Towards Structuring an Arabic-English Machine-Readable Dictionary Using Parsing Expression Grammars

Diaa Mohamed Fayed[1] Aly Aly Fahmy[2] Mohsen Abdelrazek Rashwan[3] Wafaa Kamel Fayed[4]
[1] Department of Computer Science, Faculty of Computers and Information,
[2] Department of Computer Science, Faculty of Computers and Information,
[3] Department of EECE, Faculty of Engineering,
[4] Department of Arabic Language and Literatures, Faculty of Arts,
Cairo University,
diaa.fayed@outlook.com, a.fahmy@fci-cu.edu.eg, mrashwan@ieee.org, wafkamel@link.net

***ABSTRACT*:** *Dictionaries are rich sources of lexical information about words that is required for many applications of human language technology. However, publishers prepare printed dictionaries for human usage not for machine processing. This paper presents a method to structure partly a machine-readable version of the Arabic-English Al-Mawrid dictionary. The method converted the entries of Al-Mawrid from a stream of words and punctuation marks into hierarchical structures. The hierarchical structure or tree structure expresses the components of each dictionary entry in explicit format. The components of an entry are subentries and each subentry consists of defining phrases, domain labels, cross-references, and translation equivalences. We designed the proposed method as cascaded steps where parsing is the main step. We implemented the parser through the parsing expression grammar formalism. In conclusion, although Arabic dictionaries do not have microstructure standardization, this study demonstrated that it is possible to structure them automatically or semi-automatically with plausible accuracy after inducing their microstructure.*

Keywords: Al-Mawrid, Arabic-English, Machine-Readable Dictionary, Parsing Expression Grammars

Received: 16 November 2013, Revised 19 December 2013, Accepted 24 December 2013

1. Introduction

Machine-readable dictionaries (MRDs) are electronic versions of the ordinary printed dictionaries. Publishers use MRDs to produce printed dictionaries. Fontenelle [1] differentiated between *computerized dictionaries* and *machine-readable dictionaries*. The computerized dictionaries are organized logically, so they are easily processed or transformed into lexical databases (LDBs) where each piece of information (*definitions, parts-of-speech, grammar codes, etc.*) is partly formalized and identified. The machine-readable dictionaries are flat text files with codes for driving the typesetting process. The computerized dictionaries have many applications [2] such as machine translation, natural language comprehension, pronunciation guides for speech recognition, etc.

Exploiting MRDs information by natural language processing (NLP) applications goes in two directions. The first direction is

to structure the total text of a dictionary and to transform it into a computerized dictionary. The second direction is to extract a selected set of syntactic and semantic information and to use them, with other language sources such as dictionaries, encyclopedia, and corpora, to build a computational lexicon, a lexical taxonomy, a semantic network, etc. Examples of the first direction are [2-7] and examples of the second direction are [1, 8-14].

Although MRDs research is no longer active research for Latin languages since researchers have shifted to machine-readable corpora [15, 16] as language source, MRDs research is still needed for resource-poor languages such as Arabic. The Arabic language still needs MRDs research for two purposes: to build language resources for Arabic NLP applications and to improve the Arabic lexicography in the theoretical and industrial aspects. This study to partially structure the Arabic-English Al-Mawrid dictionary automatically is the first published one.

Al-Mawrid targets human user and has unstructured text format. Therefore, we need to convert Al-Mawrid to a structured format before exploiting its data in any NLP applications. Generally, structuring or computerizing a dictionary text means that converting implicit information in the dictionary definitions into explicit information by identifying each significant piece of information and labeling it. We assume that Al-Mawrid has large and rich lexical knowledge that is useful in NLP applications. Furthermore, we assume that Al-Mawrid has a microstructure for each entry that we can capture and formalize automatically or semi-automatically. This study settled for the partly structuring of Al-Mawrid definitions at the punctuation level without handling the scattered information in the defining phrases. The proposed structuring method composes of successive steps where parsing is the main step. We implemented the parser through the parsing expression grammar (PEG) formalism.

By investigating samples of Al-Mawrid definitions, we found high regularity in their punctuations. Consequently, we assumed that a parsing technique is more suitable to capture the punctuation regularity. We adopted the PEG formalism in this study for some specific advantages such as (a) the PEG does not need a separate step of tokenization since tokenization rules take the same formats of any other grammar rule; and (b) the PEG has implemented in a Python library and the Python programming language has high performance and flexibility in processing text. Furthermore, the PEG has some other advantages that make it distinguishable. For more deep study of the PEG, you can refer to [17].

The rest of the paper is organized as follows. Section 2 introduces a short review of related works. Section 3 explains in detail the structure of Al-Mawrid structure and types of information in its definitions. Section 4 defines the formal definition of the parsing expression grammar formalism. Section 5 describes the steps of the structuring method and illustrative examples. Section 6 evaluates the overall structuring method. Finally, section 7 concludes the study and gives some future works.

2. Background and Related Works

Amsler [18] suggested that we need a database management system to enter the data of a dictionary, to edit text definitions and make semantic analysis, and to organize and store the resultant data of the analysis. The structuring task of a dictionary text is often compromised of three phases: (a) acquiring the machine-readable version of a dictionary and then loading its data into a database, (b) analyzing definitions, extracting information, and labeling them, and (c) adding the extracted knowledge to the database or building other forms of language resources such as lexical taxonomies, semantic networks, etc. Some studies merge the three phases such as Wilms [4] and others separate the phases such as Byrd [19].

MRDs researchers used two approaches to carry out the first phase of structuring a dictionary: (a) building a parser to structure the dictionary entries, and (b) using ad-hoc procedures to extract selected pieces of information, and then loading them into a database. In the following subsections, we give a short review of the methods that are used to acquire and load the data of a MRD into a database.

