Robust Rule-based Approach in Arabic Processing

Riadh Ouersighni
Military Academy
Fondek Jedid
Tunisia

ABSTRACT
A parsing system is a key element of many computer applications such as Information Retrieval, Knowledge Extraction and automatic translation. This paper presents a robust large-scale parser system for parsing Arabic sentences. From a practical point of view, the system is able to analyze real-world sentences thanks to a wide coverage of its linguistic knowledge that is realized within the DIINAR-MBC European project[1]. The parser is designed for robustness against difficult input that cannot be parsed correctly according to the standard grammar rules in the system, whether it is an extra-grammatical, ill-formed or unexpected input. Most systems use algorithmic approaches to robustness where parsing programs are extended to include heuristics to handle defect cases. This study adopts another solution based on a robust grammar-based approach for parsing. It consists of introducing robust rules in the grammar itself and relaxing constraints if necessary. The parser has been evaluated against real-world sentences and the results were very encouraging. The parser provides 95% coverage.

General Terms
Natural Language Processing, Arabic Processing

Keywords
Morphological analysis, Lexicon, Parsing, Formal grammar, Arabic language

1. INTRODUCTION
Parsing or syntactic analysis of natural language is the process of analysis by a computer of a sentence into its constituents, resulting in a parse tree showing their syntactic relation to each other according to the rules of a formal grammar. A parser is considered as the main component of a wide range of Natural Language Processing (NLP) systems ranging from man-machine Interface and Information retrieval system to automatic translation and speech processing. The problem of parsing Arabic language belongs among the most interesting and the most difficult task of Arabic processing. Indeed, despite over two decades of research effort, no practical domain-independent parser of Arabic has been developed. This is due to challenging features of Arabic language such as high degree of ambiguity, high degree of syntactic flexibility, complexity of the syntax, and omission of diacritics (vowels) in written Arabic. A number of parsers for Arabic have been made in recent years. But there is still no robust parser available for Arabic with sufficiently wide coverage. Most systems simply select types of syntactic phenomena for treatment, with considerable lexical limitations. But real world texts like article from newspaper, abstract from scientific journals or web pages usually contain all sorts of sentences which cause problems for parsers in assigning a suitable structure.

Robustness or fall-back technique is a key issue in nowadays NLP technology and a necessary precondition for building parsers able to tackle the difficult input. In real world applications, the parser should be able to deal with ill-formed sentences that cannot be parsed as a unified structure: sentences with grammatical errors and ellipses, long and complex sentences, but also some grammatical sentences that cannot be parsed owing to the presence of unknown words or to a lack of completeness in the grammar.

The need for robust Arabic parsers with a wide coverage is still increasing, especially with respect to application driven natural language processing systems such as Information Retrieval and Knowledge Extraction. For such applications, it is useful to have a parser that is able to assign a best partial parse to unexpected input in case a full parse cannot be attained, so that a maximum of information is saved.

2. RELATED WORK
Designers of application-oriented text processing systems have adopted a number of strategies for robust parsing. Some of them incorporate a robustness method at the algorithmic level. In [16] Lavie describes a parsing strategy based on GLR* parsing technique. A GLR* parser can parse almost any input sentence by ignoring unrecognizable parts of the sentence. The basic idea is to skip words that cause problems during the parsing process. The parser returns the analysis with the fewest skipped words. This way, it is guaranteed that a maximum of information is returned.

In [28] Strzalkowski presents a Tagged Text Parser extended with a skip-and-fit-recovery. When the parser reaches a predefined time-limit, it skips the problematic input and continues to recognize the rest of the input. When the end of sentence is detected, the parser tries to fit the recognized constituents into a complete parse tree.

Other strategies are based on statistical approaches. The technique presented in [17] consists of using probabilistic predictions to predict which grammar rules are likely to lead to an acceptable parse of the input. The algorithm calculates a number of probabilities with the phrase structure rules. If the probabilities exceed a certain limit, the program will mark the sentence as ungrammatical and it will produce a set of constituents that will probably lead to a parse with a higher probability. In [6] and [25] a robust method for predicting reading times is reported. Robustness first comes from the conception of the difficulty model, which is based on a morpho-syntactic surprisal index. This metric is intrinsically robust (because relying on POS-tagging instead of parsing). Robustness also concerns data analysis: he proposed to enlarge the scope of reading processing units by using syntactic chunks instead of words. As a result, words with null reading time do not need any special treatment or filtering.

