Journal of Information Engineering and Applications
ISSN 2224-5782 (print) ISSN 2225-0506 (online)
Vol.6, No.6, 2016 

# Resolving Inflected Text Structures Irregularities Using Rule-Based Models

Abid Thyab Al Ajeeli
Higher Education Committee, C.O.R., Iraq

**Abstract**
In this paper a model, for natural language inflected irregular text structure, is developed in order to automatically be able to derive stems from given text words. The proposed system is modeled in away so that it has the ability to act in two ways forward and backword which is called bi-directional Techniques. It can deduce morphemes from inflected words and, at the same time, can build inflected words from stems. The proposed system is developed and built using first-order logic techniques.The Proposed rule-based model will help researchers to do more investigation and works on multiligual applications that help facilitate many applications in our real life. Those applications can cover topics ranging from medical diagnosis systems, machine translation,…, to e-government entities through teaching expository text structure to facilitate reading comprehension. The proposed model be able learn how to extract rules from information by applying logic programming techniques to natural language data.
**Keywords:**syntax Analysis, Irregular plurals, rule-based, bi-directional, Inflected words, stems, finite atomaton

## 1. Introduction

The presence of ambiguities in natural languages is the source of a number of particularly complex and hard to resolve problems. These ambiguities, which occur at various different levels of language structures, often are not explicitly apparent to humans, for whom the correct interpretation may seem be very obvious. This is due to the very large capacity that humans have for integrating a sentence into the context of its utterance. Humans can extract, from the context of a sentence in a subconscious manner, the information necessary to resolve the ambiguity. It is an entirely different matter when it comes to creating finite automata machines for analyzing sentences.

The immediate and general context, as well as what might be called general knowledge or common sense, effectively represent the knowledge base generally required to perform this task. However, it is difficult to implement this common scense in a natural language processing system because we do not know enough about how to structure such a knowledge base.

the amount of knowledge that might be used to resolve any given ambiguity might be huge and incontrollable or cannot be completley identified ( Indurkhya & Damerau, 2010), Furthermore, the range of such data is often tied to factors such as uncertainty, prior beliefs, or even preferences (Gal et al. 1991) which are hard to understand and analyse.

Sycology liguistics researchers ( Treiman et al. 2003) suggest that comprehenders must map the spoken or written input onto entries in the mental lexicon and must generate various levels of syntactic, semantic, and conceptual structure. In language production, people are faced with the converse problem. They must map from a conceptual structure to words and their elements. people usually produce single words and then turn to the production of longer utterances. Works on computerized rule-based models need to use the same approach that simulate sycology liguistics thinkers.

Arabic language is the 5th spoken language in the world, with more than 300 million speakers as the mother tongue, and more than 120 million as a second language.
The alpabetic of the Arabic language is made up of 28 letters :

أ ب ت ث ج ح خ د ذ ر ز س ش ص ض ط ظ ع غ ف ق ك ل م ن ه و ي

The vowels in Arabic are three letters: (أ), (ي) and (و). Sometimes Arabic Natural Language researchers consider alHamza (ء), Lam-Alef (لا), and Alef Maqsoura (ى) either three letters by its own or can be one block that can be deduced from Alf (أ).

Arabic has also diatric forms (alTashkeel), which works equivalent to vowel letters in English. Al Tashkeel generates letters in new forms:

**Fatha : ◌َ , works as A in English, بَ pronounced BA.**
**Dhumma : ◌ُ works like O in English, بُ pronounced BO.**
**Kasra : ◌ِ works like i or e in English, so بِ pronounced Be.**
**Sukon : ◌ْ works without Tashkeel, بْ pronounced b as in the word rub**
**Shuddah : ◌ّ it is combination of Sukon and Fatha, Dhumma, or Kasra, works as emphasizer to pronounce the letter, so any letter with Shudda is pronounced twice, so بّ is pronounced like b in rubbing.**
**Tanween :ٍ ً ٌ which is pronouncing a letter with a Noun, so لٌ is pronounced Lon**

Unlike English and Latin Languages, Arabic writing orientation is from Right to Left.

Journal of Information Engineering and Applications www.iiste.org
ISSN 2224-5782 (print) ISSN 2225-0506 (online)
Vol.6, No.6, 2016

Words in Arabic are classified as one of three classes: Noun, Verb, or Conjunction. Arabic Nouns comes in three genders, Masculine, and Feminine. Arabic nouns comes in three form : **Single** مفرد, **Daul** مثنى , and Plural **جمع**. Nouns comes in three grammatical cases: **nominative (الرفع), Accusative (النصب), and Genitive (مجرور).**

Although Arabic has its own grammar in forming Dual, and Plural, but some nouns have special cases which is Maqsour مقصور. Maqsour is a name that is ending with vowel letter ى. This special case is treated differently than the standard ones. Depending to the noun case, Arabic verbs come in a way compatible with subject (الفاعل). If the subject is single, the verb is single, and if the subject is active participle dual or plural the verb follows accordingly. Arabic Sentence could start with Name and such case will be called nominal sentence, otherwise it is called verbal sentence (Gibson 1998). The following graph shows part of Arabic text structures.