2.1 Building Parsers

Wilms [4] automated the Funk and Wagnalls Dictionary by building a parser based on pattern matching and transition network. The format of the parser output is Prolog clauses, where parentheses enclosed fields. Byrd [19] built a tree structure for each entry across three stages. The first stage segments information on the dictionary tape to separated entries and stores them in a direct access file. The keys in the file are the headwords and the data are the bodies of the entries. The second stage identifies and labels each kind of data in an entry to consistent and standard format. Byrd developed a parser to transform each entry of the dictionary into a tree structure. In Nagao *et al.* [3], a data entry entered the printed versions of the dictionaries as it is. Then, Nagao built an augmented transition network (ATN) parser to transform the implicit relationships

that are represented by special symbols or character types into explicit relationships that are represented by formatted records and links or pointers.

2.1.1 Developing Ad-hoc procedures

Nakramura and Nagao [20] extracted specific fields of the magnetic tape of the LDOCE and loaded them into a relational database. They developed extraction programs with a pattern matching based algorithm to analyze the definition sentences in LDOCE. Soha [21] transformed the unstructured text files of the RDI dictionary into a relational database. Soha carried out the transformation in three phases: (*a*) analyzing the text definitions, (*b*) imposing adopted structure on the RDI dictionary, and (c) splitting up the text definitions using punctuation and filling a relational database. Fontenelle [1] transformed the typesetting taps of Collins-Robert French-English dictionary into a relational database. He wrote a series of software applications for retrieving records and a program for enriching the database with two fields. Alshawi [5] used a series of batch jobs and a general text editor to transform the stream of characters of the LDOCE data structures on typesetting taps into Lisp s-expressions (symbolic expressions). Mayfield and McNamee [6] developed a script language called ALBET. Kazman [2] transformed the dictionary and its supplement into a MRD using a manual system of mark-up and data entry.

3. Al-Mawrid Structure

The Arabic-English Al-Mawrid dictionary is a general-purpose dictionary. Al-Mawrid is a medium-sized dictionary that has about 33463 distinct headwords and about 58547 subentries. Investigating and understanding the structure of a dictionary entails understanding its macrostructure, its microstructure,and its kinds of information in definitions. In Hartman [22], the *macrostructure* refers to the overall access format of the dictionary. The *microstructure* refers to how the entry is structured, how information about the headword is provided and presented, and how the discourse structure of the entry is prepared appropriately for the benefit of the anticipated user.

The author of Al-Mawrid arranged the headwords alphabetically according to the first letters of the headwords. An entry of Al-Mawrid starts by a bold headword. When a headword has more than one meaning or sense, each meaning occupies a subentry that is cited in separated lines. Subentries that express collocations, idioms, or examples are cited lately. A subentry consists of a mandatory Arabic section and an optional English section (*translation*). In addition, for distinguishing the headwords, each subentry text, except the first one, is shifted from the headword by two space characters. The Arabic section consists of three fields that we name *header*, *explanation* and *cross-reference*. The first field of an Arabic section is mandatory but the other two fields are optional. A colon separates the header from its explanation. A dash may precede the cross-reference field. The subentry may express *declaration*, *question*, or *exclamation*. The definition phrases or translation equivalence (*TE*) phrases take the same name of their subentry expression.

The three fields of an Arabic section have the same structure: each field consists of one or more words or phrases that are separated by either an Arabic comma or the conjunction word. These comma-separated phrases are almost synonymous or near synonyms. The header has the headword of an entry if its subentry represents a sense of the headword. If a subentry does not represent a sense of the headword, it contains idioms, examples or restricted collocation. The morphologic, syntactic, or semantic information is scattered in the header or explanation fields. Some semantic domain or subject labels, such as "نبات" (plant), are enclosed in bracket parentheses (e.g.[نبات]) or in parentheses (e.g. (نبات)).

The English section has one or more translation equivalence groups (*TEGs*) that are separated by semicolons. Each TEG has one or more translation equivalence (*TE*) phrases that are separated by comma. The phrases of a TEG are synonyms. Figure 1 shows the entry structure of the headword "مُتَقَدِّم" in the printed version of Al-Mawrid.

Arabic dictionaries have problems with some aspects that lexicographers should solve [23]. These problems prose challenges for computerizing Arabic dictionaries. Hawaary [23] mentioned some deficiencies in the Arabic dictionaries at three levels: the theoretical foundation, the analysis, and the representation of syntactic-semantic content. In addition, Al-Mawrid has some special deficiencies. The following issues are some of the shortcomings in Al-Mawrid that complicate the structuring or computerizing task.

3.1 Inconsistency

a. Semantic or subject labels exist either in bracket parentheses [نبات] or in parentheses (نبات). A semantic relation may have

