular form. If it matches then the additions are removed and the noun is then reduced to its singular form. On the other hand, if it were in an irregular plural from, then rhyming is used and additions are removed. However, if, the noun does not match with any of the regular plural forms and does not rhyme with the irregular forms, then it is assumed to be in its singular state. Finally, nouns have derivations known in Arabic as *al mas**a**der al mou'awwala* (المصادر المؤولة). The noun's derivations are divided into three categories: derivations that start with mîm (م) and are called *mas**a**der mimiyya* (ميمية مصادر), derivations that start with a tâ' (ت) and are called *mas**a**der ta'iyya* (مصادر تائية), and finally a group of miscellaneous derivations that have no specific common format or rule. In order to simplify the indexing process, the stemmer checks for the first letter. If it were a mîm, then it is rhymed against the *mas**a**der mimiyya* patterns. If it were a tâ', then it is rhymed against the *mas**a**der ta'iyya* patterns. If however the derivation is irregular, then it is rhymed against the miscellaneous patterns. The stem is extracted by removing all the additions based on the corresponding pattern. Table 1 shows a subset of the T-derivations with the additions are underlined.

3.3. Indexer

The indexer interacts with the stemmer and stores the stemmed terms in the database. Furthermore, the indexer performs relevance calculation by assigning weights to the stored terms in order to enhance the retrieval process. This is achieved through the computation of the recall and precision measures. Furthermore, the indexer constructs the phrases based on the proximity measures in order to facilitate the search for phrases. During search, the documents with the highest degree of proximity will be returned first while the ones with the least proximity will be displayed last.

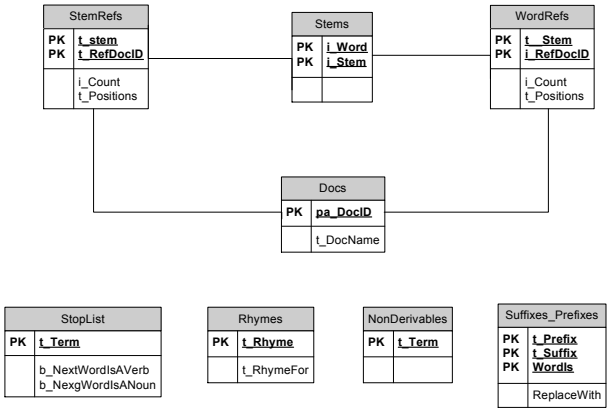

Figure 3. Database relational diagram.

3.4. Database Representation

The proposed system uses a database to store rules, terms, and phrases in addition to references to the corresponding documents. In what follows, we describe these tables.

- *Stop List Terms*: Stop list terms are noise words that are discarded by the system. The *Stop List* table stores these terms.
- *Rhymes*: The *Rhymes* table stores the patterns that will be used in stemming and indexing words and verbs along with the corresponding category to which each pattern belongs.
- *Non-Derivable*: Non-derivable terms, such as proper names, are terms for which the stemming process should not be applied and are stored in the database as they are.
- *Suffixes and Prefixes*: This table holds the values of all the suffixes and prefixes that can be attached to nouns and verbs.
- *Stems*: This table holds the indexed terms at the end of the indexing phases. Thus, once a term's stem is derived, the original term and its stem are stored in this table. This feature has been added to improve the stemming efficiency.
- *Documents*: This table holds the references to newly indexed documents.
- *Words References*: This table stores the words that have been processed along with their document reference (docID). A word is also associated with a count that allows the system to identify its occurrence within a specific document. This information is essential in determining and calculating the degree of relevance that the word has in its contextual document.
- *Stems References*: This table stores the stems for each word rather than the words themselves and is used in order to determine the terms' degree of relevance. The Stems table increases the efficiency of the indexing process by checking whether a term is new. If it is, then a new entry is added to the table along with the term's location. However, if a word has already been added to the Stems table, its corresponding entry in the Words references table is modified by adding a new position reference.

4. Experiments in NL Extensibility

In order to verify the extensibility of the system, we adapted the system to three other languages, Persian, Hebrew, and English. In what follows, we briefly describe the extensibility process vis-à-vis Hebrew and Persian only due to space restriction. However, it should be noted that it was quite simple to write a light stemmer for English using our stemmer.

4.1. Hebrew

Adapting our system for the Hebrew language, was as tough as in Arabic. While Hebrew has some similarities to Arabic in terms of grammatical and syntactical rules, it is morphologically more complex. In what follows, we present a couple of simple rules for the sake of illustration only. Please note that these rules are simply shown to illustrate the extensibility of our system and as in the case of Arabic, these rules are in no way complete and comprehensive. In order to write a stop list rule in Hebrew, it is as simple as:

SL3 IF TERM_IS_EQUAL_TO אַתּה THEN DISCARD

Where SL3 indicates that the rule deals with stop list and its priority is three. The rule removes the work אַתּה, which is equivalent to the word 'you' in English. While a non-derivable rule can be written as:

ND1 IF TERM_IS_EQUAL_TO יהושע THEN SAVE_AS_IS

Where ND1 indicates that the rule deals with a non-derivable. The rule removes the proper noun יהושע.

