DEVELOPMENT OF LEXICONS GENERATION TOOLS FOR ARABIC: CASE OF AN OPEN SOURCE CONJUGATOR

Mourchid Mohamed[1], El Faddouli Nour-eddine[2] and Amali Said[3]

[1]MIC search team, Laboratory MISC, Faculty of sciences, Ibn Tofail University Kenitra-Morocco
[2]RIME search team, LRIE laboratory, Mahammadia School of Engineers, Mohammed V[th] University in Rabat, Morocco
[3] OMEGA search team, Laboratory LERES, Faculty FSJES, Moulay Ismaïl University Meknes-Morocco

ABSTRACT

The dictionary resources are very important for Natural Language Processing (NLP). Generating high quality dictionary resources is a crucial step for the success and effectiveness of NLP application. Linguistic information about lexical database is complex, large size and various (ie, phonological, morphological, syntactic, semantic and pragmatic). Among such lexical database entries, we find conjugated verbs. To this end, we present in this paper the open source mobile application of our conjugator that we developed in Java platform under the Android. This Conjugator allowed us to generate a lexicon of more than 18667 conjugated verbs. This lexicon will be used to generate textual words. The resultant lexicon can be used in various applications such as morphological analysis (lexical approach), text indexing, etc…

KEYWORDS

NLP, Lexical databases, conjugator, Root, Pattern, textual words, morphological analysis.

1. INTRODUCTION

In the Arabic language, each word having a meaning consists of a root and a pattern. So we can represent all the Arabic words by a matrix in which the patterns are the columns and roots are the rows. A word is simply an element in this matrix [1] [3] [5] [9] (see Figure. 1).

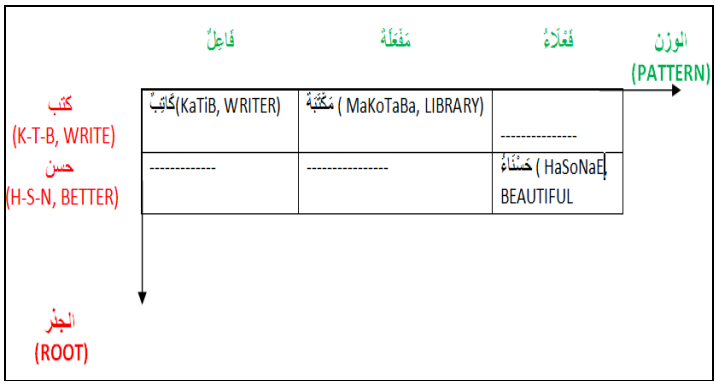

Figure 1. Matrix Root/Pattern

DOI: 10.5121/ijnlc.2016.5202

With this matrix, we can perform two operations [7][10]:

- Analysis: extract the pattern and the root of a given word.
- Generation: building a word from a column (pattern) and a row (root).

Figure 2 shows an example of using this matrix to extract the root and the pattern of the word مَكْتَبَةٌ (MaKtaba , Library) and to generate the word كَاتِبٌ from the root كتب (Root KTB, Write)and the pattern فَاعِلٌ.