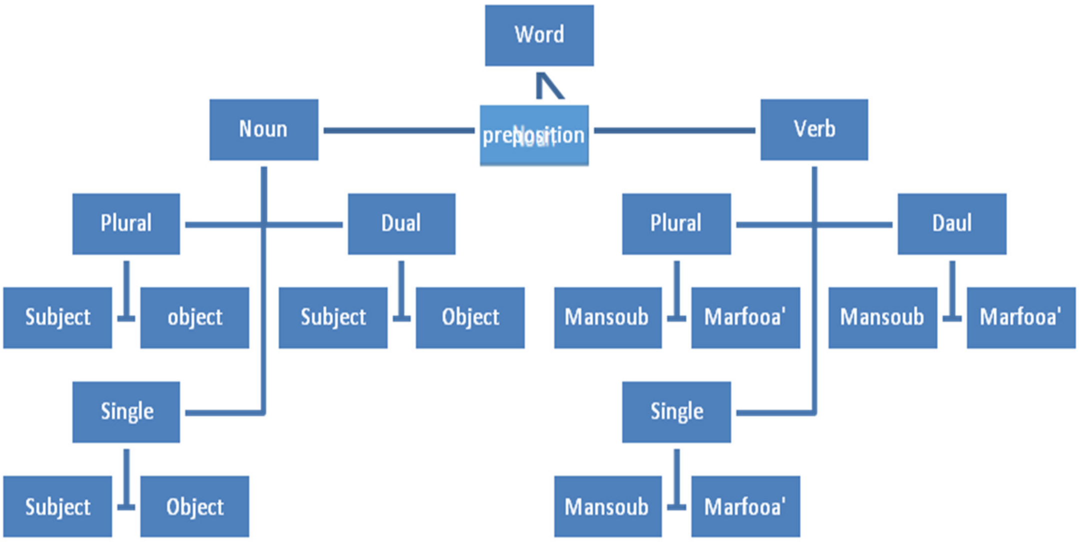


Fig (1): Arabic words Structure

words in Arabic language can be classified as shown in figure1 above. For more details, investigation, and understanding of arabic text structures will be found in the following references (Elgibali & El-Said M. Badawi 1995; Versteegh 1997; *Ryding 2005;* Drissner 2015)

## 2. Related Work

Over the past three decades research in Arabic morphology has been a popular field in the Arab World due to the importance of morphological components in any language model based on Arabic structures. In the west, Kiraz (2005) of Bell Laboratories states that "Arabic attracts attention because of its intriguing morphological systems, viz., the root-and-pattern system". For example, Abboud and McCarus (1986) have edited two volumes on Modern Standard Arabic (often abbreviated MSA). MSA is effectively classical Arabic, the language in which the Qoran was revealed.

The vocabulary of Modern Standard Arabic has been expanded, of course, to include words for modern concepts, but even so, efforts are made to keep the new vocabulary within the "rules" of word formation of classical Arabic.

In 1996, the Xerox Research in Europe produced a morphological analyzer for MAS ( Beesley, 2016). In 1997, a Java-applet interface was added to allow testing on the internet using standard Arabic orthography. The analyzer-generator systems were based on dictionaries ( Buckbwalter 1990, Beesley 1990) using finite-state technology (Beesley & Karttunen 2003).. The system was intended to serve as a pedagogical aid, a comprehension-assistance tool, and as components in larger natural language processing systems (Beesley, 2005)

The most interesting and famous aspect of the Arabic grammar is the three-letter root system, best described by an example. The three letters k-t-b, for example, carry the basic meaning "write." Various combinations of the letters—always, however, in the k-t-b order—with vowels and other consonants produce words that are variants on the basic meaning "write." For example ( Bateson, 1967):

**'book' [kitaab] كتاب 'desk', 'office' [maktab] مكتب**
**'books' [kutub] كُتُب 'scribe', 'writer' [kaatib] كاتب**
**'he wrote' [kataba] كتب 'written' [maktuub] مكتوب**

Another interesting aspect of Arabic grammar that also shows up in English, is [al] (or [il] in some dialects), which is the definite article, the Arabic word for "the." It is prefixed to the following word, and, depending on what consonant that word starts with, the word "the" ( Bakalla, 1994) may be dropped and the first consonant doubled.

Many English words which have been adopted from Arabic words are still having the definite article "al" attached the them—for example, *alkali, alcohol, alchemy, algorithm, algebra,* and *almanac.*

The mechanism used to generate individual words must be based on word morphology. Morphology systems can be used to decompose words into word stems and word affixes. Morphology is the study of the internal structure of words. Derivational morphology. How new words are created from existing words. It is a learning extraction rules than can apply inductive logic techniques to Arabic Language data ( Kuusik & Lind 2012).

In addition, such systems can be used to specify mood, genders, numbers, and persons. A study of Arabic morphology in ( Al-Sadany & Hashish 1989) has focused on the rules that cause the inflected forms, which should be derived from roots. Another study by Mahgoub and Hashish (2016) was based on the concurrence patterns. The study was aimed to construct a matrix representation of the possible inflectional forms of Arabic words.

Many systems have been designed to address this issue. For example, Hegazi, & El-Sharkawi (1986) developed a system that be able to detect roots in Arabic structures. This system is used to detect and correct mistakes in spelling and vowelization. Another example is a morphological analysis and generation system that is used to examine input words ffrom different text types and attempts to find possible analyses ( Salib & Al-Dannan 1989, Aitken 2002 ).

One of the major breakthroughs in the field of morphology was the two-level morphology. It is a general computational model for word-form recognition and generation (Koskenniemi 1983 ). Lauri Karttunen and others produced a LISP implementation of the two-level morphology and it is named KIMMO ( Karttunen et al. 1992).