[1] DIINAR-MBC is the acronym of "Dictionnaire INformatisé de l'ARabe, Multilingue et Basé sur Corpus" – project n° 961791 of the INCO-DC program, European Commission [9]. A part of this system was realized within the DIINAR-MBC project [21, 23].

All these techniques imply in most cases adjustment of the underlying parsing strategy: unknown words are automatically skipped, problematic fragments of the input is partially parsed.

Another solution to the problem is to use a rule-based knowledge approach. This strategy has been successfully used in several systems [3, 7, 14, and 19]. It consists of introducing robustness into the grammar itself rather than equipping the parser algorithm with a set of adjustment procedures.

With regard to Arabic processing, much standard parsing systems has already been carried out [1, 10, 18, 26, 27, 29, 30] and many others. In contrast, there were less works reported on robust parsing. In [3], Attia presents an Arabic robust parser using robust grammar based approach. The system is developed in the XLE (Xerox Linguistics Environment) which allows writing grammar rules that follow the LFG formalisms. For robustness, the standard grammar is extended with some robust rules. When a complete parse is not found in the standard grammar, the robust grammar allows the sentence to be analyzed as a sequence of well-formed chunks. When tested on short sentences (10 to 15 words) randomly selected from a corpus of news articles, the parser achieved 92% coverage after applying robustness techniques such as non-deterministic tokenizer, morphological guessers and a fragment grammar.

Tounsi et al. [30] presented a method for parsing Arabic sentences using Treebank-based parsers and automatic LFG f-structure annotation methodologies. The modified approach learned ATB functional tags and merge phrasal categories with functional tags in the training data. The authors reported about 77% parsing accuracy on parsing Arabic sentences.

In [5] Ben Fraj et al presented a machine learning approach using an Arabic Treebank. The knowledge enclosed in this Treebank is structured as patterns of syntactic trees. These patterns are representative models of the Arabic syntactic components. They are both layered and rich structurally and contextually. They serve as an informational source for guiding the parsing process. The parser is progressive since it proceeded by treating a sentence into a number of stages equal to the number of its words. At every step, the parser affects the target word with the most likely patterns that represent it in the context where it is put. Then, it joins the selected patterns with those collected in the previous parsing steps in order to construct the representative syntactic tree(s) of the whole sentence. If more than one tree is proposed, all the analysis trees are sorted according to their appearance frequencies in the Treebank. The preliminary tests have yielded accuracy and f-score equal to 84.8% and 77.5%, respectively.

In [2], Al-Taani et al describes a top-down chart parser for parsing simple Arabic sentences with the Context Free Grammar (CFGs). According to the authors, the parser is tested on 70 sentences extracted from Arabic real-world documents and gave an average accuracy of 94.3%.

Bataineh et al. [4] implemented a top-down parser with recursive transition network for parsing Arabic. The system has been tested on 77 sentences and gave a performance of 85.6%.

3. SYSTEM ARCHITECTURE

In general, a parsing system incorporates three main components, namely, lexicon, morphological analyzer, and syntactic parser. As Arabic is a highly inflected and derived language, each component requires extensive study and exploitation of the associated linguistic characteristics. A brief overview of the system is given here and main components will be described in detail in the following sections. The architecture of the system is given in Fig1. In this architecture, the boxes are indicating the processes of the system and the arrows indicate the flow of information between system parts.

