# Arabic Disambiguation Using Dependency Grammar

Daoud Daoud (1), Mohammad Daoud (2)

(1) Princess Sumaya University for Technology, P.O.Box 1438 Al-Jubaiha
11941 Jordan
daoud@batelco.jo
(2) Laboratoire LIG - Université Joseph Fourier 85, rue de la Bibliothèque,
38041 Grenoble, France
Mohammad.Daoud @imag.fr

**Abstract** In this paper, we present a new approach to disambiguation Arabic using a joint rule-based model which is conceptualized using Dependency Grammar. This approach helps in highly accurate analysis of sentences. The analysis produces a semantic net like structure expressed by means of Universal Networking Language (UNL) - a recently proposed interlingua. Extremely varied and complex phenomena of Arabic language have been addressed.

**Keywords:** Dependency Grammar, Arabic Language, Disambiguation, EnCo, UNL

## 1 Introduction

The Arabic grammarians (1200 years ago, Iraq) recognized government and syntactic dependency structure. Many researchers believe that the theoretical framework of ancient Arabic grammar and more specifically its theory of case assignment (***nazariyyat al'amal***), can be considered a dependency grammar(Owens 1988; Kruijff 2002).

Dependency grammar (DG) is a class of syntactic theories developed by Lucien Tesnière. It is distinct from phrase structure grammars, as it lacks phrasal nodes. Structure is determined by the relation between a word (a head) and its dependents. Dependency grammars are not defined by a specific word order, and are thus well suited to languages with free word order, such as Arabic. Specifically, Arabic reveals strong interaction between morphological and syntactic processing, which challenges the validity of NLP models that are based on different phases (layers). The available Arabic rule-based systems use the pipeline model (where morphology is performed first and syntactic processing follows) for processing and disambiguation. It is obvious that this approach is not adequate for Arabic. On the other hand, one would not expect statistical techniques to perform well on infixing languages like Arabic.

We will take a different approach from previous work. Our system is a rule-based one, which is conceptualized by using dependency grammar, in which linguistic structure is described in terms of dependency relations among the words of a sentence; it does so without resorting to units of analysis smaller or larger than the word. Although dependency grammar has its roots

to the work of early Arabic Grammarians, all of the existing (rule-based) Arabic processing systems are built on phrase structure theory. Processing text using phrase structure framework may suit languages like English, but not a nearly free order language like Arabic (Mel'tchuk 1988; Covington 1992).

We suggest performing morphological and syntactic processing of Arabic text in a single and joint framework; thereby facilitating the disambiguation process. We will first discuss the sources of ambiguity in Arabic. Then, we discuss methods of disambiguation based on the dependency grammar and the necessity of having an integrated model. Finally, we present the architecture and implementation of our system.

## 2 Properties of Arabic Language

Compared to French or English, Arabic is an agglutinative and highly inflected language shows its proper types of difficulties in morphological disambiguation, since a large number of its ambiguities come from both the stemming and the categorization of a morpheme while most of ambiguities in French or English are related to the categorization of a morpheme only. Phrases and sentences in Arabic have a relatively free word. The same grammatical relations can have different syntactic structures. Thus, morphological information is crucial in providing signs for structural dependencies. Arabic sentences are characterized by a strong tendency for agreement between its constituents, between verb and noun, noun and adjective, in matters of numbers, gender, definitiveness, case, person etc. These properties are expressed by a comprehensive system of affixation.

Ambiguities are mainly caused by the dropping of the short vowels. Thus, a word can have different meanings. In Arabic there are three categories of words: noun, verbs and particles. The dropping of short vowels can cause ambiguities within the same category or across different categories. For example, the word قبل [qbl] has the following interpretations (from Buckwaltel Arabic morphological analyser):