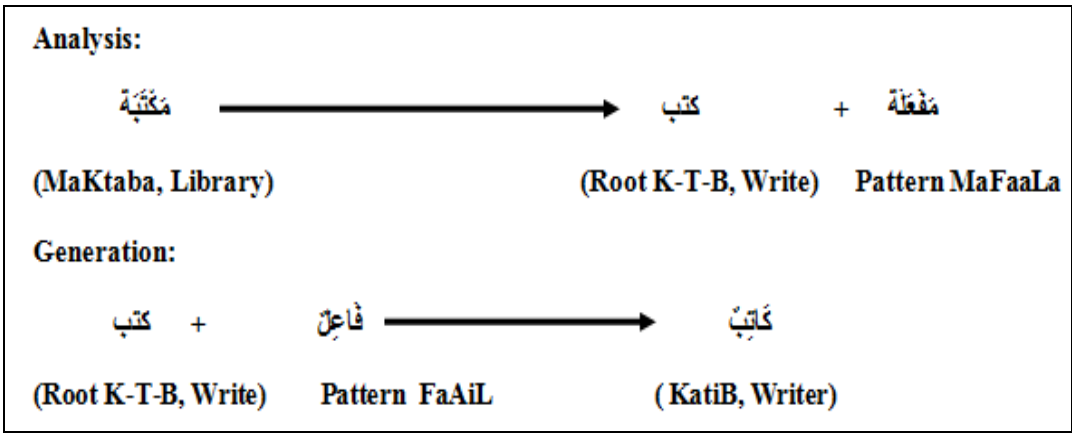

Figure 2. Operations: Analysis and Generation

In this paper, we present the approach of our conjugator developed in Java in two versions. The first is a mobile application under android, while the second is a desktop application to power a lexicon for morphological analysis.

The principle of morphological generation is presented in the second section.

In the third section, we describe the different approach steps of our conjugator and the used repository.

The fourth section is dedicated to description of the conjugator mobile application.

The exploitation of conjugator to enrich the lexicon of morphological analysis by generating textual words is described in the fifth section.

In conclusion, we discuss the results and the main perspectives of our research.

2. MORPHOLOGICAL GENERATION PRINCIPLE

Morphological generation is a succession of morphological operations applied to an initial word accompanied by a set of attributes, to produce a final form of the word.
These attributes are for :

- Names: Number (singular, dual, plural); case (nominative, accusative, gerund); determination (defined, undefined), annexation (annexed)

- Verbs: Aspect (perfect, imperfect, imperative); voice (passive, active); Number (singular, dual, plural); gender (male, female); person (first person, second person, third person); case (nominative, accusative, apocope); insistence.

2.1. Generation Methods

There are three generation methods differentiated by their approaches [5].

2.1.1. Successive Transformations Method

This method relies on applying transformations progressively until obtaining of the final form. These transformations are shown as follows:

$$\text{Initial word} + AT_1 \Longrightarrow word_1$$
$$word_1 + AT_2 \Longrightarrow word_2$$
$$\ldots.$$
$$word_{n-1} + AT_n \Longrightarrow \text{final word}$$

Where AT_i is an attribute.

The figure 3 shows how to get final word الْعُلُومَ (the sciences) from the initial word عِلْمٌ (science) using the set of attributes(Plural , Defined , Accusative).

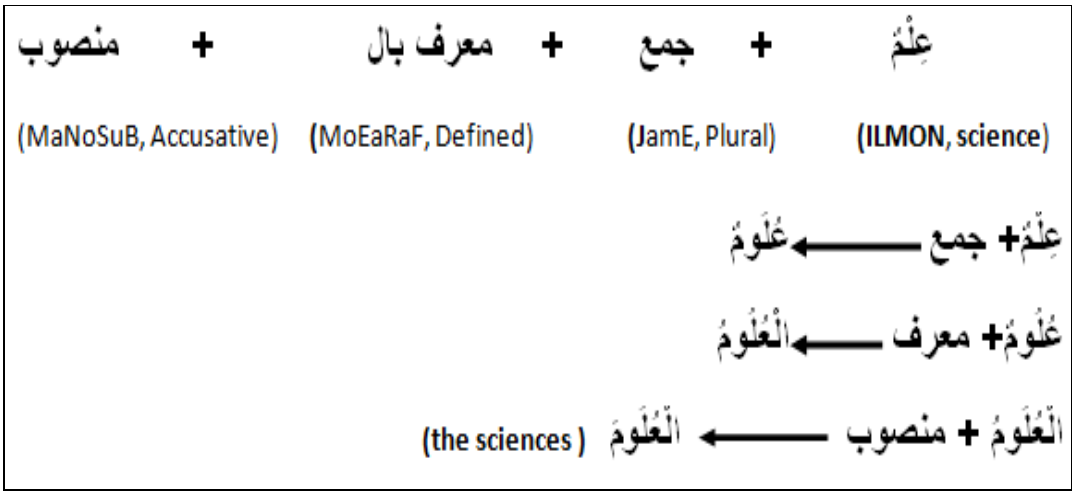

Figure 3. The final word is الْعُلُومَ (AL-OLOMA, the sciences)

2.1.2. Method of Pre-established Models

This method is based on the use of a set of predetermined models each of which is associated with a class of words having the same characteristics.