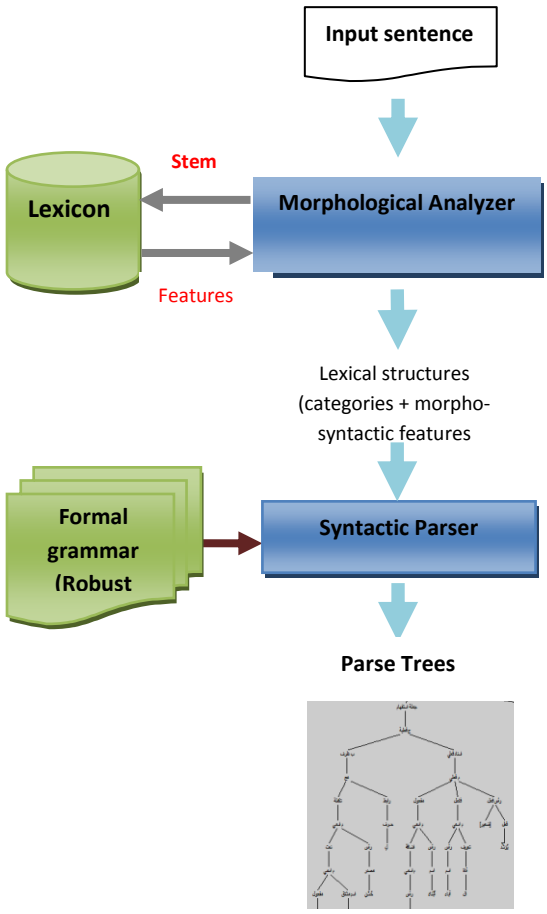

Fig 1: Overview of the architecture

The system is organized in a sequential modular system. Input sentence first passes through a morphological analyzer. The tokenization and the morphological analysis phase decompose words into a set of stem and affixes and associates a set of morpho-syntactic features to each recognized lexical unit. The morphological analyzer gives a list of all possible analyses for the words of the input sentence. Then the output of the morphological module is used as the input for the syntactic analysis phase. The syntactic analysis is carried out by a grammar-based parser which gives the syntactic structures for the input respecting to the formal grammar of the parser.

4. MORPHOLOGICAL ANALYZER

4.1 Analysis Strategy

The automatic processing of Arabic morphology is particularly challenging. This is due to the peculiarities of the Arabic language such as rich and complex morphology and highly ambiguous writing system since Arabic is typically spelled without short vowels and other diacritical markers.

The morphological analyzer uses a rule-based morphological segmentation algorithm and a large stem-based lexicon [23]. A written word is considered as a suite of morphemes. The

analyzer identifies these morphemes by decomposing them into proclitics[2], prefixes[3], stem[4], suffixes[5] and enclitics[6] and associates a set of features to each recognized lexical unit including possible segmentation(s), vowelled form(s), basic derivation forms (roots, lemmas, derived forms), potential grammatical categories and features such as gender, number, person, mood, case, voice, form, transitivity, human, deftness, etc.

Since the morphological analyzer uses a stem-based lexicon, the flexional forms were obtained by the morphological process of derivation, prefixation and suffixation. This complicates the morphological analysis algorithm but gives very interesting results in terms of processing time and memory space. The result of the analysis is given as a structure of lexical units that is used as the input for the parsing module.

4.2 The Lexicon

The morphological analyzer uses a large stem-based lexicon generated form DIINAR (DIctionnaire INformatisé de l'ARabe [9] and [24]). The DIINAR lexical Data Bases encompasses 19,457 verbs, 70,702 "derived verbal" entries (verbal nouns, active and passive participles, 'analogous' adjectives, nouns 'of time and operating place', 39,099, nominal stems, 445 tool-words and a prototype of 1,384 proper names.

Every entry is associated with morpho-syntactic features at word-level and ensuring grammar-lexis relations between the lexical basis of a given word-form and other word-formatives.

The total amount of minimal words (i.e. of lemmas with their prefix and suffixes) generated from the database is 7,774,938.

The lexicon contains:

- all the 121,522 unvocalized stem-entries of the DIINAR database
- all the vocalic schemes of each stem
- all possible combinations of (prefixes, suffixes) for each couple of (stem and vocalic scheme), and a set of features, containing morphosyntactic information.
- A specifiers of compatibility with possible clitics for each trio of stem, vocalic scheme, prefixes/suffixes combination.

The stem-based lexicon is organized in a letter tree structure. The principal advantage of the tree structure is that it greatly facilitates access while at the same time considerably reducing the lexicon size. The lexicon used for parsing is a 13 Mb binary file [24].

4.3 Testing the morphological Analyzer

The system has been tested on 37952 words from the ARCOLEX[7] corpus. Five text genres were used. The evaluation is preliminary; it can only serve as an indicator of the analyzer's performance and coverage. It gives us also an idea of the ambiguity rate encountered before parsing.