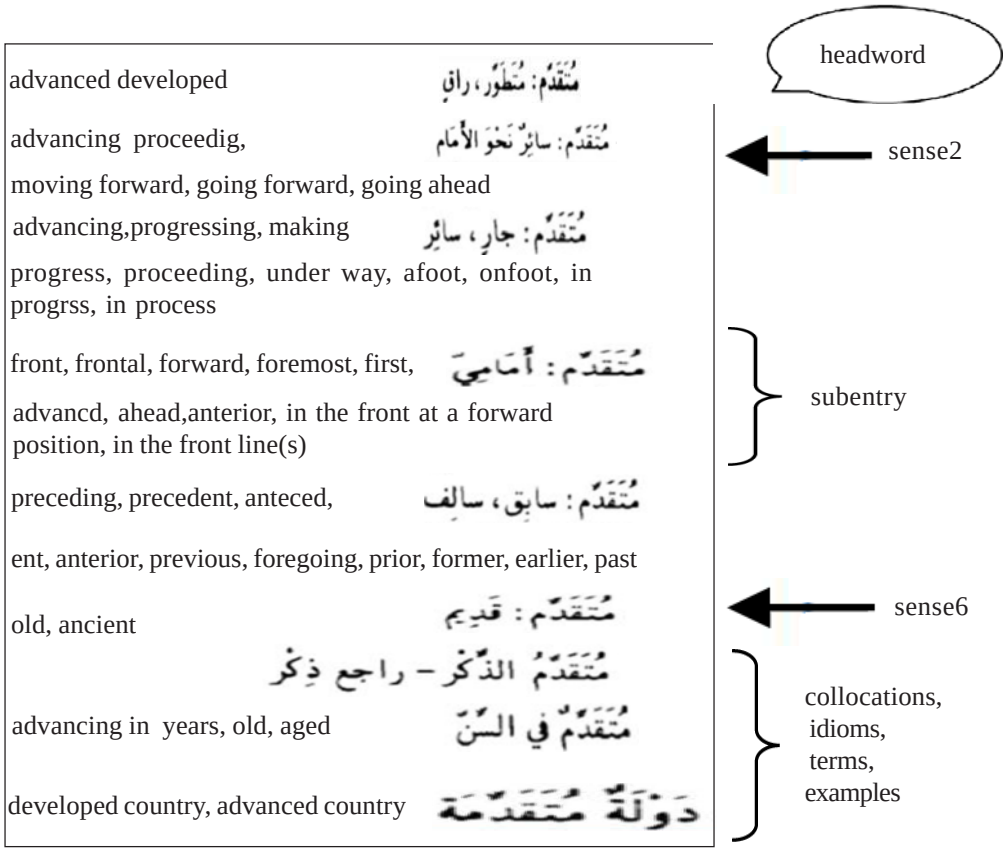

Figure 1. Excerpt of text of AL-Mawrid entry of the headword " مُتَقَدِم "

more than one key word that manifests it. For example, the antonym relation is expressed by any of the words "ضد", "عكس", and " لا ".

b. The adjective "صفة" and name "اسم" are the only part-of-speech tags in Al-Mawrid and they are very rare.

c. There is no a consistent clue that determines whether a subentry is either a meaning of the headword or an example, a collocation, an idiom, etc. A colon mark distinguishes the meaning only if it exists after a copy of the headword.

d. The explanation in a subentry may be expressed by phrases that follow a colon mark, by phrases in parenthesis like "عقيد (في الجيش)", or by nothing at all.

3.2 Ambiguity

The cross-reference word or phrase may refer to more than one headword and each headword may have more than one subentry. Furthermore, a subentry may represent either a meaning of the headword or an example, an idiom, a term, and a restricted collocation.

3.3 Parentheses

The parentheses information has large varieties and has multiple purposes of functions. For example, the parentheses may contain prepositions, synonyms, examples, explanations, antonyms, morphological information, spelling variations, etc.

3.4 Indefinite key phrases

The author of Al-Mawrid does not cite key phrases or key words that manifest morphologic, syntactic, or semantic information in the introduction of Al-Mawrid, so we need to discover these words before either structuring definitions or extracting information.

4. Parsing Expression Grammar

Parsing expression grammars (*PEGs*) formalism was built on the Backus-Naur Forms (*BNFs*) [24] and the Regular Expressions

[1]Wikipedia, *Backus–Naur Form*. 2013; Available from: http://en.wikipedia.org/wiki/Backus%E2%80%93Naur_Form.

(*REs*) notations to form an alternative class of formal grammar. The PEG is essentially analytic rather than generative[1].

A PEG is defined [17] formally as a 4-tuple $G = (V_N, V_T, R, e_S)$, where V_N is a finite set of nonterminal symbols, V_T is a finite set of terminal symbols, R is a finite set of rules, e_S is a parsing expression termed the *start expression*, and $V_N \cap V_T = 0$. Each rule $r \in R$ is a pair (A, e), which we write $A \leftarrow e$, where $A \in V_N$ and e is a parsing expression. For any nonterminal A, there is exactly one e such that $A \leftarrow e \in R$. R is therefore a function from nonterminal symbols to expressions, and we write $R(A)$ to denote the unique expression e such that $A \leftarrow e \in R$.

Consider e, e_1, and e_2 are parsing expressions. Furthermore, The variables a, b, c, d are used to represent terminals, A, B, C, D for nonterminals, x, y, z for strings of terminals, and e for parsing expressions. Inductively, *parsing expressions* are defined as follows [17]:

1. ε the empty string.
2. a, any terminal, where $a \in V_T$.
3. A, any nonterminal, where $A \in V_N$.
4. $e1e2$, a sequence.
5. $e1/e2$, prioritized choice.
6. $e*$, zero-or-more repetitions.
7. $!e$, a not-predicate.

We used the Pyparsing library[2] that is a python implementation of the Parsing Expression Grammar [25, 26].

5. Structuring Method

By investigating samples of Al-Mawrid definitions, we found that the microstructure punctuation of Al-Mawrid has high percentage of regularity that we can capture and formalize. However, the morphologic, syntactic, and semantic information are scattered in the defining phrases randomly and inconsistently. Therefore, our strategy to formalize an entry of Al-Mawrid is to capture the punctuation by a parsing technique firstly, and then extract the scattered information by an information extraction technique. This *divide-and-conquer* strategy avoids building complex grammars in order to capture both punctuations and scattered information in one shot.