INPUT STRING: قبل
LOOK-UP WORD: qbl
SOLUTION 1: (qabola) [qabola_1] qabola/PREP
(GLOSS): + before +
SOLUTION 2: (qaboli) [qabola_1] qaboli/PREP
(GLOSS): + before +
SOLUTION 3: (qabolu) [qabolu_1] qabolu/ADV
(GLOSS): + before/prior +
SOLUTION 4: (qibal) [qibal_1] qibal/NOUN
(GLOSS): + (on the) part of +
SOLUTION 5: (qabila) [qabil-a_1] qabil/VERB_PERFECT+a/PVSUFF_SUBJ:3MS
(GLOSS): + accept/receive/approve + he/it <verb>
SOLUTION 6: (qab~ala) [qab~al_1] qab~al/VERB_PERFECT+a/PVSUFF_SUBJ:3MS
(GLOSS): + kiss + he/it <verb>

Additionally, Arabic uses a diverse system of prefixes, suffixes, and pronouns that are attached to the words, creating compound forms that further complicate text manipulation. Identifying such particles is crucial for analyzing syntactic structures as they reveal structural dependencies such as subordinate clauses, adjuncts, and prepositional phrase attachments. This means that there are multiple ways in which a word can be categorized or broken down to its constituent morphemes. For instance, the word كوارث can be segmented differently as presented in table 1:

| | |
|---|---|
| catastrophes/tragedies | Noun (broken plural) |
| Like/such as + inheritor | ka/PREP+wAriv/noun |

Table 1: Ambiguity caused by compound forms

On other cases, correct morphological analysis is required to resolve structural ambiguities among Arabic sentence. For example, consider the first sentence in table 2, the "ين" suffix attached to "ولد" provides information about number (dual) and case ending (accusative). The accusative sign determines the syntactic roles of each constituents of the first sentence although it is in the basic order VSO. In the second sentence, the same suffix disambiguate the syntactic roles despite that the object precedes the subject. In the third sentence, the verb hit "ضربا" follows the two boys "الولدان" and there is a number agreement between both of them. Additionally, the two boys "الولدان" takes the nominative sign and hani "هانيا" takes the accusative sign suggesting that: Hani is the object and the two boys are the subject.

| Sentence | | | Word order | Syntactic roles |
|---|---|---|---|---|
| الولدين<br>the two boys | هاني<br>Hani | ضرب<br>Hit | VSO | *Hani* is the subject<br>*The two* boys are the object |
| هاني<br>Hani | الولدين<br>the two boys | ضرب<br>Hit | VOS | *Hani* is the subject<br>*The two boys* are the object |
| هانيا<br>Hani | ضربا<br>hit | الولدان<br>the two boys | SVO | *Hani* is the object<br>The two boys are the subject |

Table 2: Examples of structural ambiguities

These examples show that there is a circular dependency between syntactic processing and morphological analysis. The segmentation and morphological analysis is driven by the structural dependencies within the sentence. Equally, syntactic roles are disambiguated by morphological dependencies. Thus, independent morphological and syntactic analyzers for Arabic are not adequate.

# 3 The Role of Dependency Grammar in Disambiguation

Parsing morphologically rich, free word order languages is a challenging task. However, in case of Arabic, it is possible to employ the set of dependencies and constraints in disambiguation process. In many cases these dependencies are manifested by apparent case assignment, prefixing, infixing suffixing and agreement. Thus, the morphological analysis of a word-form, and in particular its morphological segmentation, cannot be disambiguated without reference to structural dependencies, and various morphological features of syntactically related forms provide useful hints for morphological and syntactic disambiguation. From the development point of view, processing and disambiguation of Arabic depend in the following sources of information:

- The lexicon: provides basic and initial information about lexical items (grammatical attribute).
- Adjacency constraints: specify the compatibility or the incompatibility of two neighboring morphemes. For example, a preposition cannot be followed by a preposition.
- Morphological dependencies (Mel'tchuk 1988): describes the type and direction inflected from one constituent to another. As shown in Figure 1 a verb that follows the subject should agree in number and gender, thus the verb is morphologically

dependent on the subject. On the other hand, the subject is morphologically dependent on the verb in case ending.

- Syntactic dependencies (Mel'tchuk 1988): determine binary relations between the lexical items in the sentence. In Figure 1, the verb *hit* is the head of *two boys* (subject) and *hani* (object).

As shown figure 1, it is not necessarily that the syntactic dependent of a head is also morphologically dependent. *Hit* and *the two boys* are exhibiting mutual morphological dependencies.