Thereby the word generation is done in two steps:

- Determination of suitable model from the characteristics of the original word.
- Performing a substitution operation from the initial word and the corresponding model.

The figure 4 depicts the steps followed to obtain the mould from a set attributes.

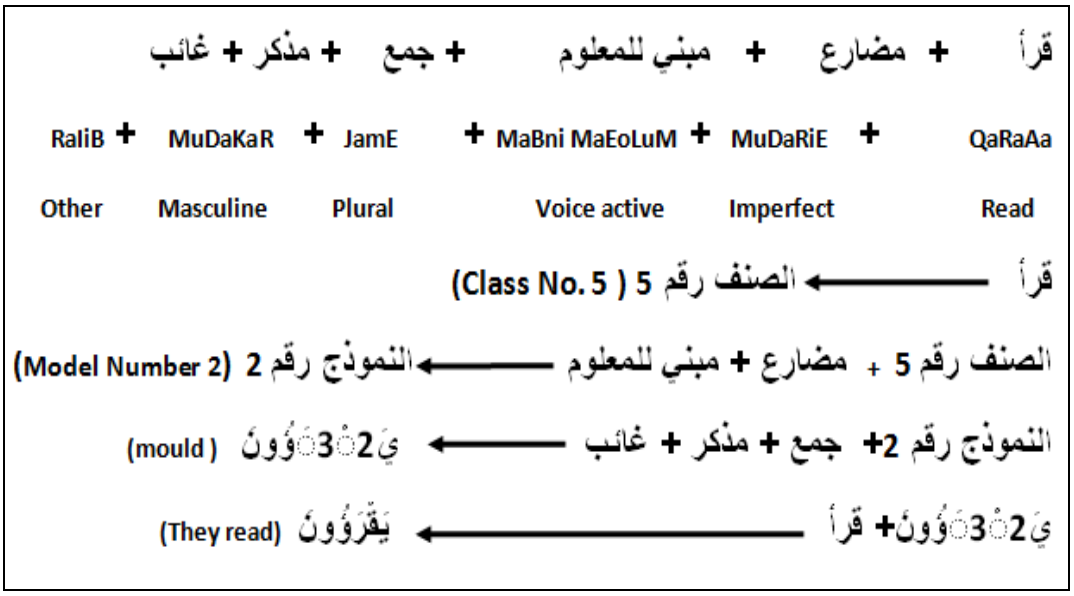

Figure 4. Obtaining the mould

2.1.3. MIXED METHOD

We apply first the pre-established model method. Then we apply the successive transformations method on the result of the first step (See figure 5).

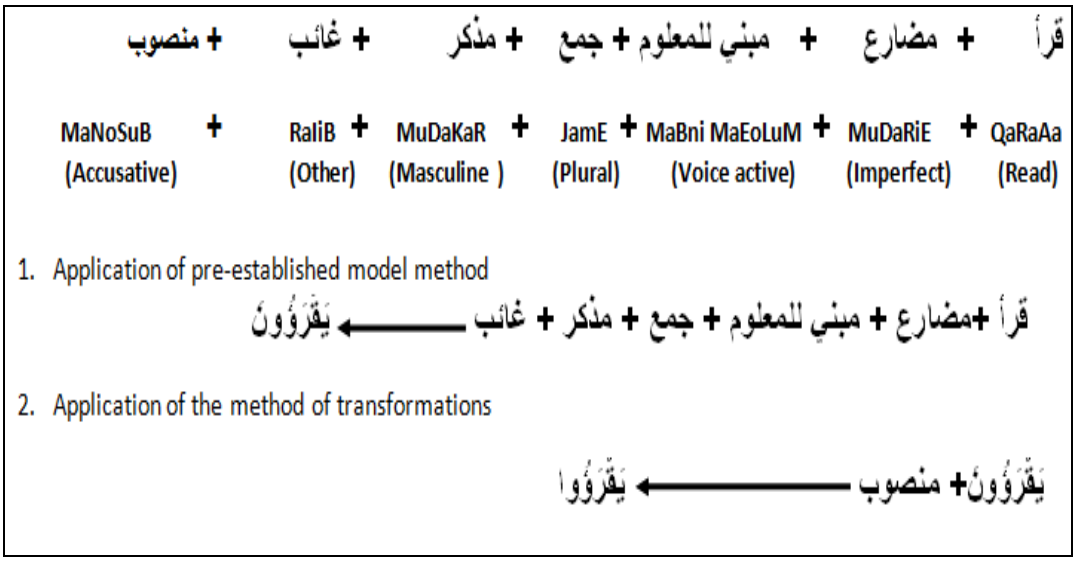

Figure 5. The final word is يَقْرَؤُوا (YAQoRaAu, They read)

3. CONJUGATOR OF VERBS [5]

The method adopted for conjugation of the verb is the mixed method.