Table 1. Results of the Morphological analyzer Testing

Number of words	Arabic words	% Recognition	Number Analyses / word	% ambiguity	Segmen-tations / word	Number words / Second
37952	**35157**	**89%**	**3,93**	**77%**	**1,25**	**1315,28**

4.4 Discussion

The results showed a coverage rate of 89%. The average number of morphological analysis per word is 3.93, the average number of segmentations per word is 1.25 and 77% of the words are ambiguous. Compared to other languages (in French 20% and in English only 11% of the data is ambiguous [12]), Arabic words seems to be very ambiguous and the task of disambiguating is still very difficult. This high level of ambiguity can be explained with the fact that the morphological analyzer recognizes morpho-syntactic features involved in the structure of the word-form, such as verb transitivity, human or non-human complements, gender, number, mode in verbs and nominal cases, etc. such information strongly expand the number of analyses yielded by the system. The more features are added, the more analyses gain in accuracy, the higher the number of answers are found for a given word-form.

5. SYNTACTIC PARSING

5.1 Software Environment

The choice of software environment for the development of the parser is a decision that to a great extent influences the general behaviour of the system. There is usually a trade-off between the speed and efficiency and the use of a high-level linguistic formalism. The AGFL (Affix Grammars over Finite Lattice) system [13] was chosen for implementing the process of Syntactic Analysis, because AGFL allows for compactly and intuitively written grammars. It is a completely developed processing environment for grammar-based parsing. The grammars are automatically transformed into parsers, and important characteristics of the grammar (like left-recursion, rewrite rules that generate empty strings, etc.) are logged. More importantly, AGFL parsers are extremely fast (up to 2,000 words per second) and can be easily incorporated into larger software programs. Furthermore, the AGFL is proved to be appropriate for developing robust parser. Robust AGFL grammar has been successfully used in several full-text information Retrieval systems [14].

AGFL grammars are a restricted form of Context Free grammars. Context-Free production rules are extended with affixes (features) for expressing agreement between the parts of speech. These are passed as parameters to the rules of the grammar. The domain of every nonterminal affix is described by a set of Context-Free metarules producing a finite set of terminal affixes. The full syntax of AGFL is defined in [13].

The AGFL parsing is based on the Recursive Back-up [13]. It is a generalization of Recursive Descent Parsing to ambiguous grammars, extended with on-the-fly computation of features. According to [13], in the worst case, recursive backup parsers may exhibit exponential behavior. By establishing a time limit upon the parsing process, parsing of "expensive" sentences is aborted. In this way a trade-off between performance and coverage is established.

[2] Morphemes attached to the word that follows them. They represent coordinations, conjunctions, prepositions, etc.

[3] the prefixes include only the verbal morphemes (prefixed) of the imperfect tense.

[4] it is the nucleus of the word-form, which obtained after the identification of the other morphemes (proclitic, prefix, suffix, enclitic).

[5] are morphemes situated immediately after the stem.

[6] are morphems attached to a lexical category (noun or verb). In Arabic the enclitics are attached personal pronouns.

[7] ARCOLEX (Arabic Raw Corpora for Lexical-purpose) realized within the DIINAR-MBC Project.

5.2 Strategy for robustness

In order to be able to parse ill-formed or unexpected input the parser should be made robust. When building a robust parser it is necessary to make some preliminary considerations concerning the global strategy of the approach to the problem. This means that we have to decide whether we are going to build robustness techniques at the algorithmic level or else introducing robustness at the declarative level of the parser or alternatively using probabilistic and learning approaches.

The parsing algorithms are primarily designed to analyze "clean" grammatical input. In order to be able to handle difficult input, parsers are extended to include heuristics which implies adjustment of the underlying parsing algorithm.

It should be stressed that most problems with unrestricted texts are linguistics in nature. Maintaining the principle of separation between declarative and algorithmic components, it is obvious that for linguistic problems the solution must be considered at the declarative level. This means that we prefer a grammar-based solution by introducing robust rules in the grammar rather than equipping the parsing program with ad-hoc adjustment procedures and altering the behaviour of the parsing algorithm.

According to this approach inspired by [19], the system will first try to create complete syntactic structures for the sentence by means of the main rules; if this fails, try to analyze the ill-formed sentence as a sequence of well-formed chunks by means the robust rules.