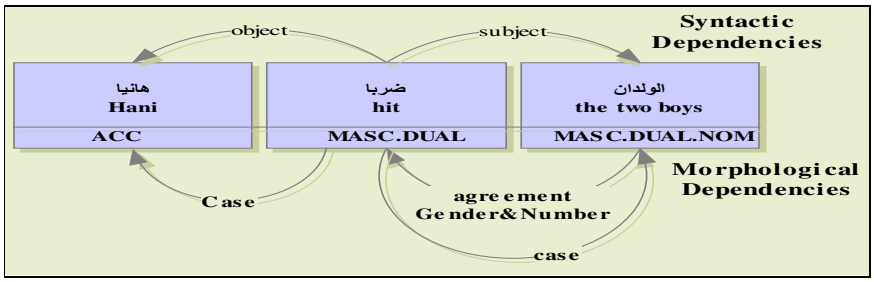


Figure 1: Example of morphological and syntactic dependencies

To demonstrate how the above information can be employed in disambiguation, consider the sentence shown in Figure 2. The ambiguity in the sentence is stemmed from the following two word forms:

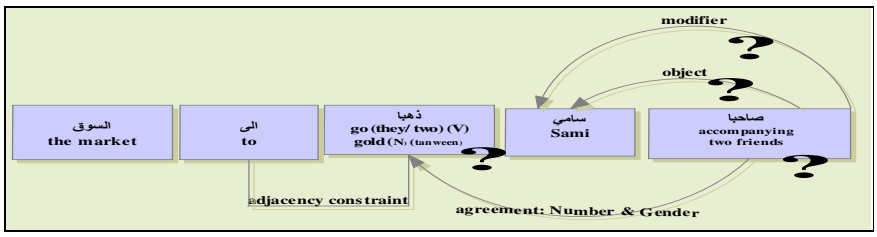


Figure 2: Example of ambiguity resolution

صاحب+ا ➔ (*accompanying) or ( two friends*)
ذهب+ا ➔ (*they went*) or (*gold* [accusative])

The disambiguation process is started by using the adjacency condition that a noun cannot be followed by a preposition (الى **to**). Thus, ذهبا (*they went*) is a verb (go) [MASC, DUAL] not a noun. (*Sami*) سامي (a named entity) cannot be the subject of the verb as there are no morphological dependencies (agreement in number). On the other hand, a morphological dependencies exists between ذهبا and صاحبا suggesting that it is (*two friends*) and that it is the subject. This solution is verified by the existence of a morphological dependency between صاحبا (*two friends*) and سامي (*Sami*): the suffix that indicates duality ending is ان (NOM), but when the noun is the first part of the IDAFA construction the suffix should be ا which is the case in the above sentence. So, *Sami* is the second part of the IDAFA construction.

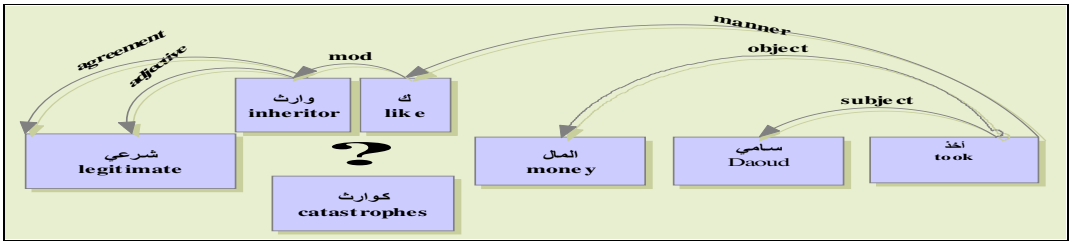


Figure 3: An example of syntactic dependencies disambiguation

In the sentence shown in figure 3, disambiguation is driven by syntactic dependencies. The verb (took) is the head of two dependents which are the subject and the object of (took). This is considered a NUCLEAR PROCESS that contains two participants in association with a 'process' element. Following (Dik 1989), any additional constituent is either:

- Indirect participant in a process.
- Additional information about a condition or circumstances pertaining to a process.