Thus, the conjugation operation of a verb proceeds in six steps:

- Determining the verb class (see Figure. 6) by consulting the lexicon t of triliteral verbs or make a treatment based on the length of the verb and the location of certain consonants.

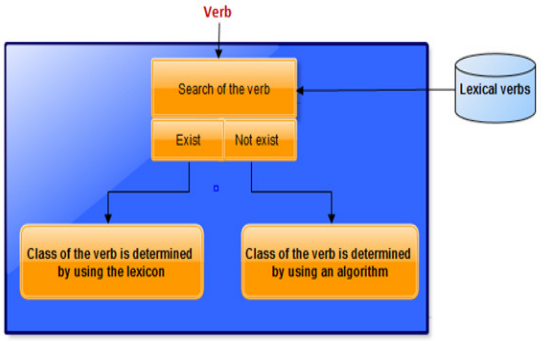

Figure 6. Determining of the verb class

- The adequate model is determined from the triple (class tense, time, voice). (see Figure 7).

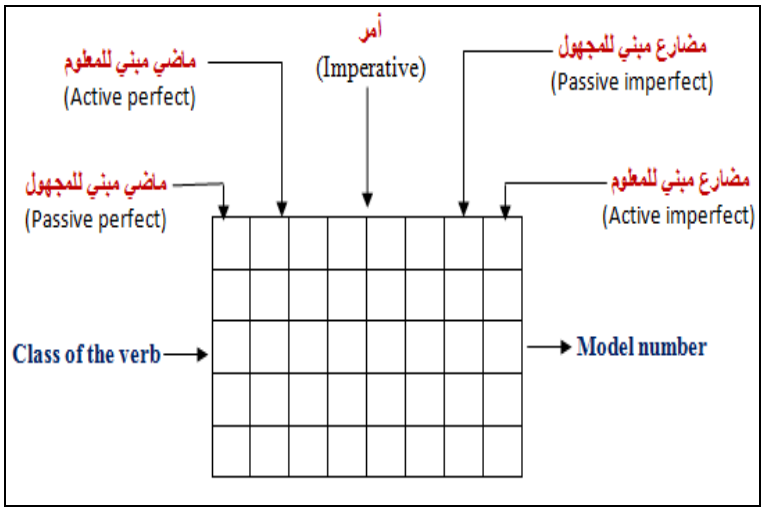

Figure 7. Determining the model.

- The triple (person, number, gender) and the model number allow to extract the desired mould (see Figure 8).

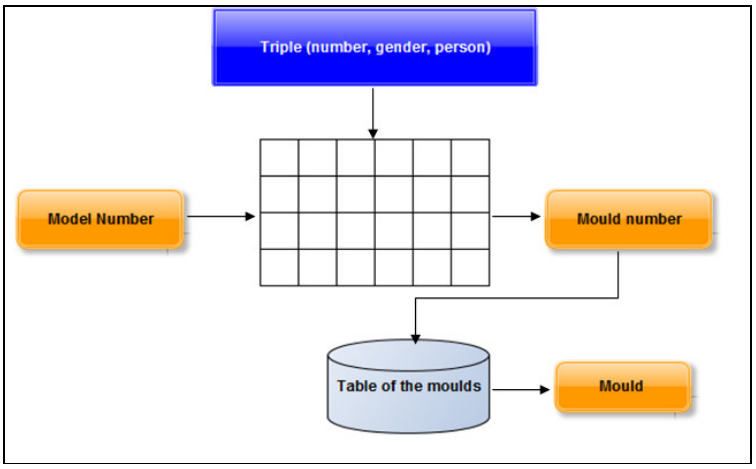

Figure 8. Determination of the mould

- Substitution operation consist to replace the mould numbers with the corresponding consonants of the verb as shown in the example of table 1

Table 1. The substitution operation**.**

Verb	**Mould**	**Substitution**
دَخَلَ (DaKaLa, Enter)	1ُ2ْ3ِ	يَدْخُلُ (YaDoKuLu,Enter)

- Application of the transformation rules (T.R) to deal the case (Apocope, accusative, insistence) of imperfect tense.