5.3 Formal Grammar

5.3.1 Main Grammar

The main formal grammar, in which standard Arabic structures are described, is based on the EAG (Extended Affix Grammar) of Modern Standard Arabic developed by Everhard Ditters and presented in [10] and [11]. This grammar covers most frequent syntactic phenomena, allowing representing a syntactic structure of simple clauses and also the structure of certain types of complex sentences such as negative forms, elliptical forms, several interrogative forms, some kind of coordination and complex determiners. This grammar is translated in the AGFL formalism. The main grammar obtained encompasses some 850 syntactic rules.

5.3.2 Sub Grammar

Is a set of rules based on regular expressions ensuring the interface between output from morphological module and syntactic module. The morphological and syntactical levels of the system were carried out separately in two different environments and the two need to be brought together to produce a coherent system. The output of the morphological module is used as the input for the parsing module by means of an AGFL sub-grammar based internal interface. The first problem on the input side of the AGFL syntactic parser is the fact that the output of the morphological is a lexical lattice, instead of a string, which AGFL would expect. Since the AGFL parser only handles strings, lattice information is coded into string format. When a word corresponds with more than one category, all categories are copied into the output-string. It is the task of the syntactic analyzer to find out which category is suitable.

The basic idea of the integration technique is inspired by [8]. It consists of extending the core grammar by a sub-grammar, which describes categories rather than lexical stems. Therefore, the grammar has to describe categories as terminal nodes. Of course, the lexical stems have to be added as a kind of suffix to the category. Then the grammar has to be adapted in such a way that it only recognizes the output of the morphological component.

5.3.3 Robust rules

As mentioned in the previous section, this work adopts a tolerant grammar-based approach to robustness. In practice the main AGFL grammar is extended with rules that will perform the robust parsing. These rules should be developed that are more tolerant than standard grammar rules. The robust grammar encompasses some 70 rules.

5.4 Implementation of robustness

The AGFL formalism offers a number of mechanisms that are suitable for developing robust grammar [14]. First it is possible to define sequence with regular expressions for skipping or matching unexpected word. This technique is used at lexical robustness level for parsing unknown words, but also names, abbreviations, dates, etc. To do this, two nonterminals $SKIP and $MATCH are used, with regular expression as parameter. This makes it possible to describe open classes of words with a simple structure.

Another important mechanism is the best-first parsing called "graceful degradation". When a complete syntactic analysis is not found in the standard grammar, "graceful degradation" allows the sentence to be analyzed as a sequence of well-formed chunks.

A more important feature is the mechanism of stratification [14]. It means an ordering on the parsing and a partitioning into classes, suitable for avoiding unwanted ambiguities. This is realized by means the commit-operator in the rules. The commit-operator (!) is a special form of the (;) separating alternatives, which ensures that, if one of the previous alternatives succeeds and leads to at least one parsing, the subsequent alternatives are ignored. It can be used to indicate a preference of certain alternatives over others, of "correct" syntactic forms over doubtful ones.

In the next we will explain how we use these mechanisms. Robustness can furthermore be divided in lexical robustness and syntactical robustness.

5.4.1 Lexical robustness

Handling an unknown word in a sentence consist of assigning a category on the basis of its position in the syntactic structure and also the morphology of the word itself. First, we have to anticipate on which positions an unknown strings might occur. Unknown words can occur everywhere in the input, but the obvious positions are those positions on which open classes are expected: Nouns, Verbs, Adjectives and Names.

For example, the rule which rewrites a kind of Noun phrase in the parser looks as follows:

```
NP (HEADREAL, HUM, DEF, GENDER, NUMBER, THIRD, CASE) :
  PREDART,
  HEAD (HUM, DEF, GENDER1, NUMBER1, THIRD, CASE1),
  POM (DEF, GENDER2, NUMBER2, PERSON, CASE2),
       AGREEMENT IS ( HUM, GENDER1, GENDER2, GENDER,
       NUMBER1, NUMBER2, NUMBER).

  HEAD (COM, HUM, DEFNESS, GENDER, NUMBER, THIRD, CASE) :
       COMMON NOUN (DEFNESS, GENDER, NUMBER, CASE, HUM) !
       UNKNOWN NOUN.

     UNKNOWN NOUN :  $MATCH(".*").
```

As we can see, a HEAD of a Noun Phrase is rewritten into a COMMON NOUN or a nonterminal UNKNOWN NOUN. The mechanism of stratification with the commit-operator (!) in this rule make sure that this alternative will only apply when the previous alternatives did not lead to a parse.