Regarding the irregular plural, it was analyzed by Soudi et al. (2001) and the focus was on the "broken" plural. In Arabic texts there are two types of plural, one is called a standard that one can use rule-based model to predict but the broken or non standard one is not easily predictable.

The research outlined the problems in computational treatment of Arabic Morphology. The article considered solutions to the problems by providing the broken plural pattern in a lexicon and then a series of morphological rules operate on the singular noun to generate the plural noun in the morphological component. A second solution was to provide the singular and the plural stems in the lexicon and then add the required affixes in morphology. The two approaches have been studied in more detail in (Zwicky 1986).

## 3. The Rule-Based Model

One approach of reducing ambiguities is to study regular and irregular words of a language. This study aids in determining the context of a sentence. For example, is the sentence concerned with singular, dual, regular (sound) plurals, or with irregular plurals. Studying singular and plural words is also useful in facilitating automatic translation.

We start this study by proposing a hypothesis that w′ is the plural word of w, which can be confirmed by the fact that there is a dictionary for w. Mathematically, this can be modeled as follows.

Null hypothesis H0: w′ plural of w.

**H1: w′ is not the plural of w**

The problem with Arabic language compared to English language is that the Arabic words are heavily inflectional and they can have prefixes, infixes, and/or suffixes. Irregular plural words are considerd as an exception and can be modeled using logic facts as in the form below.

```
Plural (child, children):- !.
Plural (mouse, mice):- !.
.      .      .
Plural (sheep, sheep):- !.
```

Enhancement can be made to spell checker software, grammar and style checkers, understandability, checking textual consistency, reverse engineering when the patterns of nouns are known.

3.1. General Cases

Add “s” to singular words as follows:

- Words ending in “y”, change “y” to “i” and add “es”, except after a vowel. Example fly → flies, but play → plays.
- Words ending in ch, sh, o, s, x, or z add “es”.
- Double consonant after a short vowel(Below are examples of using rules to building variations of words), e.g.
  hop → hopping (note, short vowels are generally single vowel letters before certain consonants p, b, t, d, g, m, n)
- Replace “e” in CVCing where C stands for consonant and V is for vowel. For example, hate → hating, chase → chasing

## 4. Arabic Sound Plurals

As in any language, Arabic plural words have two forms either sound (regular) or irregular (broken plural). Arabic plural words are unlike English plurals. In the Arabic language one must distinguish the gender. The pattern of regular plurals may depend on the position of the word itself whether the word is a subject, object, or a prepositional phrase.

We can summarize regular plurals, for masculine, as a function of first-order predicate logic.

$\forall w$ f(w) → singular + [و] + [ن] ; if W's position is a subject or predicate
$\forall w'$ f(w′) → singular + [ي] + [ن] ; otherwise

The first logic rule states that a sound plural for masculine , if its position is a subject or predicate, can be constructed by concatenating the singular noun to "و" and "ن" , or vise versa by extracting the singular from the sound plural by removing the last two letters. This facility can be implemented efficiently using the concepts of first-order logic unification mechanisms. Otherwise, using the same argument, the second rule can be implemented by concatenating the singular noun to "ي" and "ن".

Regular plurals for feminine can be modeled as a function of first-order predicate logic as follows:

$\forall y$ f(y) → singular + [ ا ] + [ت] ; if Y's position is a subject or predicate
$\forall y'$ f(y′) → singular + [ ا ] + [ت] ; otherwise

sound feminine plurals can be constructed by concatenating "ا" and "ت" to the singular.

The finite-machine automata below shows how the above rules can be used to derive any morpheme when arcs are traversed in either direction to generate regular plurals for masculine and feminine. For example, in figure 2, the path through the states 0 → 1 → 2 → 3 → 4 → 5 can generate regular plurals for masculine. When we are in state 1 and we read a, then this would mean the regular plural is a subject or a predicate. Figure 2 shows how regular plurals can be generated for masculine and feminine.

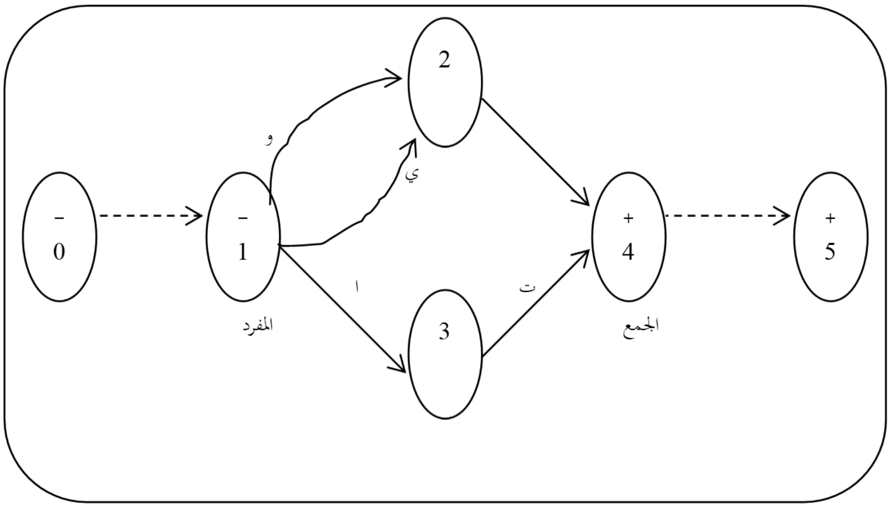


Figure 2: Plurals for masculine and feminine

The machine in figure 2 illustrates how a program can construct or unify plural or singular nouns for both masculine and feminine. The broken lines in figure 3 above may be used to detect prefixes and suffixes associated with irregular plural nouns.