In addition, rules to remove a suffix or prefix can be written as:

SF2 IF ENDS_WITH וח THEN REMOVE_LETTERS AND TERM_IS_A_NOUN

Where SF2 indicates a suffix rule with priority two. The rule removes the suffix letters וח that are added to a noun to represent its feminine state.

PF1 IF STARTS_WITH הָ THEN REMOVE_LETTERS AND TERM_IS_A_NOUN

Where PF1 indicates a suffix rule with priority one. The rule removes the prefix הָ that is equivalent to word "the" in English and *al*(ال) in Arabic.

4.2. Persian

The Persian language is simpler than Arabic or Hebrew in terms of syntactic and semantic rules. There is no masculine, feminine, or *Mouthanna* (the state for two entities) states. The Persian language does not have

patterns; however, there are too many suffixes, some prefixes and the stop list for that language is quite long. Therefore, writing a Persian light stemmer was simple and fast. In what follows, we present a couple of simple rules for the sake of illustration only. For example, in order to remove a noise word, one would write the following rule:

SL2 IF TERM_IS_EQUAL_TO دو THEN DISCARD

Where دو is a noise word similar to the 'two' in English, which considered as a *stop list* term and thus has to be discarded.

Another rule is shown next to remove a noise word, كدام, meaning *which* in English. The rule can be written as follows:

SL4 IF TERM_IS_EQUAL_TO كدام THEN DISCARD

To revert a word in plural to its singular form, one would write the following rule:

SF2 IF ENDS_WITH ان OR ها THEN REMOVE_LETTERS

Where ان and ها in the above rule are suffixes and are equivalent to the plural 's' in English.

Another rule that removes prefixes is illustrated next. The rule removes the attached prefix ن, which is equivalent to 'not' in English.

PF1 IF STARTS_WITH ن THEN REMOVE_LETTERS

Finally, the following rule removes the stop list terms تو (too), وي (we), شما (shma), إيشان (ishan) which are equivalent respectively to the pronouns *you, he, you* (plural), and *we* in English.

SL2 IF TERMS_IS_EQUAL_TO تو OR وي OR شما OR إيشان THEN DISCARD

5. Experimental Results

The proposed system has been implemented using the C Language and uses a Java front-end in addition to a Microsoft SQL Server as a back-end RDBMS. In order to test our system, we conducted two types of experiments. The first experiment was based on indexing duration; the second was based on search hits. The first experiment was based on a random sample of Arabic documents that we picked from the Internet. The documents deal with different issues and are of different sizes. All documents were processed and the stems were extracted and stored in the "Stems" table. The Stems table improved the performance of the system dramatically by reducing the indexing duration as shown in Figure 4. In general, as the number of words increases, the time needed to perform the stemming and indexing increases as well. However, by saving the intermediate results, the indexing duration has been reduced significantly as shown in Figure 4. On average, the results indicated that the indexing speed has been enhanced by approximately 75%. The indexing time is not proportional to the document's size but is rather affected by the decreasing factors such as the already indexed documents. Figure 5 shows the accuracy of search hits performed while indexing. The more documents are indexed the higher the accuracy becomes. The accuracy reaches 100% as the search progresses.

6. Conclusion

We have presented a new and extensible method for information retrieval and content analysis in natural languages (NL). The method is stem-based and is based on a *rule engine*. The rule engine allows the system to be adapted to *any* natural language by modifying the natural language semantic rules and grammar. The system has been fully tested using Arabic, and partially using English, Hebrew, and Persian. Results have been promising.

Currently, we are working on improving the system to handle some difficult exceptions in rhyming such as *al afaal al mocatallat al akher (*اللأفعال المعتلة الآخر). One of the main problems with these verbs is that their derivations change the morphological representation of their original stem. For example, let us consider the verb *rama* (رمى) or threw in English. The morphological representation of its present conjugate is *yarmi* (يرمي). Unlike the regular verbs, the present tense of this verb has changed the last character, the *alef maksoura (*ى or ألف مقصورة) into a *ya'* (ي) and therefore when the indexer attempts to extract its stem by matching it against the pattern *yafcal* (يفعل) the stem will be*rami* (رمي) instead of *rama* (رمى).

Acknowledgement

The authors would like to express their sincere thanks and gratitude for Prof. Albert Naccache for his insight and help in linguistics and in Arabic processing. Thanks also are due to Prof. Dalal Abbas for her help in Persian and to Dr. Khalil Jaffal for his help in Hebrew.