The morphology structure of the unknown word can be used for assigning a plausible category by means of wild cards using regular expression. Words beginning in "ال"and ending in "ات", are classified as Nouns. So instead of recognizing Nouns with $MATCH(".*") it will be possible to recognize an unknown definite, fem, plural Nouns with the lexical robustness rule:

```
UNKNOWN NOUN (DEF, FEM, PLUR, CASE, HUM) :
        $MATCH("ال[*]+ات").
```

Since the suffix "ات" indicates plural, feminine and the prefix "ال" indicates deftness in Arabic.

The next lexical rule is also an example of lexical robustness to recognize unknown noun masculine plural:

```
UNKNOWN NOUN (DEFTNESS, MASC, PLUR, NOM, HUM) :
$MATCH("*.ون").
```

Word with prefix "ون" and suffix "ت" that occur in the grammar rule of the Verb phrase can be parsed as unknown verb (Indicative tense, Active voice, second, plural, …).

```
UNKNOWN VERB(INDIC, ACTIVE, 2, MASC|FEM, PLUR, COMPL) :
$MATCH("ون[*]+ت").
```

5.4.2 Syntactic robustness

The parser tries initially to recognize the complete sentence according to the main grammar, denoted by the first alternative. Failing this, it should recover all recognizable PHRASE PART and skip those fragments which are unrecognizable by means of the robust rules (island parsing). These rules serve thus serve as a last resort.

As an example, consider the root of a grammar from which a parser will be obtained which servers to extract noun Phrases from a sequence of utterances. Recognition proceeds from left to right.

```
SENTENCE : PREDICATION; ENONCIATION !
           UNKNOWN SEQUENCE.
```

According to the stratification technique (commit-operator), the parser tries to analyze the complete sentence as PREDICTION or ENONCIATION, denoted by the first alternative. If this will fail, the following alternative (UNKNOWN SEQUENCE), denoting the robust rule, will be tested as a last resort.

```
UNKNOWN SEQUENCE :  PHRASE PART ,
                    [UNKNOWN SEQUENCE].
PHRASE PART :
    NOUNPHRASE (DEFNESS, GENDER, NUMBER, PERSON, NOM) !
    VP (TENSE, PERSON, GENDER, NUMBER) !
    ADJP (DEF, GENDER, NUMBER, CASE)!
    ADVP !
    CL !
```

The nonterminal "UNKNOWN SEQUENCE" rewrites into one or more constituents, defined by the nonterminal "PHRASE PART". The rule above recognizes strings containing a number of "PHRASE PART" in any order.

```
NOUNPHRASE (DEFNESS, GENDER, NUMBER, PERSON, CASE) :
   NP (HEADREAL, HUM, DEFNESS, GENDER, NUMBER, PERSON,
CASE) !
   UNDEFINED NP.

UNDEFINED NP :  NP PART , [UNDEFINED NP].
NP PART : PREDET ,
  HEAD (HEADREAL, HUM, DEF, GENDER, NUMBER, THIRD, CASE) !

HEAD (HEADREAL, HUM, DEF, GENDER, NUMBER, THIRD, CASE) !
      POM (DEFNESS, GENDER2, NUMBER2, PERSON, CASE2)] !
      POM (DEFNESS, GENDER2, NUMBER2, PERSON, CASE2)].
     .....
```

6. PARSER EVALUATION

The evaluation has been carried out on a set of 200 real-world Arabic sentences randomly selected from the Arcolex corpus. This corpus has not been used to build up the parsing rules. Sentences have different sizes from 6 to 20 words (average sentence length is 10.52 words). The aim of this experiment was to investigate whether the parser is sufficiently robust for Arabic real-world applications.

TABLE 1. Evaluation Results

Number of sentences	200	
Parsed	141 (70.5%)	95%
Robustly parsed	49 (24.5%)	
Not parsed	10	5%
Average number of valid analyses per sentence	23.12	

As seen in Table 1, the parser provides 95% coverage. 141 sentences (about 70.5 %) were completely parsed according to the main grammar, 49 sentences (about 24.5%) were robustly parsed using the robust rules, and 10 sentences (about 5%) could not be parsed.