For each case, we define the actions to perform on the consonants and vowels of the verb. Table 2 shows an example of the transformations actions:

Table 2. Some transformations actions.

Action number	**Action**	**Coding of actions**
1	Change the last character of the vowel part by "ـَ"	M
2	remove the last character of the consonant part	S
3	remove the last character of the vowel part	S
4	Add the character 'ا' to the last position	A

Table 3 is an example of application of the transformation rules.

Table 3. Application of the RT.

Rule number	Transformations actions	Example (Before)	Example (After)
1	1	يَعْلَمُ	يَعْلَمَ
3	2,3	يَعْلَمَانِ	يَعْلَمَا
4	2,3,4	يَعْلَمُونَ	يَعْلَمُوا

- Spellchecking: This step involves performing the possible corrections on the verb, based on the lexicon of the corresponding anomalies and corrections; an example is giv-en in table 4.

Table 4. Anomaly and correction.

Verb anomaly	Verb after correcting
أأَمَنُ	آمَنَ

4. IMPLEMENTATION AND TESTING OF OUR CONJUGATOR

We realised our conjuguator based on the rules outlined in the previous sections. It consists of four modules (see Figure 9).

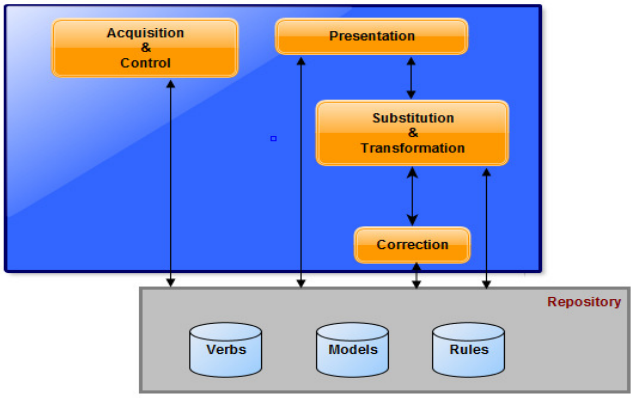

Figure 9. Conjugator modules

- The presentation module enables communication with the user, which can be a person or an application in which our conjugator can be integrated. This module makes the necessary checks on the verb to conjugate and determines the pattern to apply.
- The substitution and transformation module enables to apply the transformation rules on the pattern determined by the previous module.
- The correction module applies correction rules on the result provided by the transformation module to obtain the final shape.
- The acquisition module and control will be used by a linguist to feed the repository by the verbs, the models and the correction rules missing.

The repository of our conjugutor contains all relevant data: verbs, models and correction rules. These data can be stored in several forms (relational DB, XML, Jason ...).

We implemented our conjugator, in its first version, as an Android mobile application using the Java language for coding, the DBMS SQLite for managing the repository and XML to generate mobile interfaces. The mobile application is open source and can be downloaded from the RIME search team web site (http://rime.emi.ac.ma/arabic_conjugator/conjugator.apk).

Figure 10 shows the interface that allows the user to enter the verb, choose the aspect, the voice, and the mode of conjugation.

Figure 10. Data entry

These data will be recovered by the presentation module that will determine the model to apply. The transformation rules for the latter will be applied by the substitution module whose result will be processed by the correction module. The final result obtained will be communicated to the user (see Figure 11).

Figure 11. Conjugation of the verb 'كتب' at the accomplied tense

The current repository of our conjugator contains:

Table 5. Statistics.

Arabic term	**Number**
Roots	6413
Triliteral roots	5635
Roots quadrilitaires	592
Canonical schemas	199
Irreducible trilitaires verbs (مجردة)	6138
Reducible trilitaires verbs (مزيدة)	11832
Irreducible quadrilitaires verbs (مجردة)	452
Reducible quadrilitaire verbs (مزيدة)	245

5. GENERATION OF TEXTUAL WORDS

In the previous section, we used our conjugator as core of a mobile application.