### *4.1 Irregular Plurals (Broken Plurals)*

Irregular plurals for names (words), in Arabic, have many forms. The main forms or patterns (templates) can be characterized as in the table I below.

**Table I: Irregular templates for nouns**

| مثال Example | الوزنTemplate | الجذرٌRoot | | مثال Example | الوزنTemplate | الجذرٌRoot |
|---|---|---|---|---|---|---|
| قَصعَة قِصاع | فِعال | فَعلَة | | حمل أحمال | أفعال | فِعل |
| رَقَبَة رقاب | فِعال | فَعَلَة | | قُفل أقفال | أفعال | فُعل |
| مَعِدة مَعِد | فَعِل | فَعِلة | | عِنَب أعناب | أفعال | فِعَل |
| غَزَال غِزلان | فِعلان | فَعَال | | عُنُق أعناق | أفعال | فُعُل |
| زَمَان أزمِنة | أفعِلة | فَعَال | | خَصم خُصُوم | فُعُول | فَعل |
| غَزَال غِزلان | فِعلان | فَعَال | | نَصل نِصال | فِعال | فَعل |
| قَذَال قُذُل | فُعُل | فَعَال | | نَصل أنصُل | أفعُل | فَعل |
| غُلام غِلمان | فِعلان | فُعال | | جَمَل جِمال | فِعال | فَعَل |
| غُلام أغلِمَة | أفعِلَة | فُعَال | | حَمَل حُملان | فُعلان | فَعَل |
| إصبَع أصابِع | أفاعِل | إفعَل | | رُمح رماح | فِعال | فُعل |
| دِرهَم دَراهِم | فَعالِل | فِعلَل | | حِمل حُمول | فُعول | فِعل |
| صَحيفة صَحَائف | فَعَلاء | فَعيلة | | جُرَذ جِرذان | فِعلان | فُعَل |
| رَغيف رُغفان | فُعلان | فَعيل | | أنثى إناث | فِعَال | فُعلى |
| عَمُود أعمِدَة | أفعِلَة | فَعُول | | صَحراء صَحارى | فَعالى | فَعلاء |
| غَزَال غِزلان | فِعلان | فَعَال | | نَصل نِصال | فِعال | فَعل |
| عَمود أعمِدَة | أفعِلَة | فَعُول | | مَرماة مَرامي | فَعالي | فَعلاة |
| عَمُود أعمِدَة | فُعُل | فَعُول | | سِعلاة سَعالي | فَعالي | فِعلاة |
| قِرطاس قَراطيس | فَعاليل | فِعلال | | عِبرَة عِبر | فِعَل | فِعلَة |
| سفرجَل سَفارِج | سَفاعِل | سفعَلَل | | صُورَة صُوَر | فُعَل | فُعلَة |

Table I above is not an exhaustive but it covers majority of templates and demonstrates the possibility of counting majority of arabic templates. All templates can be programmed as **first-order logic**, in which each entry the above table is considered a rule by itself.

There are three main methods for constructing plurals in arabic.

- **Masculine regular plural:** this method is straight forward and has the form append the sequence of letters "و" and "ن" to the singular word if the position of the singular is subject or predicate. Examples are:

| | |
|---|---|
| **مدرس** | **مدرسون** |
| **معلم** | **معلمون** |
| **مشاهد** | **مشاهدون** |
| **منتصر** | **منتصرون** |

- **Feminine regular plural:** This form is constructed by appending "ا" and "ت" to the singular feminine word. Examples are:

| | |
|---|---|
| **جامعة** | **جامعات** |
| **جماد** | **جمادات** |
| **حيوان** | **حيوانات** |

- **Irregular Plural:** When regular masculine or feminine cannot be constructed then an irregular plural will be constructed but the problem with irregular plurals are the multiple forms associated with those constructions. Examples are shown table II:

Table II: Irregular Multiple Forms

| Singular | Plural |
|---|---|
| **بلد** | **بلدان، بلاد** |
| **ألف** | **ألوف، آلاف** |
| **زمان** | **أزمنة، أزمان، أزمن** |
| **ضلع** | **أضلاع، ضلوع** |
| **غصن** | **أغصان، غصون** |
| **طالب** | **طلبة، طلاب** |
| **أجمة** | **أجمات، أجم، آجام، إجام، أجم** |

Each singular in the table above has more than one plural forms and they are frequently used in the daily life. This variety in the construction makes the Arabic language a peotry language. Some singulars may have more than one semantics (Sadan 2012) ). From type, its plural and its semantic can be deduced. Table III shows a number of examples :

Table III: Multiple Plural Forms

| Singular | Plural |
|---|---|
| بحر | أبحر، بحار، بحور (للشعر) |
| بيت | بيوت، أبيات (للشعر) |
| ثعبان | ثعابين، ثعب( مجرى الماء في الوادي) |
| جبهة | جباه (للوجه)، جبهات |
| جفن | جفان، جفنات، جفون (للعين) |
| حاجب | حواجب (للعين)، حجاب (للأمير) |
| جواد | جياد، جود، أجواد ( للكرماء) |
| أسود | سود (الألوان)، أساود (العظيمة من الحيات) |
| | |

In Arabic literature, the sound plural is used for small numbers and irregular plurals and plurals of the plural are used for exageration and for large quantities. There are rules in general for using plurals and singulars. Singulars are used for nouns with specific properties, processes, names, and geographical regions.