Appendix A

Arabic Transliteration Conventions

The transliteration used in this document is adapted from Dichy [3]. It includes no special character, for the sake of portability. 'Emphatic' (pharyngealized) consonants as well as the voice-less pharyngeal *hâ'* are in boldface. Underlining is used to distinguish constrictive consonants from their occlusive

counterpart, or from a 'neighboring' phoneme. 'Long' vowels bear a circumflex accent.

- *Short Vowels*: a, u, i.
- *Long Vowels*: 'alif = â ; wâw = û ; yâ' = î.
- *Consonants (in Alphabetic order)*
- *Morphograms: tâ' marbûta* = +a&.

hamza = '	bâ' = b	tâ'= t	t̲â' = t̲	jîm = j
hâ' = **h**	xâ' = x	dâl = d	dâl =d	râ' = r
zây = z	sîn = s	sîn = s	**s**âd = **s**	**d**âd = **d**
tâ' = **t**	**d̲**â' = **d̲**	ᶜayn =ᶜ	~~g~~ayn = ~~g~~	fâ' = f
qâf = q	kâf = k	lâm = l	mîm = m	Nûn = n
*h*â' = h	wâw=w	yâ' = y.		

Appendix B

Rules Conditions and Actions

The general form of a *rule* is:

> <Priority> IF <Condition> <Value> [OR <Value> OR <Value>…] [AND <Condition> <Value>…] THEN <Action> [<Values List>] [AND {<Action>} [<Values List>] AND] …

<Conditions> section

1. *TERM_IS_EQUAL_TO*: This condition indicates an action that depends on the following <action> section.
2. *STARTS_WITH*: This condition is used to handle prefixes.
3. *ENDS_WITH*: This condition is used to handle suffixes.
4. *RHYMES_WITH*: This condition is used in order to rhyme a term with a specific pattern <value>.
5. *STEM_ENDS_WITH*: This condition is used to control the removal of suffixes. It is mainly used with the REPLACE_WITH action.
6. *STEM_IS_EQUAL_TO*: This condition is used to check the value of a certain stem after the lemmatization process takes place and then deals with it according to what the <action> section denotes.

<Actions> section

1. *DISCARD*: This action instructs the indexer to discard the term in the <Value> argument of the condition section. This mainly causes the argument to be added to the StopList list.
2. *NEXT_WORD_IS_A_NOUN*: This action indicates that the next term is an improper noun.
3. *NEXT_WORD_IS_A_VERB*: This action indicates that the next term is a verb.
4. *REMOVE_LETTERS*: This action causes the suffixes and prefixes found in the <Value> argument to be removed from the words they encountered in.
5. *SAVE_AS_IS*: This action is used mainly with proper nouns, names, and foreign words. The term is added to the NonDerivable list.
6. *RHYME_IS_EQUAL_TO*: This action indicates the presence of a pattern that should be added to the Rhymes table.
7. *TERM_IS_A_NOUN*: This action indicates that the term is a noun. It is usually used with prefixes and suffixes.
8. *TERM_IS_A_VERB*: This action indicates that the term is a verb and is usually used with prefixes and suffixes.
9. *EXTRACT_STEM*: This action instructs the indexer to extract the stem of the term found in the <Value> argument of the condition section.
10. *REPLACE_WITH*: This action instructs the indexer to replace the characters of the term in the <Value> argument with whatever values are provided in its <Value> argument.

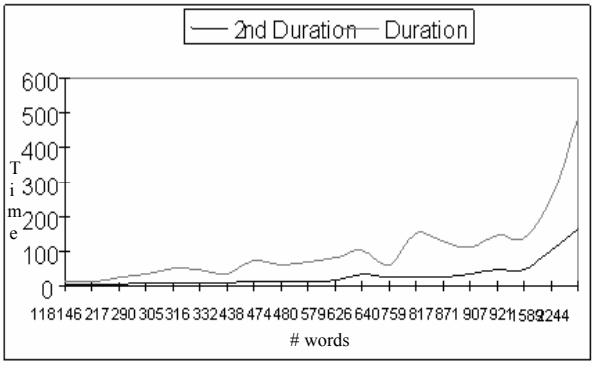

Figure 4. Indexing duration.

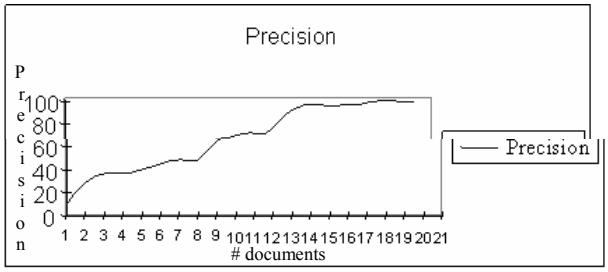

Figure 5. Hits accuracy.