This work addressed only the first phase of the structuring phases of a dictionary text: acquiring and loading the data of Al-Mawrid into a database format. We cannot structure the total text of a large dictionary like Al-Mawrid in one automatic shot, but we often accomplish this by some combination of automatic, semiautomatic, and manual steps. We subdivided the structuring task into cascaded steps and the output of each step is the input to the consequent step. We implemented each step by one module. The modular design makes both testing code and evaluating step easier. Another advantage of the modular design is to avoid building complex and convoluted grammar for Al-Mawrid that can take much time and effort. The complex grammar of Al-Mawrid definitions results from the inconsistency and rich variability formats of definitions. In the following section, we explain the structuring steps in detail.

5.1 Acquiring the Dictionary Data

A data entry explored the software of Al-Mawrid and copied the entries information of each headword into a spreadsheet file. The data entry dropped out the pictures when coping. In addition, the electronic version of Al-Mawrid has no any font styles. Each subentry of a headword occupies two separated lines in the spreadsheet file, one for Arabic section and the other for the English translation. If English translation is absent, its line is empty.

5.2 Assigning Headwords to their Description Data

The purpose of this step is to recover the principal database structure of Al-Mawrid. The principal structure of a dictionary is composed of an index of the headwords and the entries information that is related to these headwords. Recovering the principal database structure enables us to know the information associated with each headword separately. Consequently, we can make more exploring and processing on the associated information, collecting basic information about the dictionary

[2] McGuire, P. (2003). *Pyparsing Wiki Home*. Available from: http://pyparsing.wikispaces.com/home.

entries, and assigning the results to the respective headwords. The associated information with a headword can be definitions, idioms, terms, collocation, or examples, and translations. We developed procedures to transform the spreadsheet files into a rudimentary database of two flat text files. In this rudimentary database, one file is the index file that contains the distinct headwords and index numbers that point to the associated information in a data file. The second file is a data file that contains the associated information of the headwords.

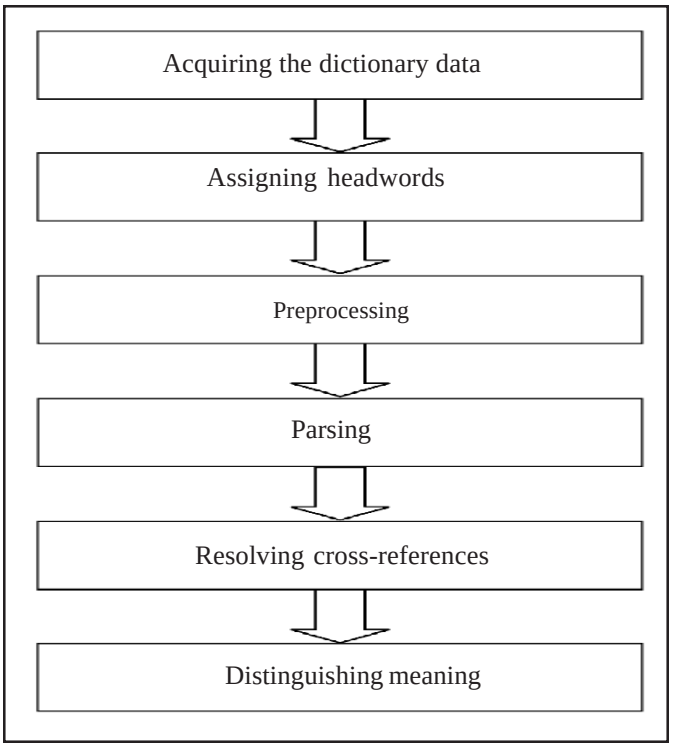

Figure 2. Steps of structuring

5.3 Preprocessing

In this step, we carried out three operations on the data: (a) cleaning the data from some un-useful words and faulted characters, (b) flattening or consuming some parentheses information, and (c) converting domain or subject labels in the parentheses to labels in square brackets. We removed some un-useful words and characters such as "...", "*etc*.", and "الخ '". Al-Mawrid uses parentheses abundantly and for diverse purposes. The volume and diversity of the parentheses information in Al-Mawrid is one of many challenges to structure its data.

In the English section of a subentry, there are two kinds of parentheses: the *connected parentheses* and the *separated parentheses*. The connected parentheses express multiple word forms or morphological information in compressed form. An example of this kind is the string "*susten* (*ta*) *tion*." This a compressed form that expresses two alternative word forms: "*sustentation*" and "*sustention*". The second kind is the separated parentheses. An example of that kind is the string "*to seek protection* (*in, with, from*)". The separated parentheses have many variations and contain many kinds of information, so they need further study to cope with and to structure. We did not analyze separated parentheses in this study except the kind of *one-word* parenthesis.

The one-word parenthesis contains a set of continuous characters that expresses preposition, alternative word form, prefixes, etc. The one-word parentheses may be connected or separated. We structured all the one-word parentheses by one technique. We made two copies of the strings that contain parentheses: one copy without the parentheses information and another copy with parentheses information. Then, we concatenated the two copies of the strings and inserted a comma in between them. For example, we transform the string "*the* (*Heavenly*) *Father*" to "*the Heavenly Father*" and "*the Father*". This technique is based on a note of the author in Al-Mawrid introduction: all the information in the parentheses is optional and can be removed.

In the Arabic section, all parentheses are of separated kind. Arabic parentheses contain various and multiple information, so they need more deep analysis before we can structure them. This study handled only parentheses that contain labels of semantic or subject domain like "(نبات) عائق" that is transformed into " عائق[نبات]". The purpose of this transformation is to normalize all semantic or subject labels to be in square bracket, so they can be captured and formalized in a PEG grammar.