## 5. The Proposed Model

The proposed model is based on one variation or finite automata mechanisms called modified Mealy machine. The machine is similar to finite automata approach except that alphabet input is changed while we are traveling along the edges. A Mealy machine is a collection of four items (Q, Σ, δ, Γ) ( Hopcroft et al. 2001, Bechhofer & Goble 2001).

- Q is a finite set of states q0, q1, q2, … where q0 is designated as the start state.
- Σ is an alphabet letters {أ، ب، ج ...} for formatting input strings.
- Γ is an alphabet of output characters, which is not necessary a subset of Σ. In our case, Γ is equivalent to Σ.
- δ represents rules that direct transitions between states, where states are connected by edges. Each edge is labeled with a compound symbol of the form i/o where i ∈ Σ and o ∈ Γ. Figure 1 shows a modified Mealy machine. For example, if we traverse the edges 0 → 1→ 4→ 11→ 17→ 23→ n this path changes any pattern of type (فعيل) into patterns of type (فعلان). I.e. (رغيف to رغفان).

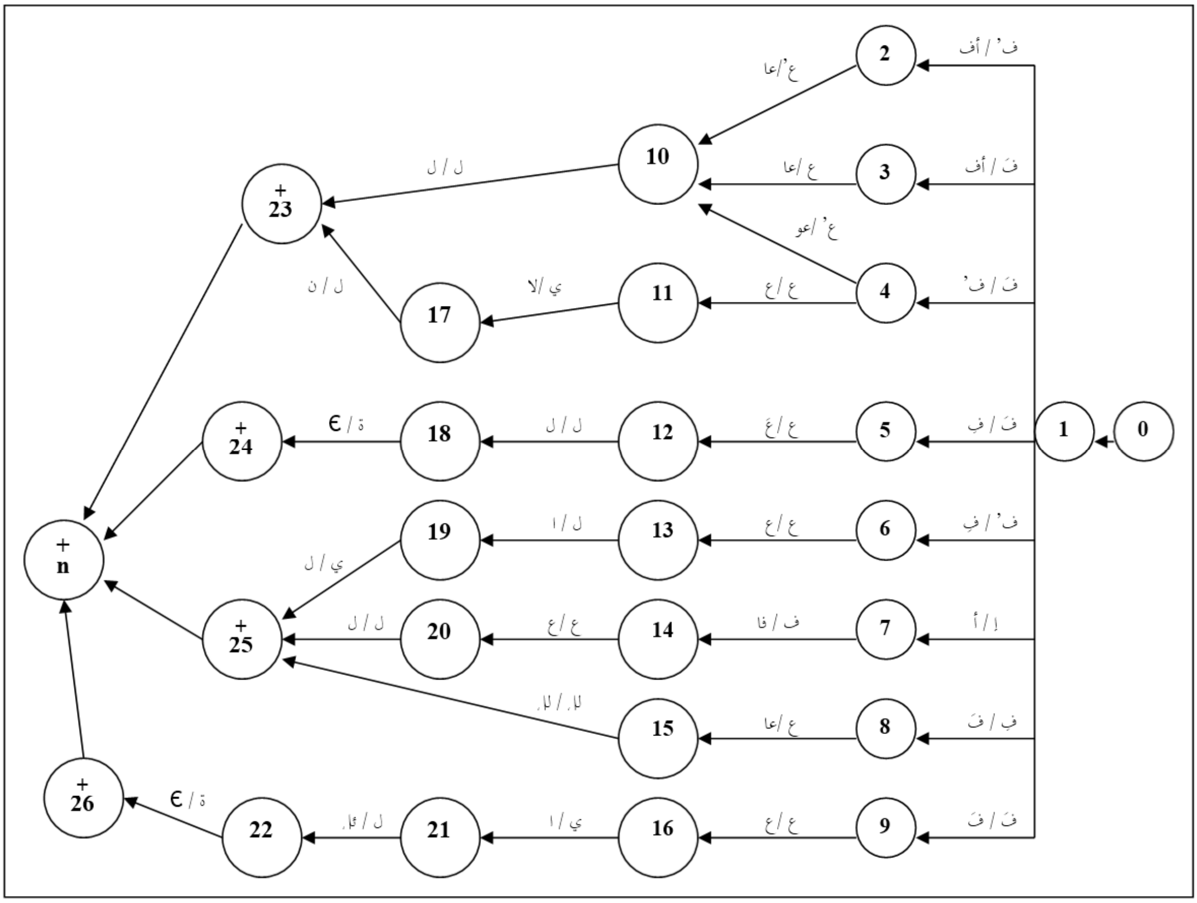


**Figure 3: Modified Mealey Machine**

The path 0 → 1 → 9 → 16 → 21 → 22 → 26 → n provides a mechanism of transforming patterns of type (فَعيلة) to patterns of type (فَعائل), as in (صحيفة to صحائف).

Irregular plurals for adjectives have also many different forms. The main forms or patterns are recognized in table VI below.