References

[1] Beesley K., "Arabic Finite-State Morphological Analysis and Generation," *in Proceedings of the 16th International Conference on Computational Linguistics*, vol. 1, pp. 89-94, 1996.

[2] Darwish K., Doermann D., Jones R., Oard D., and Rautiainen M., "TREC-10 Experiments at Maryland: CLIR and Video," *in Proceedings of TREC'2001*, Gaithersburg: NIST, 2001.

[3] Dichy J., "On Lemmatization in Arabic: A Formal Definition of the Arabic Entries of Multilingual Lexical Databases," *in Proceedings of the Workshop on Arabic Language Processing*, Toulouse, 2001.

[4] Dichy J. and Ammar S., *Les Verbes Arabes*, Bescherelle, Paris, 1999.
[5] Ekmekcioglu F., Lynch M., and Willett P., "Stemming and N-gram Matching for Term Conflation in Turkish Texts," *Information Research News*, vol. 7, no. 1, pp. 2-6, 1996.
[6] Greengrass M., Robertson A., Robyn S., and Willett P., "Processing Morphological Variants in Searches of Latin Text," *Information Research News*, vol. 6, no. 4, pp. 2-5, 1996.
[7] Khoja S. and Garside R., "Stemming Arabic Text," Computing Department., Lancaster University, available at: http://www.comp.lancs. ac.uk/comp uting/users/khoja/, 1999.
[8] Kraaij W. and Pohlmann R., "Viewing Stemming as Recall Enhancement," *in Proceedings of ACM SIGIR*, pp. 40-48, 1996.
[9] Krovetz R., "Viewing Morphology as an Inference Process," *in Proceedings of ACM SIGIR Conference*, pp. 191-202, 1993.
[10] Learkey L., Ballesteros L., and Connell M., "Improving Stemming for Arabic Information Retrieval: Light Stemming and Co-Occurrence Analysis," *in Proceedings of the ACM SIGIR'02*, August 2002.
[11] Monz C. and de Rijke M., "Shallow Morphological Analysis in Monolingual Information Retrieval for German and Italian," *in Proceedings of the Workshop of Cross-Language Evaluation Forum (CLEF'2001)*, pp. 262-277, 2001.
[12] Moulinier I., McCulloh A., and Lund E., "West Group at CLEF 2000: Non-English Monolingual Retrieval," *in Proceedings of the Workshop of Cross-Language Evaluation Forum (CLEF'2000)*, pp. 176-187, 2001.
[13] Salton G., "Another Look at Automatic Text Retrieval Systems," *CACM,* vol. 9, no. 7, pp. 648-656, 1986.
[14] Salton G., Allan J., and Buckley C., "Approaches to Passage Retrieval in Full Text Information Systems," *in Proceedings of SIGIR*, pp. 49-58, 1993.
[15] Salton G., Zhao Z., and Buckley C., "Automatic Text Decomposition Using Text Segments and Text Themes," *in Proceedings of Hypertext'96*, pp. 53-65, 1996.
[16] Yu C., Salton G., and Siu M., "Effective Automatic Indexing Using Term Addition and Detection," *Journal of the ACM*, vol. 25, no. 2, pp. 210-225, April 1978.

Haidar Harmanani received his BSc, MSc, and PhD degrees in computer engineering from Case Western Reserve University, Cleveland, Ohio, in 1989, 1991 and 1994, respectively. He joined the Lebanese American University (LAU), Byblos, Lebanon, in 1994 as an assistant professor of computer science. Currently, he is an associate professor of computer science and the chair of the computer science and mathematics division at LAU, Byblos. Dr. Harmanani has been on the program committee of various international conferences. His research interests include electronic design automation, high-level synthesis, design for testability, design and testing of SoC, and cluster parallel programming. He is a senior member of IEEE and ACM.

Walid Keirouz received the BE degree in civil engineering from American University of Beirut (AUB) in 1981, and the MSc and PhD degrees in civil engineering from Carnegie-Mellon University Pittsburgh in 1983 and 1988, respectively. In 1988, he joined the Schlumberger Laboratory for Computer Science in Austin, Texas as a research scientist. In 1994, he joined the Lebanese American University. In 1997, he was a visiting researcher in civil engineering at the University of Washington in Seattle, WA, USA. He spent the 1998-1999 academic year as a visiting researcher at the National Institute of Standards and Technology (NIST) in Gaithersburg, MD, USA. In 2001, he joined the American University at Beirut where he is currently an assistant professor. He is the author of about twenty publications and the holder of two patents recognized by the US Patent Office. He is a member of ACM and IEEE.

Saeed Raheel received his BSc and MSc in computer science from the Lebanese American University in 1998 and 2003, respectively. Mr. Raheel has worked with various regional companies in the domain of web solutions. Currently, he is a technical engineer where he is involved in the design and troubleshooting of documents imaging and archiving.