5.4 Parsing

Since the author of Al-Mawrid does not give a detail explanation about the structure of definitions, we should induce the grammar of Al-Mawrid by investigating ideal examples. To induce the grammar of the microstructure, we used ideal selected examples of Al-Mawrid data as an experiment sample. We built the grammars of the parser using the bootstrapping technique. The main output of the parser program is a tree structure. In addition, the parser program produces a by-product corpus of both Arabic defining phrases and English translation equivalences (TEs) phrases. We can use the resulting corpus in further analysis to discover string patterns that manifest additional morpho-syntactic and semantic information.

Since we do not know, in advance, the sublanguage of the dictionary definitions, as explained above, we adopted the data-driven (bootstrapping) technique to build the parser. We built the grammars by investigating ideal examples of the dictionary bit by bit using the parsing expression grammars (PEGs). Figure 3 illustrates the methodology that is used to build the grammars. We stopped the bootstrapping process when the parser can parse all the dictionary entries. Figure 4 shows an example of an input and an output of the parser.

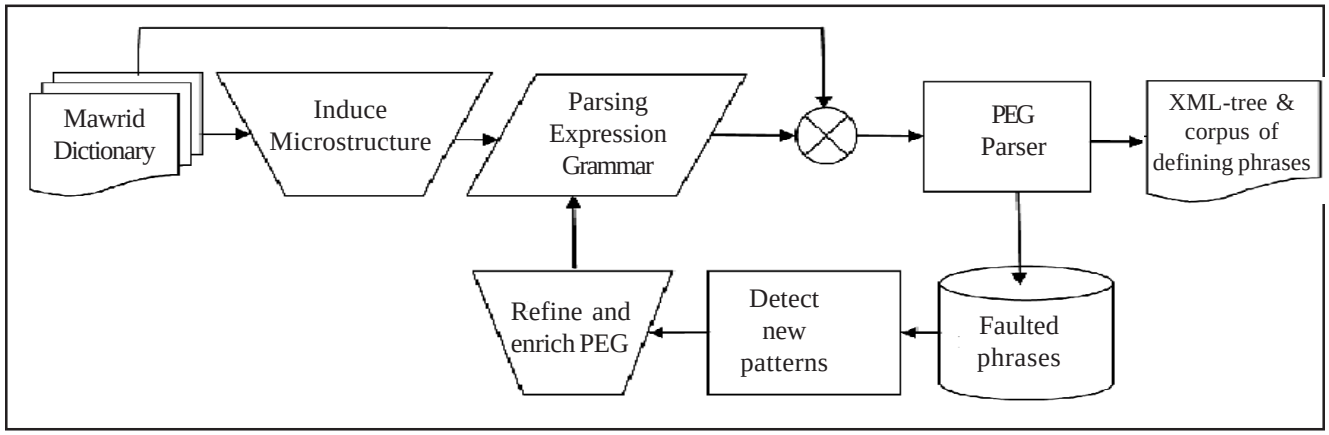

Figure 3. Bootstrapping the microstructure grammar

Assembling the defining and translation phrases in a corpus enables us to process and analyze them by statistical methods to discover the string patterns that manifest syntactic and semantic relations. Table 2 shows examples of the defining phrases corpus.

5.5 Resolving Cross-References

The cross-reference in a dictionary is a mechanism that guides users to refer to other entries or subentries of the dictionary. The cross-reference in Al-Mawrid refers to an entry or a subentry that is related to the given meaning or subentry.

The cross-reference device in Al-Mawrid is either the key word " راجع " (refer-to) or the key phrase "راجعها في أماكنها" (refer-to-them-in-their-place). The first refers explicitly to a word or a phrase after the key word " راجع " and the second specifies the referred words or phrases that the user should go before the key phrase " راجعها في أماكنها. The number of cross-references in Al-Mawrid is about 13 % of the subentries. There are only about fifty of the second type in Al-Mawrid.

5.5.1 Cross-reference Problems

The cross-references in Al-Mawrid have some complexities: (a) a cross-reference may refer to one or more entries or subentries, (b) a cross-reference word may have more than one meaning, and (c) a cross-reference may be a phrase that is not a headword but that exists in a subentry as an idiom, a collocation, a verbal phrase, or an example. Therefore, we need to know which a headword we will search the dictionary to get the meanings of that phrase. Figure 5 illustrates the various complexities of the

cross-references.

Figure 5 explains the complexity of the cross-reference ambiguity. The headword "أَحَاطَ" has a subentry "أَحَاطَهُ بالرِّعَايَةِ أو العِنَايَةِ أو الاهْتِمام - راجع رَعَى، عُنِيَ بـ، اِهْتَمَّ بـ". The cross-reference phrases are "رَعَى", "عُنِيَ بـ" and "اِهْتَمَّ بـ". The cross-reference resolver search the index file to get the indices of the phrases: "رَعَى, عُنِيَ بـ" and "اِهْتَمَّ بـ". The word "رَعَى" is the only word that has indices numbers; the other two phrases return zero indices because they have prepositions. The index file contains only words as indices not phrases.

index		data			
آئِب	12	12	آئِب:	راجع، عائِد	returning, coming back, going back; returnee, returner

```
<entry enumber = "3" headword = " آئِب ">
 <group gnumber = "1">
<sense snumber = "12">
 <arabic>
<header>
<declaration> آئِب </declaration>
</header> <definition>
<declaration> راجع </declaration>
<declaration> عائِد </declaration>
 </definition>
 </arabic>
 <english>
 <header>
<subsense>
 <declaration>returning</declaration>
 <declaration>coming back</declaration>
 <declaration>going back</declaration>
 </subsense> <subsense>
<declaration>returnee</declaration>
<declaration>returner</declaration>
 </subsense>
</header>
 </english>
 </subsense>
 </group>
 </entry>
```