**Table VI: Irregular Plurals for Adjective templates**

| مثال Example | الوزنTemplate | الجذرٌRoot | | مثال Example | الوزنTemplate | الجذرٌRoot |
|---|---|---|---|---|---|---|
| ساجِد سُجّد | فُعّل | فاعِل | | صَعب صِعاب | فِعَال | فَعل |
| رامٍ رُمَاة | فُعَلة | فاعِل | | جُنب أجناب | أفعال | فُعل |
| بائِع بَاعَة | فَعلَة | فاعِل | | يَقِظ أيقاظ | أفعال | فَعِل |
| جاهِل جُهَلاء | فُعَلاء | فاعِل | | بَطَل أبطال | أفعال | فَعَل |
| أسوَد سُود | فُعل | أفعَل | | غَيُور غُيُر | فُعُل | فَعُول |
| أكبَر أكابِر | أفاعِل | أفعَل | | عَدُو أعداء | أفعال | فَعُل |
| أعمى عُميان | فُعلان | أفعَل | | جَبَان جُبَناء | فُعَلاء | فَعَال |
| … | … | … | | … | … | … |
| عَذراء عَذاري | فَعالي | فَعلاء | | قتيل قَتلى | أفعال | فَعيل |
| عَاذِلة عُذّل | فُعّل | فَاعِلة | | كَريم كُرَماء | فُعَلاء | فَعيل |
| كَرِيمة كِرَام | فِعَال | فَعِيلة | | شَحيح أشِحَة | أفعِلَة | فَعيل |

Table VI is also not exhaustive. A sample of Prolog codes for the proposed system that automatically can derive irreqular plural for adjective templates. Prolog code is listed in Appendix A..

## 6. Conclusions

The research in this paper aims to deduce stems from irregular plural words. The rule is based on first-order logic. The mechanism used is based on a variation of finite automata called modified Mealy machine. The machine is similar to finite automata approach except that alphabet input is changed to its expected forms while we travel along the edges.

The rerearch aimed to generate a scheme framework for dealing with multilingual text irregularities as well as text generation. Although the works established the bases of text structure and analysis, I suggest to study irregularities in more depth to extract a representation of the meaning of the text. From this representation, a new text is generated by using a text rule-based model and action rules. This work can be extended using functional grammar of text structures to include, in addition to word analysis and sentence analysis, construction of text patterns, reference solving and inference.

The rule-based model can also be extended to record different types of objects and actions associated with the extraction of the objects including processes of sentence generation, action rules, semantic network, syntactical information, object introduced by discourse, affirmations on those objects and links between those affirmations.

The proposed system is written in PDP. Prolog is chosen because it is more suitable for natural language processing. Thus,It is appropriate for unification and query presentation.

## 7. Future Works

Although the research in this paper is aimed to deal with irregularities of natural language, I found the necessity to outline future researc areas that need to be opened for further discussions and enhancements. There are debates concerning issues about the balance between computation and storage arise in all of these domains. Clearly, a good deal of information must be stored in the computerizex lexicon, including the forms of irregular verbs. It is an approach to simulate human lexicon. One need to ask are forms that could in principle be derived by rule (e.g., يمشي walked) computed each time they are heard or said, are they stored as ready-made units, or are both procedures (Treiman et al.2003). Such issues have been debated in both the comprehension and production literatures, and will be important topics for future research. Another broad-based debate is that between interactive and modular views. As we have seen, there is no clear resolution to this debate. It has been difficult to determine whether there is a syntactic component in language production that operates independently of conceptual and phonological factors. Similarly, comprehension researchers have found it difficult to determine whether an initial analysis that considers a restricted range of information is followed by a later and broader process, or whether a wide range of linguistic and nonlinguistic information is involved from the start.

The debate between rule-based and statistical views of language processing provides a good example of how theoretical tensions and the research they engender has furthered progress in computerized linguistics. Statistical approaches, as embodied in connectionist models, have served the field well by emphasizing that certain aspects of language involve probabilistic patterns. People appear to pick up and use statistical information of this kind in areas of language processing. In such cases, we do well to go beyond the notion of all-or-none rules. For example, nouns verbs, and adjectives in Arabic always agree in gender. Our ability to follow such patterns, suggests that Chomsky's notion of language as an internalized system of rules still has an important place to perform in views of language processing (Clark 1996).

**References**

Abboud, P. F., & McCarus (Eds.) (1986), "Elementary Modern Standard Arabic", Vo. 1 & 2 ( 2nd. Edition), Cambridge University Press.

Aitken, J. S. ( 2002), Learning Information Extraction Rules: An Inductive Logic Programming Approach, Proceedings of the 15th european Conference on Artificial Intelligence, ECAI2002, Loyon, France, July 2002.

Bakalla, M. H. (1994), "Arab and Persian Phonetics". In: Asher, R. E., editor-in-chief. *The Encyclopedia of Languages and Linguistics*. Vol. 1, pp. 187-191. Oxford: Pergamon Press.

Bateson, M. C. ( 1967), Arabic Language Handbook. Washington: Center for Applied Linguistics.

Bechhofer, S. & Goble, C. (2001), The Saurus Construction Through Knowledge Representation, Data & Knowledge Engineering Journal, 37 (1): 25-45, April 2001.

Beesley, K. R. (1990), Finite-State description of Arabic morphology. In Proceedings of the second Cambridge Conference on Bilingual Computer in Arabic and English, September 5-7.