In this section, we use our conjugator to generate textual words using the lexicon of verbs noted M0. These textual words will enrich the dictionary that can be exploited for morphological analysis.

5.1. GENERATION OF INFLECTED FORMS

The Arabic words are built or bending. The words built have an invariant form whatever their position in the text; this is the case of tools words (prepositions, conjunctions, etc...).

The bending words have a termination that changes depending on the case: nominative, accusative, apocope, etc…

We try to generate the inflected forms (lexical kernel of the textual word), noted M0C, for each verb in the lexicon M0. The principle of generation processing of inflected forms is as follows:
M0 is the set of verbs

M0 = v1, v2,...

```
For each m in M0
    if  m.VALG =' imperfect' then
      + Inflected_Form1 ← Generate_Accusative(m)
      + Inflected_Form2 ← Generate_ Apocope (m)
        + Add(Inflected_Form1, M0C) {to add Inflected_Form1 to the lexicon M0C}
        + Add(Inflected_Form2, M0C) {to add Inflected_Form1 to the lexicon M0C}
    End if
End for
```

Where: VALG = Grammatical value of the verb (perfect, imperfect, imperative)

5.2. GENERATION OF TEXTUAL WORDS

The Lexicons M0, M0C represent respectively the lexical words and textual words not having prefixes and suffixes. Arabic prefixes are sets of letters attached to the beginning of the lexical word and written as part of it (for example, prepositions ‘بِ’ ‘Bi’ ‘with’ , ‘لِ’ , ‘Li’, ‘for’, ‘كَ’ ‘Ka’, ‘like’ and conjunctions ‘وَ’ ‘Wa’, ‘and’ , ‘فَ’ ‘Fa’, ‘and’, the determiner 'ال,' 'l aL', 'the') , while suffixes are sets of letters and pronouns attached to the end of the word and written as part of (for example, the object pronoun ‘هم‘ ‘hum‘ , ‘them’) [6].

5.2.1. CONCEPT OF MICRO-SYNTAX [2][4] [8]

We have defined a micro-syntax that represents the set of rules for arranging correctly the attachment of prefixes and suffixes to a word of the lexicons M0 or M0C to generate a textual word.

Formally, we can write: Textual word = $\sum_{i=1}^{n} Pi$ + FS + $\sum_{i=1}^{m} Si$ with:
Pi: is a prefix.
FS: is a simple form that can be a lexical word owned at M0 or M0C.
Si: is a suffix.

The attachment of prefixes and suffixes to words presents four major problems:

1) The sequences of prefixes ($\sum_{i=1}^{n} Pi$) and suffixes ($\sum_{i=1}^{m} Si$) are not arbitrary.
   Example: the sequence ‘وبال’ (‘WaBilaL’, ‘and with the ’) is valid while ‘بوال’ (‘BiWalaL’) is not.
2) Some prefixes or suffixes can be attached only to specified simple forms; for example, the prefix "س" (Sa, will) can be attached to verbal forms at the imperfect, while the prefix "ال" (l aL, the) can be only attached to nominal forms.
3) Some prefixes cannot be glued to a given word, at the same time as certain suffix; for example, we can say "الولد" (laLWaLaDu, the boy) or "ولده" (WaLaDuHu, his son) but we can’t say "الولده"(laLWaLaDuHu).

4) The attachement of certain prefixes or suffixes to a FS is not limited to a simple concatenation, it implies in many cases a change in the FS termination.

Example:

(his meaning) مَغْزَاه ⟵ ـهُ +(meaning)مَغْزَى

5.2.2. GENERATION ALGORITHM OF TEXTUAL WORDS

Let P be the set of simple prefixes and compounds: P = {P1, P2 ...}.

The constitution of these sequences is done manually from any particles that exist in Arabic (preposition, conjunctions ...).

Analogously to the prefixes, we build the set of suffixes: S= {S1, S2 …}.