Figure 4. Parser input and output example

5.5.2 Resolving the Cross-reference Ambiguities

We can resolve the cross-reference ambiguities in two steps. First, we replace the cross-reference words or phrases by a set of index numbers. Because of the ambiguity of some words, the set may have more than one number. Second, we identify

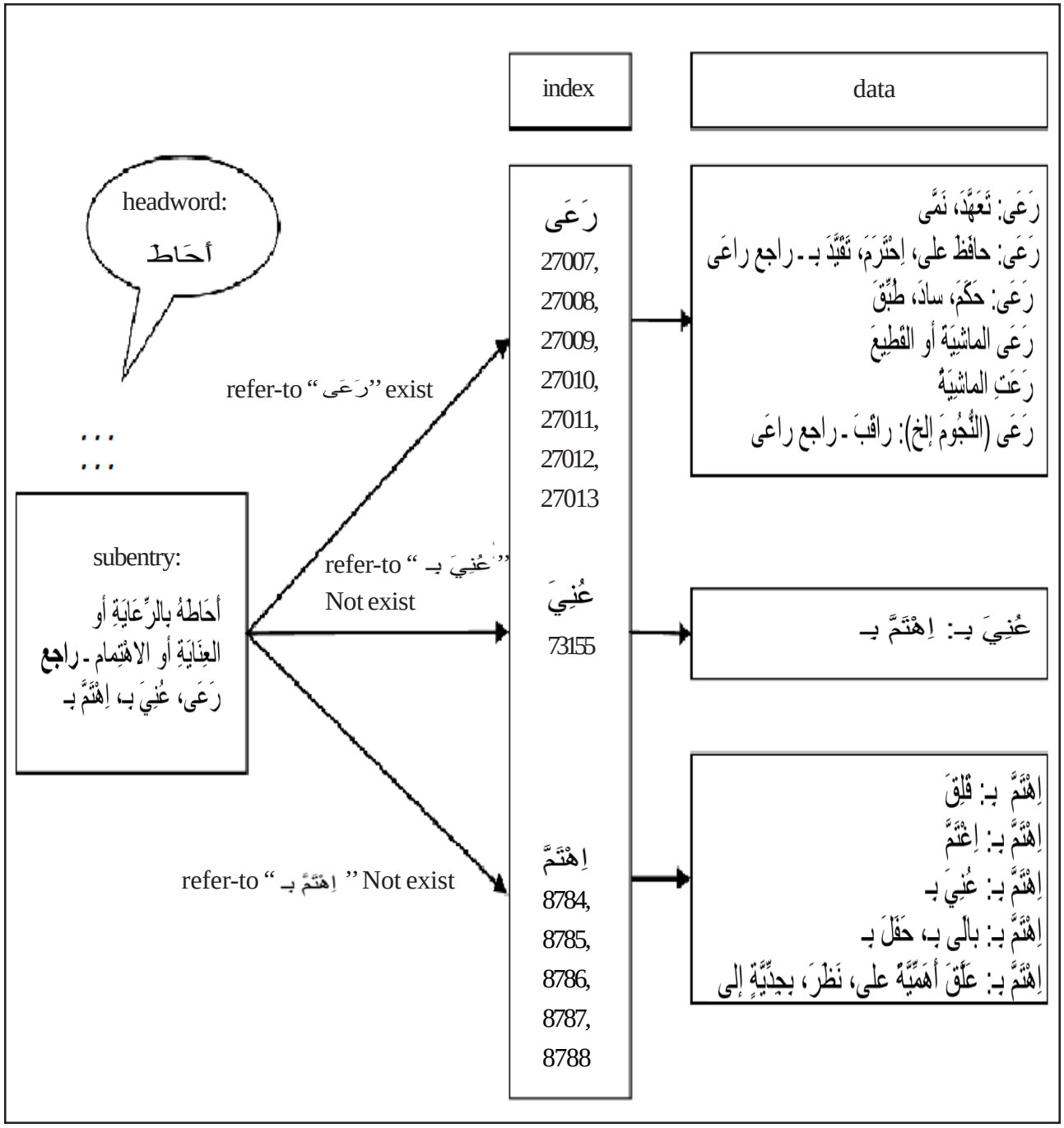

Figure 5. Example of cross-reference ambiguity

exactly what sense is required, preserve its index number, and remove the remaining index numbers. Actually, we made the two steps only for the cross-reference key phrase "راجعها في اماكنها" because their size is small. On the other hand, the cross-reference key word "راجع" has large size, so we made only the first step for them to resolve ambiguity. The second step needs further study and analysis to solve ambiguities. Figure 6 is an example that illustrates the first step of resolving the cross-reference ambiguity. The cross-reference numbers is the "*refs*" value attribute of the crossreference element.

5.6 Distinguishing Meanings

The information that is associated with a headword may include meanings, idioms, terms, collocations, and examples. The Author of Al-Mawrid cites the subentries of a headword without distinguishing between kinds of them. The author depends

```
<sense snumber = "1193">
<arabic>
<header>
<declaration> أحَاطهُ بالرِّعَايَةِ أو العِنَايَةِ أو الاهتِمام </declaration>
</header>
<crossreference refs = "27007, 27008, 27009, 27010, 27011, 27012, 27013" >
 <declaration> رَعَى، </declaration>
<declaration > عُنِيَ بـ، </declaration>
<declaration > اِهْتَمَّ بـ </dc>
</crossreference>
</arabic>
<english/>
</sense>
```