In Modern Standard Arabic, both indirect participants and circumstances are realized by two basic types of grammatical structure:

- Accusative nominals.
- Prepositional phrases of various kinds.

This is left us with one solution to "كوارث"; it is a prepositional phrase, meaning "like/such as + inheritor". Thus, it should be segmented correctly by recognizing the first character as a preposition (ka) and the rest of the morpheme as the word "وارث inheritor". This solution is verified by the existence of both syntactic and morphological dependencies with the word following it "شرعي legitimate".

In light of the above, it is clear that in some cases syntactic dependencies provide cues to perform segmentation and morphological analysis. On the other hand, morphological analysis and adjacency constraints are necessary to disambiguate syntactic structures. Thus, the pipeline model (where morphology is performed first and syntactic processing follows) will not suffice. In this model, a morphological analyzer provides all possible solutions to the syntactic parsing which leads to high magnitude of computational complexity of parsing. To demonstrate this, a word form in the Penn Arabic Treebank (ATB) has, on average, two morphological solutions (Nizar and Owen 2005). The complexity of any parsing algorithm will have a term order of:

$$\prod_{i=1}^{N} a_i$$

where $a_i$ is the number of alternative solutions of the *i*th word (Allan and Hanady 2008). Therefore, the average complexity of parsing a 20 words Arabic sentence using the pipeline model can reach up to 1048576 (2^20). Thus, linguistic information tend to be more effective at selecting between alternative solutions at the lower levels of the analysis and less effective at doing so at the higher levels (Macdonald, Pearlmutter et al. 1994).

Different systems that process Arabic with some degree of disambiguation are described in the literature (Othman, Shaalan et al. 2004; Allan and Hanady 2008; Attia 2008). All of them are rule-based systems adapting the pipeline model. Attia (Attia 2008) tried to reduce ambiguity by putting restriction on the lexical items during the morphological analysis phase. He

reported that his system took 141 minutes (CPU time) to parse a test suite of 229 sentences. The system described in (Allan and Hanady 2008) took a more restricted approach by selecting one solution during the morphological phase without having any syntactic information. Unfortunately, we could not use any of the above systems directly for comparisons.

On the other hand, statistical techniques have widely been applied to automatic morphological analysis for many languages including English, Turkish and Malay (Larkey, Ballesteros et al. 2007). The main challenge for such systems is that in Arabic, any particular word will appear less often than in English for a given text length and type. Thus, an Arabic datasets will have a higher degree of sparseness than comparable English counterparts (Goweder and De Roeck 2001). This is significant as it may affect the success of standard statistical techniques on Arabic data. However, Diab, Hacioglu, and Jurafsky (Diab, Hacioglu et al. 2004) reported a remarkable performance for Arabic morphological Analysis using Support Vector Machines (SVMs). They claim above 99% accuracy on tokenization and 95.49 accuracy on POS tagging. Their tools are trained on a sample of 4519 sentence of ATB. For the same size of English dataset, they reported a 94.97 accuracy on POS tagging, a result that contradict the fact that the token to type ratio is smaller for Arabic texts than for comparably sized English texts (Goweder and De Roeck 2001; Larkey, Ballesteros et al. 2007). Habash and Rambow (Nizar and Owen 2005) also reported high accuracy rates in their system for tokenizing and morphologically tagging Arabic words. They used similar approach reported in (Diab, Hacioglu et al. 2004), but by incorporating the Buckwalter morphological analyzer (Buckwalter 2002) into their system. However, Larkly, Ballesteros and Conner (Larkey, Ballesteros et al. 2007) reported that their simple light stemmer outperformed Diab's morphological analyzer. One of their explanations to this result is: "Arabic text contains so many definite articles that one could obtain the claimed >99% tokenization accuracy simply by removing *AL* from the beginning of words."

Having this in mind, we will take a different approach from previous work. Our system is a rule-based one, which is conceptualized by using dependency grammar. Although dependency grammar has its roots to the work of early Arabic Grammarians (Kitab al-Usul of Ibn al-Sarraj,d. 928), all of the existing (rule-based) Arabic processing systems are built on phrase structure theory.