Beesley, K. R. & Karttunen, L. (2003). Finite-State Morphology, CSLI Publications 2003.

Beesley, K. R. (2016), Finite-State Morphological Analysis and Generation of Arabic at Xerox Research: Status and Plans in 2001, Retrieved from http://citeseer.ist.psu.edu/454175.html, accessed June 2016.

Buckwalter, T. (1990), Lexicographic notation of Arabic noun pattern morphemes and their inflectional features, In Proceedings of the second Cambridge Conference on Bilingual Computer in Arabic and English, September 5-7.

Clark, H.H. (1996). Using language. Cambridge: University Press.

*Drissner, G. (2015), Arabic for Nerds. Berlin, Germany: createspace*

Elgibali and El-Said M. Badawi. *( 1996), Understanding Arabic: Essays in Contemporary Arabic Linguistics in Honor of El-Said M. Badawi*, 1996.

El-Sadany, T. & Hashish M. A. (1989), Arabic Morphological Systems, IBM Systems Journal, 25th Anniversary for Scientific Centers Issue, Vol 28, No. 4, 1989.

Gal, A., Lapalme, G., Siant-Dizier, P. & Somers, H. (1991), "Prolog for Natural Language Processing", John Wiley & Sons Ltd.

Gibson, E. (1998), Linguistic complexity: Locality of syntactic dependencies. Cognition, 68, 1-76.

Hegazi, N. & El-Sharkawi, A. (1986), Natural Arabic Language Processing, Proceeding of the 9th National, Computer Conference and Exhibition, Riyadh, Saudi Arabia, pp.10-5-3, 1986.

Hopcroft, J. E., Motwani, R. & Ullman, J. D. (2001), "Introduction to Automata Theory", Addison-Wesley.

Indurkhya N. & Damerau, F. J. (2010), Hand Book of Natural Language processing, Second Edition, 2010 by Taylor and Francis Group, LLC

Karttunen, L., Kaplan R. & Zaenen A. (1992), Two-level Morphology with composition, Proceedings of the 14th International Conference on Computational Linguistics COLLING-92, Volume I, pp. 141-148, Nantes, France.

Kiraz, G. A. (2016), Arabic Computational Morphology in the West, Bell Laboratories, Retreived from http://citeseer.ist.psu.edu/59459.html, June 2016.

Koskenniemi, K. (1983), Two-level Morphology: a general computational model for word-form recognition and production, University of Helsinki, department of General Linguistics, publications No 11.

Kuusik, R. & Lind, G. (2012), An effective Inductive Learning Algorithm for Extracting Rules, In Proceedings of the 2011 2nd International Congress on Computer Application and Computational Science, Vol. 2., Edited by Gaol, F. L & Nguyen, Q. V., Published by Springer-Verlag Berlin Heidelberg 2012.

Zwicky, A. (1986), The General Case: Basic Form versus Default Form, Berkeley Linguistics Society, 12:305-314.

Mahgoub, H. E. & Hashis M. A. (2016), A Matrix Represenation of the Inflectional Forms of Arabic Words: Astudy of Co-Occurrence Patterns, http://acl.ldc.upenn.edu/, Accessed May 2016.

Motaz, *S. (2010), Arabic Morphological Tools for Text Mining.* Lefke, Cyprus : s.n., 25-26 Nov, 2010. 6th Internation Conference on Electrical and Computer Systems (EECS' 10).

*Ryding, K. C. (2005), A reference grammar of Modern Standard Arabic (6th printing ed.). Cambridge: Cambridge University press.*

Sadan, *A. (2012), the Subjunctive Mood in Arabic Grammatical Thought*, Volume 66 of Studies in Semitic Languages and Linguistics. Leiden: Brill Publishers.

Saliba, B. & Al-Dannan, A. (189), Automatic Morphological Analysis of Arabic: A study of content word analysis" Proceeding of the first Kuwait Computer Conference, Kuwait, Mar, pp. 3-5, 1989.

Soudi, A., Cavalli-Sforza, V. & Jamari, A. (2016), A Computational Lexeme-Based Treatment of Arabic

Morphology. Retreived from http://libra.sfsu.edu/~vcs/ACL_Arabic_Workshop_2001.pdf, June 1, 2016.

Treiman, R., Clifton, C., Jr, Meyer, A. S., & Wurm, L. H. (2003), Language comprehension and production. Comprehensive Handbook of Psychology, Volume 4: Experimental Psychology. New York: John Wiley & Sons, Inc. Pages 527-548. Copyright John Wiley & Sons.

Versteegh, *K. (1997), The Arabic Language* , Edinburgh: Edinburgh University Press, 1997

Weller, K. (2010). Knowledge Representation in the Social Web, Walter de Gruyler GmbH & Co. KG, Berlin/ New York.