Figure 6. Cross-reference example

```
<entry enumber = "20869" headword = "عَيْن">
<group gnumber = "1">
<sense snumber = "37360" ism = "1">
 <arabic>
<header>
<declaration> عَيْن</declaration>
 </header>
 <definition>
<declaration> عُضْوُ البَصَر </declaration>
</definition>
 </arabic>
<english>
<header>
<subsense>
 <declaration>eye</declaration>
</subsense>
</header>
 </english>
```

```
<sense snumber = "37388" ism= "0">
<arabic>
<header>
<declaration>فرْضُ عَيْن،</declaration>
 </header>
</arabic>
<english>
 <header>
  <subsense>
 <declaration>individual duty</declaration>
  </subsense>
</header>
</english>
 </sense>
```

Figure 7. Examples of the ism distinguisher

on the understanding of the dictionary user in distinguishing among all these kinds. However, structuring a dictionary entails making these kinds as distinguishable as possible.We noticed that most of the subentries that contain meanings of a headword start by a copy of the headword itself and colon mark. We exploited this phenomenon by implementing a heuristic that acts as a distinguisher. We implemented the distinguisher as a flag that has a binary value. If a subentry represents a meaning of headword, the flag value equals to one. On the other hand, if a subentry represents an idiom, a term, a collocation, or an example, the flag value equals to zero. We implemented the flag as a binary attribute "*ism*" (is-meaning or is-sense) in the "*sense*" element of an entry subtree. Figure 6 illustrates two examples of the "*ism*" attribute. However, distinguishing among

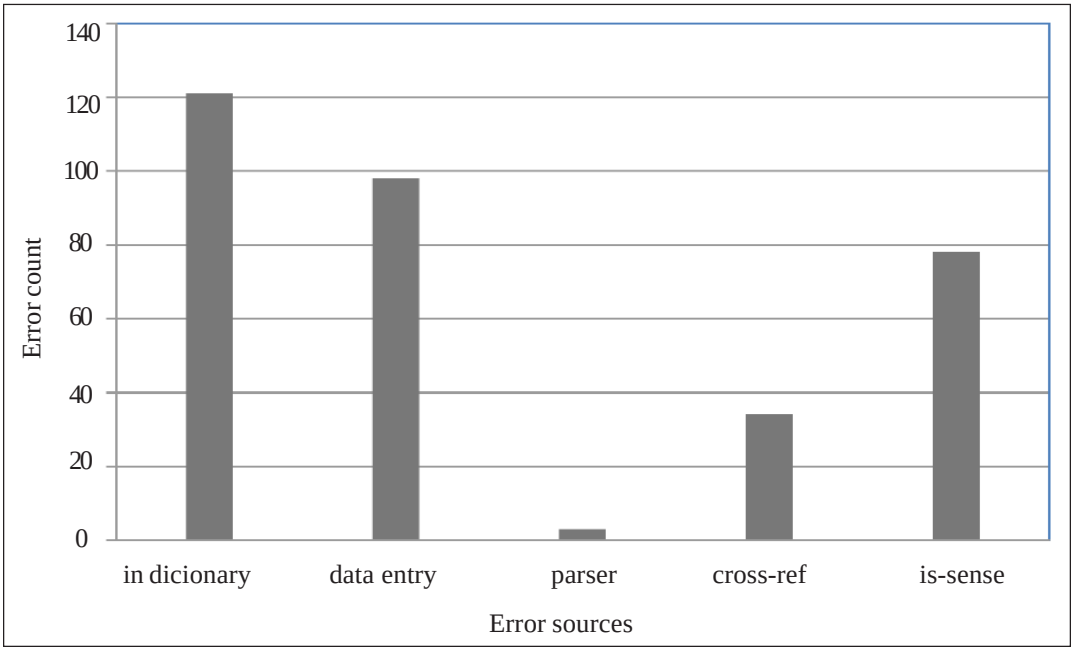

Figure 8. Error count and error sources of the structuring steps

Source of error	**Examples of errors**
dictionary	1.The phrase in the cross-reference does not exist in the headword list. 2. The colon that distinguishes meaning is absent. 3. The head or cross-reference contains parentheses and/or comma. 4. The head or cross-reference has diacritics. 5. Entry meanings are shifted to another previous or followed entry. 6. Mismatches exist between the electronic and the printed versions.
data entry	1. Some entries and subentries are dropped out. 2. Subentries are shifted to other entries. 3. Mismatches exist between electronic and printed version.
parser	1. Faulted characters exist in the Arabic string. 2. Spaces exist in a head phrase of subentry.
cross-reference (cross-ref)	1. Refs contain parenthesis. 2. Refs are phrases or multi-words that do not exist in the index file. 3. Some headwords are missed. 4. Space is missed in a head phrase
is meaning (ism)	1. Header contains parenthesis. 2. Refs contains parenthesis. 3. Some headwords are absent. 4. Some colons and comma are absent. 5. Some headers are phrases. 6. Header contains two or more words separated by comma. 7. Refs contains preposition.

Table 2. Some types and of errors and causes

an idiom, a term, a collocation, or an example requires further knowledge and processing steps such as looking up an English dictionary.

6. Results and Discussion

We selected the Ayn "ع" letter from Al-Mawrid to evaluate the structuring steps. The Ayn "ع" letter represents about 4.6 % of Al-Mawrid data volume. The only criteria for selecting the Ayn letter are that its volume is suitable for both the evaluation experiment and the copyright considerations. Figure 8 shows the sources of errors and errors count. Table 3 illustrates the types of errors for each source. From Figure 8, we inferred that the parser produced the smallest number of errors; this indicates that the entries of Al-Mawrid have high-organized punctuation. We noticed also that the main sources of errors are the dictionary and the data entry. By analyzing the errors of each error source in Table 3, we noticed that the errors of cross-refs and is-meaning are duplicated errors of the first two error-sources: the in-dictionary and the data-entry. As a result, the main independent error-sources are the in-dictionary, the data-entry, and the parser. The cross-refs and is-meaning are dependent error-sources. If we compute only errors of the independent error-sources, the percentage of accuracy is about 91 %. On the other hand, if we take all the errors of all error-sources, the accuracy is about 87%.