In the next section, we will describe our synchronized model, which is able to perform morphological and syntactic processing of Arabic in as single, integrated and synchronized framework, thus allowing shared information to support disambiguation in multiple levels.

# 4 The Computational Model

Our system is coded using EnCo (Uchida 1999) which we used previously in developing the first Arabic-UNL enconverter. EnCo is a rule-based programming language specialized for the writing of enconverters (translators from a NL into UNL), and provided by the UNL center.

## 4.1 The UNL

Universal networking language (UNL)(Boguslavskij 2001; Uchida and Zhu 2001; Uchida and Zhu 2003; Boitet 2005) is a semantic, language independent representation of a sentence that mediates between the enconversion (analysis) and deconversion (generation). The pivot

paradigm is used: the representation of an utterance in the UNL interlingua is a hypergraph where normal nodes bear UWs ("Universal Words", or interlingual acceptions) with semantic attributes, and arcs bear semantic relations [13]. The sentence "Khaled bought a new car" can be expressed in UNL as:

***agt****(buy(icl>do(obj>thing),icl>purchase).@past.@entry, Khaled)*
***obj****(buy(icl>do(obj>thing),icl>purchase).@past.@entry, car(icl>automobile))*
***mod****(car(icl>automobile),new)*

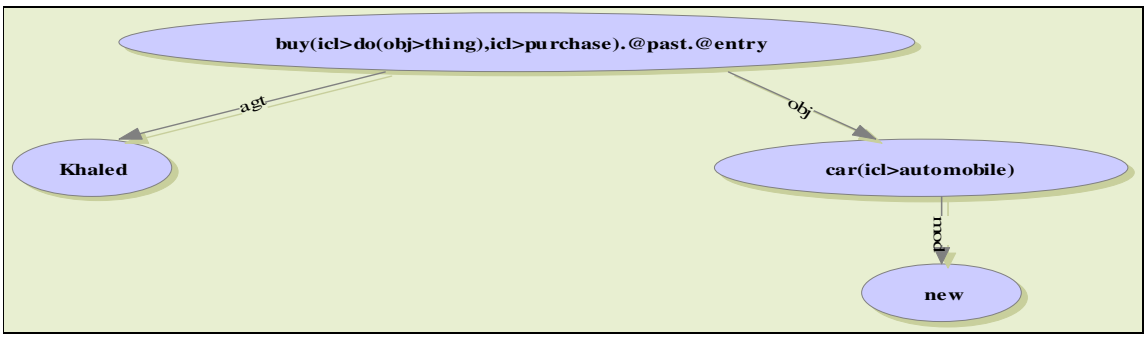


Figure 4: A UNL graph

## 4.2 The EnCo rule-based programming language

EnCo(Uchida 1999) is a rule-based programming language specialized for the writing of enconverters[1] . EnCo works in the following way. An input string is scanned from left to right. During the scan, all matched morphemes with the same starting characters are retrieved from the dictionary and become candidate morphemes. The rules are applied to these candidate morphemes, according to the rule priority, in order to build a semantic network for the sentence. The character string not yet scanned is then scanned from the beginning according to the applied rule; the process continues in the same manner. The output of the whole process is a semantic network expressed in the UNL format. If the dictionary retrieval or the rule application fails, it backtracks.

## 4.3 Overall Analysis Strategies Using EnCO

Developing EnCo rules requires a controlling mechanism that specifies which rule should be fired and which rules should not be fired. For that, we use tactical symbols written or removed from the input tape. Without using the KB (knowledge base), the only way to analyze Arabic is to depend on linguistic knowledge and on what exists in the sentence. Without having this controlling mechanism, this task would be impossible.
For example, suppose we have the following sentence:
ساق خالد السيارة الجديدة بسرعة كبيرة

[1] We use the term “enconverter”, and not “parser”, because the process involves a lexical transfer from the “lexical space” of the NL at hand (while many have several “levels” such are morphs, morphemes, word forms, lexemes, lemmas, derivational lexical families, and word senses) to the “lexical space” of UNL (the UWs, and their hierarchy).