The average number of valid analyses per sentence is 23.12. This high level of ambiguity can be explained with the fact that the parser has a broad coverage (lexicon and grammar). In addition, the grammar was extended with robust rules which may cause additional ambiguity. The more linguistic knowledge are added to the system, the more analyses gain in accuracy, the greater the number of parses are found for a given sentence, the longer a parser takes in the analyzing. Ambiguity is a major problem for large-scale parser.

The performance of the parser could be compared to the Attia's Arabic LFG-based robust parser [3] which uses robust grammar-based approach similar to our strategy for robustness. The Attia parser is evaluated on 207 short sentences (10 to 15 words) and provides 92% coverage. 69 sentences (33% coverage) found a complete parse according to the standard grammar, and 138 sentences could not be completely parsed using the grammar alone. The coverage is raised to 92% when using a set of robustness techniques.

TABLE 2. Comparison with Attia parser

	Ouersighni parser		**Attia parser**	
Number of sentences	200		207	
Number words/sentence	6 to 20		10 to 15	
Parsed	70.5%	**95%**	33%	**92%**
Robustly parsed	24.5%		59%	
Not parsed	5%		8%	

7. CONCLUSION AND FUTURE WORK

This paper presents a large-scale robust parser for unrestricted Arabic sentences. The parser is intended for real-world applications such as Information Retrieval and Knowledge extraction where it is useful to have partial parses, even with low accuracy, for every input, so that a maximum of information is saved.

In order to deal with ill-formed input, the parser uses a robust grammar-based approach where the main grammar is extended with mechanisms which are suitable to produce robust parsing. The results observed in the experiment are very satisfactory in terms of coverage. The evaluation results showed an improvement in performance. The parser provides 70.5% coverage when using the main grammar alone. The coverage is raised to 95% when using a set of robustness mechanisms.

In contrast, the results showed a high level of ambiguity and a decrease in efficiency. This ambiguity may lead to an enormous amount of possible parses for an input sentence. It is obvious that robustness is a highly desirable property for natural-language processing systems. In practice, however, as the coverage increases, the ambiguity increases and the efficiency often decrease. Finding an optimum between coverage, efficiency, accuracy and ambiguity is therefore one of the bigger challenges in our future work.

The system is, of course far from complete, building a large-scale robust parser for Arabic texts is not a task which may be complete quickly. The work completed so far constitutes a base for further research. Future work will focus on the following related issues:

- Disambiguation in the morphological level (tagging)
- Disambiguation in the Parsing level
- Optimization of the grammar coverage
- Improving performance: reducing both parse time and ambiguities, and keeping them within an acceptable level.

8. REFERENCES

[1] Al-Daoud, E. and Basata, A, 2009, "A Framework to Automate the Parsing of Arabic Language Sentences," Computer Journal of The International Arab Journal of Information Technology, vol. 6, no. 2, pp. 191-195.

[2] Al-Taani, A., Msallam, M. and Wedian, S., 2010, "A Top-Down Chart Parser for Analyzing Arabic Sentences", The International Arab Journal of Information Technology (IAJIT).

[3] Attia, M., 2008, "Handling Arabic Morphological and Syntactic Ambiguity within the LFG Framework with a View to Machine Translation," UK, PhD Thesis.

[4] Bataineh, B. and Bataineh,E., 2009, An Efficient Recursive Transition Network Parser for Arabic Language, in Proceedings of the World Congress on Engineering WCE, UK, pp. 124- 127.

[5] Ben Fraj, F., Ben Othmane-Zribi, and Ben Ahmed, M., 2010, "Parsing Arabic Texts Using Real Patterns of Syntactic Trees" The Arabian Journal for Science and Engineering, Volume 35, Number 2C.

[6] Blache, P and Azulay, D.O., 2002, "Parsing ill-formed inputs with constraint graphs", Lecture notes in computer science ISSN 0302-9743, Computational linguistics and intelligent text processing: Mexico City, 17-23 February.

[7] Chanod, J.P., 2001, "Robust Parsing and Beyond", in J.C. Junqua and G. van Noord (eds.) Robustness in Language and Speech Technology, Dordrecht, Kluwer, pp. 187-204.

[8] Coppen.P.A., 1996, The use of AGFL in sequential Modular NLP systems in proceedings of the first AGFL Workshop, CSI (computing Science Institute Nijmegen), Nijmegen University.