7. Conclusions and Future Work

The accuracy percentage of the structuring steps, especially of the parser step, proved our previous assumption; Al-Mawrid has high punctuation regularity in its microstructure, and so it can be formalized automatically. In addition, investigating the by-product corpus of defining phrases showed that the defining phrases have prospective rich information that can be extracted and formalized. The main lesson of this study is that we can formalize Arabic dictionaries, with plausible accuracy, thought they have no standardization formats. In future study, we can pursue the following tasks on the present work:

1. Annotating the dictionary definitions by part-off-speech (*POS*) tags and Semantic tags,

2. Analyzing and structuring the phrases in parentheses,

3. Resolving the cross-references completely such that each cross-reference refers to the exact meaning of a specific headword,

4. Extracting syntactic and semantic information from the corpus of defining phrases,

5. Disambiguating words of the defining phrases and translation equivalence phrases,

6. Analyzing the conjunction words in both Arabic "أو" and English "*or*".

References

[1] Fontenelle, T. (1997). Using a Bilingual Dictionary to Create Semantic Networks. *International Journal of Lexicography*, 10 (4) 275.

[2] Kazman, R. (1986). Structuring the Text of the Oxford English Dictionary through Finite State Transduction, in Dept. of Computer Science. University of Waterloo.

[3] Nagao, M., et al. (1980). An Attempt to Computerized Dictionary Databases. Association for Computational Linguistics.

[4] Wilms, G. J. (1990). Computerizing a Machine Readable Dictionary. *In*: Proceedings of the 28th annual Southeast regional conference. ACM New York, NY, USA.

[5] Alshawi, H., Boguraev, B., Carter, D. (1989). Placing the Dictionary On-line, in Computational Lexicography for Natural Language Processing. Longman Publishing Group. p. 41-63.

[6] Mayfield, J., McNamee, P. (2002). Converting On-Line Bilingual Dictionaries from Human-Readable to Machine-Readable Form. ACM.

[7] Hastings, A. (1994). Loading a Bilingual Dictionary Into LDB.

[8] Castro-Sánchez, N., Sidorov, G. (2011). Automatic Acquisition of Synonyms of Verbs from an Explanatory Dictionary using Hyponym and Hyperonym Relations. *Pattern Recognition*, p. 322-331.

[9] Wang, T., Hirst, G. (2009). Extracting Synonyms from Dictionary Definitions, in Computer Science. University of Toronto.

[10] Klavans, J., Whitman, E. (2001). Extracting Taxonomic Relationships from On-line Definitional Sources using Lexing. ACM.

[11] Dolan, W., Vanderwende, L., Richardson, S. D. (1993). *Automatically Deriving Structured Knowledge Bases from On-Line Dictionaries*, in *PACLING*. Pacific Association for Computational Linguistics: Simon Fraser University, Vancouver, BC. p. 5-14.

[12] Montemagni, S., Vanderwende, L. (1992). Structural Patterns vs. String Patterns for Extracting Semantic Information from Dictionaries.

[13] Ahlswede, T., Evens, M. (1988). *Generating a Relational Lexicon from a Machine–Readable Dictionary. International Journal of Lexicography*, 11(3) 214.

[14] Martin, S. C., Roy, J. B., George, E. H. (1985). *Extracting Semantic Hierarchies from a Large On-line Dictionary*. *In*: Proceedings of the 23rd annual meeting on Association for Computational Linguistics. Chicago, Illinois: Association for Computational Linguistics.

[15] Ide, N., Véronis, J. (1994). Machine Readable Dictionaries: What have we learned, where do we go. Citeseer.

[16] Wilms, G. J. (1995). Automated Induction of a Lexical Sublanguage Grammar Using a Hybrid System of Corpus-and Knowledge-based Techniques. Citeseer.

[17] Ford, B. (2004). Parsing Expression Grammars: A Recognition-Based Syntactic Foundation. *In*: Proceedings of the 31st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages. ACM.

[18]Amsler, R. A. (1980). The Structure of the Merriam-Webster Pocket Dictionary. The University of Texas at Austin.

[19] Byrd, R., et al. (1987). Tools and Methods for Computational Lexicology. Computational Linguistics, 13 (3-4) 219-240.

[20] Nakramura, J. -i., Nagao, M. (1988). *Extraction of Semantic Information from an Ordinary English Dictionary and its Evaluation*. *In*: Proceedings of the 12th conference on Computational linguistics. 1988. Budapest, Hungry: Association for Computational Linguistics.

[21] Eid, S. M. (2010). Automatic Generation of Thesaurus from Arabic Lexical Resources, in Electronics and Communication Engineering. Faculty of Engineering, Cairo University: Giza, Egypt.

[22] Hartmann, R. R. K., James, G. (2002). Dictionary of Lexicography. London and New York: Routledge.

[23] Garshol, L. M. (2003). BNF and EBNF: What Are They and How Do They Work. acedida pela última vez em, 16.

[24] McGuire, P., ed. (2007). Getting Started with Pyparsing. O'Reilly Media, Inc.

[25] McGuire, P. (2006). Introduction to Pyparsing: An Object-oriented Easyto-Use Toolkit for Building Recursive Descent Parsers. World, p. 1-15.

Arabic References

هواري, ع.ا.إ.ع.ا., و.ك. فايد and ,م.ع.ا. رشوان, لغة التعريف في المعجم العربي العام الحديث: إشكالية الصياغة والمحتوى، رؤية لغوية حاسوبية in ,قسم اللغة العربية وآدابها. 2008, جامعة القاهرة: الجيزة.