*Khalid drove the new car at a high speed.*
To analyze this sentence correctly, we should discover the boundaries of the entities that exist in the sentence. Since "Khalid" is not followed by an adjective, it is allowed to be an agent of the verb "drive" and it is removed from the node-list (tape). On the other hand, since "car" is followed by an adjective which has the same gender, it is not allowed for "car" to be an object before handling the adjective first ("car" is a dependent of "drive", and "new" is a dependent of "car": it is not allowed to process the head before its dependents).

### *4.3.1 Disambiguation Mechanism*

At any particular moment in time, EnCo is in a describable configuration. Between this moment and the next discrete time stamp, the machine reads its input from the tape, refers to rules controlling its behavior, and considering both the input and the current configuration, determines what behavior to exhibit (i.e. erase/write on tape, move left, move right, create a an arc in the UNL graph, etc.), which determine the next configuration.

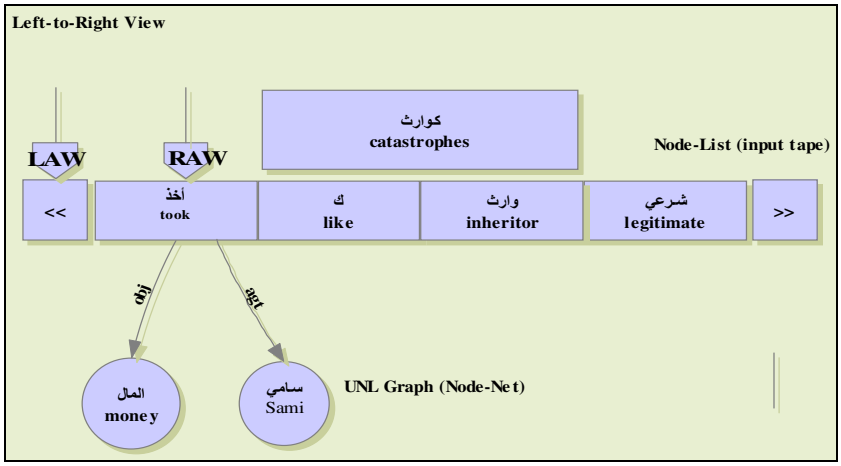


Figure 5: A describable configuration of EnCo

All information needed for disambiguation (adjacency, morphological dependencies, and syntactic dependencies, in addition to basic lexical attributes retrieved from the dictionary) is accessible at any moment of processing. This information is expressed by the symbols attached to each node in the input tape. Figure 5, demonstrates the availability of syntactic dependencies needed to disambiguate "كوارث". The engagement of the verb *took* in "agt" and "obj" relationships, provides information to the enconverter to perform the correct segmentation and word selection. More to the point, the enconverter will backtrack if it had done wrong selection. For example, consider the following rule:
?R{V1,obj,agt:::}{NDE:::}P255;

This rule will force the enconverter to backtrack when it reaches the following configuration: the left node is a verb engaged into two syntactic relations (agt and obj) and the right node is an entity or a noun. The UNL expression of (Sami took the money as a legitimate (valid) inheritor) is shown below:

```
;====================== UNL ======================
;أخذ سامي المال كوارث شرعي
[S]
agt(take(icl>event):00.@entry.@past,          Sami:04)
aoj:01(valid:0L,  inheritor:0G)
```

```
mod:01(like:0F.@entry,    inheritor:0G)
obj(take(icl>event):00.@entry.@past,          money:0B.@def)
man(take(icl>event):00.@entry.@past,          :01)
[/S]
;=================================================
;;Time  0.1          Sec
;;Done!
```

To implement this enconverter 1500 rules were coded. Long sentences have been analyzed accurately with this system.

To demonstrate this the following sentence was analyzed in only .3 Sec CPU time:

هزم الفريق السعودي هولندا على استاد فلسطين في مباراته الاخيرة في يوم الاحد وتمكن الفريق السعودي من تحقيق النصر بثلاث اهداف جميلة بعد ان لعبو بطريقة جماعية وبذلك يصل الفريق السعودي الى النهائيات محققا احلام الجمهور السعودي

*The Saudi team defeated Holland on Palestine Stadium in its last match in Sunday and the Saudi team was able to achieve victory by three wonderful goals as a result of their collective play, so the Saudi team reaches the finals achieving the dreams of the Saudi audience.*

# 5 Conclusion

During the development period of the Arabic enconverter, the number of lexical items added to UNL-Arabic dictionary reached 120,000 entries. This covers the UWs provided by UNL center and the most frequent Arabic lexicon. More sophisticated features are added to each entry to cover morphological, syntactic and semantics aspects. In designing those features, we took into consideration the analysis and generation processes. Functional words are also added to the dictionary along with all prefixes and suffixes needed for Arabic morphology.