We define valid combinations between those elements of P and S. It constitutes thus the Pi couples, Sj accompanied by a motion to indicate for which categories of words could be apply (name, verb, etc. ...). This gives what we call a text morphological generation rule (TMR):

Pi , Sj , Motion
Motion which may be of type:
Noun: all word of type noun, adjective are compatible with this Pi couple Sj.
Verb: perfective, imperfective or imperative are compatible with this Pi couple Sj.
Verb imperfective: Only words with the unaccomplished are compatible with this Pi couple Sj.

Examples :

‘وس’ (WaSa, and will) , ‘هم’(Hum, them) , imperfective
‘وب’, (Wabi, and with) , ‘هم’(Him, them) , Noun

In practice, items are handled by classes micro-syntactic. For example, the two previous generation rules will be represented by:

[Wabi] , [Him] , Noun
[WaSa] , [Hum] , imperfect

The proposed generation algorithm of textual words composing lexicon noted M1 is:
M=M0 ∪ M0C is the set of words to process

Let R be the lexicon rules of textual morphological generation: R= {r1,r2,….}
For each m in M

For each r in R
textual_word ← Generate_textual_word(m, r) { apply the rule r to the word m }
Add(textual_word, M1) {to add textual_word to the lexicon M1}
End for
End for

6. CONCLUSIONS

In this paper, we presented the architecture of a system of conjugation of Arab verbs. It operates according to a five-step process: determining the class of verb, determining the model, the substitution operation, applying transformation rules, and spells correcting. The results of the performed tests are very satisfactory.

We used the conjugator to enrich the lexical database by the verbs for morphological analysis using the dictionary-based approach. We proposed also a micro-syntax and an algorithm to generate the textual words to enrich the same lexical database.

We intend to use our conjugator as the core of a learning environment for Arabic and especially conjugation for kids and non-native speakers.

To complete the necessary dictionary for morphological analysis, we intend to generate the textual words for nouns.

REFERENCES

[1] Ali Nabil (1988) "Arabic Language and Computer", [in Arabic], Ta'areeb.
[2] Hlal Yahya, (1979) "Learning Methods for Morphosyntactic Analysis (Experienced in the case of Arabic and French)", thesis doctoral degree, University Paris.
[3] Hlal Yahya, (1987) "Generation from the root and pattern", Conference on the progress of linguistics in Arab countries.
[4] Hlal Yahya, (1990) "Morphology and syntax of the Arabic language", In Proceedings of the Arab School of Science and Technology Applied Arabic Linguistics for Informatics (pp.201).
[5] Mourchid Mohamed, (1999) "Generation Morphological and Applications", Specialty thesis of 3rd round, Mohammed V University in Rabat-Morocco.
[6] Nizar Y. Habash, (2010) Introduction to Arabic Natural Language Processing, Morgan & Claypool Publishers series.
[7] Abdelhadi Soudi, Antal van den BoschNizar, & G¨unter Neumann (2007) Arabic Computational Morphology, Knowledge-based and Empirical Methods, Springer
[8] Joseph Dichy, Ali Farghaly, (2003) "Roots & Patterns vs. Stems plus Grammar-Lexis Specifications: on what basis should a multilingual lexical database centred on Arabic be built?" Workshop on Machine Translation for Semitic Languages: issues and approaches – New Orleans, USA.
[9] Riyad Al-Shalabi, (2005) "Pattern-based Stemmer for Finding Arabic Roots", Information Technology Journal, 4(1): p. 38-43.
[10] A.Yousfi, (2010) "The morphological analysis of Arabic verbs by using the surface patterns", IJCSI International Journal of Computer Science Issues,7(3(11)): p. 33-36.

AUTHORS

M. MOURCHID Doctorate degree in Computer Science in 1999; Assistant Professor at the Computer Science Department at the Faculty of sciences, Ibn Tofail University in Kenitra Morocco; Ongoing research interests: Natural Language Processing, Web Semantic, and information systems.

N. El Faddouli Doctorate degree in Computer Science in 1999; Assistant Professor at the Computer Science Department at the Mohammadia School of Engineers (EMI); 15 recent publications papers between 2007 and 2015; Ongoing research interests: e-learning, Big Da ta, Natural Language Processing, and information systems. Language Processing, and information systems

S. AMALI Doctorate degree in Computer Science in 1999; Assistant Professor at the Computer Science Department at the Moulay Ismaïl University; Ongoing research interests: e-learning, Natural Language Processing, and information systems.