**Appendix A: A sample of Prolog Code**

```
Run (List, W_rest1, R):-
        List = [A, " " , C, " " , D|R],        List1 = [A, "ا" , C, D],
        Collect (List1, W_rest1),   write (" جمع تكسير
        Nl, write ( "،" : أصل الفعل هو W_rest1),        !.

% (فالة) باعة : بائع %
run( List, W_rest1, R):-
        List = [A, " " , Cة" , "ا", "|R],        List1 = [A, "ئ", "ا", C],
        Collect( List1, W_rest1),   write( " جمع تكسير
        Nl, write( "،" :أصل الفعل هوW_rest1), !.

% (فالة) باعة : بائع %
run(List, W_rest1, R):
        List = [A, "ا" , Cة" ,"|R],   List1 = [ A, "ئ" , "ا" , C],
        Collect(List1, W_rest1),   write1 = [A, "ئ" , "ا" , C],
        Nl, write("،" : أصل الفعل هو W_rest1),        !.
% (أفعلاء) أصدقاء : صديق %
        List, W_rest1, R):-
        List = ["،"أ A, B, Cء" , "ا" , " |R],      List1 = [ A, B, "ي" ,C],
        Collect(List1, W_rest1),   write(" ,(" جمع تكسير
        Nl, write("," : أصل الفعل هو W_rest1),        !.
% (أفعلاء) أصدقاء : صديق %
run(List, W_rest1, R):-
List = ["،"أ A, B, Cء" ،"ا" ," |R],      List1 = [A, B, "ي", C],
Collect(List1, W_rest1),   write(" ،(" جمع تكسير
Nl, write("،" : أصل الفعل هو W_rest1), nl,        !.

% (أفلة) أزمة : زمام %
run(List, W-rest1, R):-
        List = [","أ A, C ة", "|R],   List1 = [A, C, "ا" , C],
        Collect(List1, W_rest1),   write(" ،(" جمع تكسير
        Nl, write("،" : أصل الفعل هوW_rest1), nl,        !.
% صفات
(فعلى)        قتلى : قتيل
run(List, W_rest1, R):-
        List = [A, B, Cى" ," |R],   List1 = [A, B, "ي" , C],
        Collect(List1, W_rest1),   write(" جمع تكسير
        Nl, write("،" : أصل الفعل هو W_rest1), nl,        !.

% (أفاعل) أصابع : إصبع %
run(List, W_rest1, R):-
        List = ["، "أ A, "ا",B, " ", C|R],        List1 = ["،"إ A, B, " ", C],
        Collect(List1, W_rest1),   write( ،(" جمع تكسير
        Nl, write( "،" : أصل الفعل هوW_rest1), nl        !.

% (فعائل) صحائف : صحيفة %
run(List, W_rest1, R):-
        List = [A, B,"ئ" ،"ا" , D|R],        List1 = [A, B,"ي", D,["ة",
        Collect(List1, W_rest1),   write( " جمع تكسير
```

```
        Nl, write( "،" : أصل الفعل هوW_rest1),nl,        !.

(فعالل) دراهم : درهم
run(List, W_rest1, R):-
        List = [A, B, "ا", C, D|R],  List1 = {A, B, C, D],
        Collect(List1, W_rest1),   write( ،(" جمع تكسير
        Nl, write("،" : أصل الفعل هوW_rest1), nl,        !.

% (أفعلة) أعمدة : عمود ، عماد
% أخمرة : خمار
% أرغفة : رغيف
run(List, W_rest1, R):-
        List = ["،"أ A, B,C"،" |R],  List1 = [A, B,"و",C],
        List2 = [A, B, "ا", C],       List3 = [A,B, "ي", C],
        Collect(List1, W_rest1),    collect(List2, W_rest2),
Collect(List3, W_rest3),    write(" ،(" جمع تكسير
Nl, write("أصل الفعل هو :",W_rest1), nl,write("،" : أصل الفعل هو W_rest2),
Nl,write("،" : أصل الفعل هو W_rest3), !.

% فعال
run\ List, W_rest1 R):-
        List = [A, " ", B, "ا", C|R], List1= {A, B, C],
        List2 = [A, " ", B, C, [ "ى",          List3 = [A, B, C, ["ة",
        List 4 = [A, B, "ي", C],               collect(List1, W_rest1),
        Collect(List2, W_rest2),    collect(List3, W_rest3),
        Collect(List4, W_rest4),    write("  ،(" جمع تكسير
        Nl,write("أصل الفعل هو: ",W_rest1),  nl,write("،" : أصل الفعل هو W_rest2),
        Nl,write("،" : أصل الفعل هوW_rest3),
        Nl,write("أصل الفعل ،" : هو W_rest4,("(للصفات)",          !.
% (فعال) إناث : أنثى
run(List, W_rest1, R):-
        List = [A, B,"ا", C |R],      List1 = [A,B,C],
        List2 = [A, " ", B, C, ["ى" ,          List3 = {A, B, C, [  "ة",
        Collect(List1, W_rest1),    collect(List2, W_rest2),
        Collect(List3, W_rest3),    write("  ،(" جمع تكسير
        Nl,write("\t,(" : أصل الفعل هو
        Nl,write(W_rest1 , "أو"  ,W_rest2, "أو" , W_rest3),      !.

% (فعلان) حملان : حمل
% رغفان : رغيف
% شجعان : شجاع
% عميان : أعمى
run(List, W_rest1, R):-
        List = [A, " ", B, Cن" ،"ا" |R],         List1 = [A, " ", B, " ", C],
        List3 = [A, " ", B, "ا" , C], List2 = [A, B, "ي", C],
        List4 = ["، "أ A, B, ["ى",    collect(List1, W_rest1),
        Collect(List2, W_rest2),    collect(List3, W_rest3),
        Collect(List4, W_rest4),    write("  ،(" جمع تكسير
        Nl,write("\t,(" : أصل الفعل هو
Write(W_rest1,  "أو" ,W_rest2 أو" , \n", W_rest3,  "أو", W_rest4 ،( " (للصفات)،
```