The synchronized computational model of EnCo with Dependency Grammar provides us with the right mean to disambiguate a language such as Arabic. This approach outperform pipeline model in terms of computational time and accuracy. Our system disambiguate efficiently words that exhibit ambiguities across different categories (noun-verb ambiguity, particle- verb ambiguity), but less efficient in words that fall within same category (noun-noun, verb-verb). This is expected, as morphological and syntactic dependencies become less decisive in disambiguation in those situations. Our future work will focus in this issue.

# References

Allan, R. and M. Hanady (2008). "Towards including prosody in a text-to-speech system for modern standard Arabic." Comput. Speech Lang. **22**(1): 84-103.

Attia, M. A. (2008). Handling Arabic Morphological and Syntactic Ambiguity within the LFG Framework with a View to Machine Translation School of Languages, Linguistics and Cultures, University of Manchester. **Doctor of Philosophy**.

Boguslavskij, I. (2001). UNL from the linguistic point of view. MMA'01, SigMatics & NII, Tokyo.

Boitet, C. (2005). Gradable quality translations through mutualization of human translation and revision, and UNL-based MT and coedition. Universal Networking Language, advances in theory and applications. J. Carde˜osa, A. Gelbukh and E. Tovar. Mexico. **12:** 393—410.

Buckwalter, T. (2002). "Buckwalter Arabic Morphological Analyzer Version 1.0." Linguistic Data Consortium (LDC).

Covington, M. A. (1992). A dependency parser for variable–word–order languages. Computer assisted modeling on the IBM 3090: Papers from the 1989 IBM Supercomputing Competition. K. R. Billingsley, H. U. Brown III and E. Derohanes. Athens, Greece, Baldwin Press. **2:** 799–845.

Diab, M., K. Hacioglu, et al. (2004). Automatic Tagging of Arabic Text: From raw text to Base Phrase Chunks. HLT-NAACL.

Dik, S. C. (1989). The Theory of Functional Grammar, Foris.

Goweder, A. and A. De Roeck (2001). Assessment of a significant Arabic corpus. Arabic NLP Workshop at ACL/EACL 2001, Toulouse, France.

Kruijff, G.-j. M. (2002). Formal and computational aspects of dependency grammar: History and development of DG, SSLLI Course Notes, FoLLI, the Association of Logic, Language and Information.

Larkey, L. S., L. Ballesteros, et al. (2007). Light Stemming for Arabic Information Retrieval Arabic Computational Morphology. A. Soudi, A. v. d. Bosch and G. Neumann, Springer Netherlands.

Macdonald, M. C., N. J. Pearlmutter, et al. (1994). "The lexical nature of syntactic ambiguity resolution." Psychological view **101**(4): 467-703.

Mel'tchuk, I. (1988). Dependency Syntax: Theory and Practice, State University of New York Press.

Nizar, H. and R. Owen (2005). Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics. Ann Arbor, Michigan, Association for Computational Linguistics.

Othman, E., K. Shaalan, et al. (2004). Towards Resolving Ambiguity in Understanding Arabic Sentence. the International Conference on Arabic Language Resources and Tools, NEMLAR, Egypt.

Owens, J. (1988). The foundations of grammar : an introduction to medieval Arabic grammatical theory. Amsterdam J. Benjamins Pub. Co.

Uchida, H. (1999, 1999). "Enconverter Specifications." from http://www.undl.org.

Uchida, H. and M. Zhu. (2001). "The Universal Networking Language Beyond Machine Translation." from http://www.undl.org.

Uchida, H. and M. Zhu. (2003, 2003). "The Universal Networking Language specification, version 3.0." from http://www.undl.org.