[9] Dichy, J. and Hassoun, M., 2005, The DIINAR.1-"معالي" Arabic Lexical Resource, an outline of contents and methodology". In the ELRA Newsletter, Vol. 10, n°2, April-June 2005 : 5-10.

[10] Ditters. E, 1992, A formal approach to arabic syntax: the noun phrase and the verb phrase, Phd, Nijmegne University, Holland.

[11] Ditters. E, 2001, A formal Grammar for the description of sentence structure in Modern Standard Arabic, in proceedings of the Arabic Language Processing Workshop, Association for computational linguistics (ACL) 39th Annual Meeting and 10th Conference of the European Chapter, Toulouse.

[12] El-Beze M, Merrialdo B. Rozeron B. and Derouault A., 1994, Accentuation automatique de textes par des méthodes probabilistes, Technique et sciences informatique. Volume 13- n°6/1994, pages 797-815.

[13] Koster, C.H.A. and Oltmans. E. (Eds) 1996, proceedings of the first AGFL workshop, Computing Science Institute, Nijmegen.

[14] Koster, C.H.A. and Tiberius, C., 1996, AGFL Grammars for full-Text Information Retrieval, in proceeding of the NLDB.

[15] Koster. C.H.A., 1991, "Affix Grammars For Natural Languages", in H.Albas & B.Melichar (eds), Attribute Grammar Applications and Systems, SLNCS, 545, Springer, pp-469-484.

[16] Lavie, A., 1994, An Integrated Heuristic Scheme for Partial Parse Evaluation. In Proceedings of the 32nd meeting of the Association for Computational Linguistics (ACL 94), pages 316-319, Las Cruces, New Mexico, New Mexico State University.

[17] Magerman, D. and Weir, C. 1992, Efficiency, Robustness, and Accuracy in Picky Chart Parsing. In Proceedings of the 30st meeting of the Association for Computational Linguistics (ACL 92), pages 40-47, Newark, Delaware, University of Delaware.

[18] Mohammed, M.A. and Omar, N., 2011, "Rule Based Shallow Parser for Arabic Language", Journal of Computer Science 7, Science Publications.

[19] Oltmans, E., 1999, "A Knowledge-based Approach to Robust Parsing", the Netherlands, PhD Thesis.

[20] Othman, E., Shaalan, K and Rafea, A., 2003, A Chart Parser for Analyzing Modern Standard Arabic Sentence, MT Summit IX Workshop on Machine Translation for Semitic Languages: Issues and Approaches, USA.

[21] Ouersighni, R, 2001, "A major offshoot of the DIINAR-MBC project: AraParse, a morpho-syntactic analyzer of unvowelled Arabic texts". In ACL 39th Annual Meeting. Workshop on Arabic Language processing: Status and Prospect, Toulouse, pp. 66-72.

[22] Ouersighni, R, 2008, Towards Developing A Robust Large-Scale Parser for Arabic Sentences, in Proceedings of the International Arab Conference on Information Technology, pp. 15-18.

[23] Ouersighni, R. 2002, La conception et la réalisation d'un système d'analyse morphosyntaxique pour l'arabe : utilisation pour la détection et le diagnostic des fautes. PHD, Lyon2 University.

[24] Ouersighni, R. and Ghenima, M. 2009, Un système d'analyse morphologique à large couverture de l'arabe, actes de la 2ème Conférence internationale Systèmes d'Information & Intelligence Economique (www.siie.fr), IHE édition pp.559-572, 12-14, Hammamet, Tunisie.

[25] Rauzy, S. and Blache, 2012, P. Robustness and processing difficulty models. A pilot study for eye-tracking data on the French Treebank, in proceedings of Eye-tracking and NLP workshop, COLING-2012

[26] Shaalan, K, Farouk, A. and Rafea, A, 1999, Towards An Arabic Parser for Modern Scientific Text, In Proceeding of the 2nd Conference on Language Engineering, Egyptian Society of Language Engineering (ELSE), pp. 103-114, Egypt.

[27] Shaalan, K., 2010, "Rule-Based Approach in Arabic Natural Language" Processing. Int. J. Inf. Commun. Technol., 3: 11-19.

[28] Strzalkowsk, T. 1993, Natural language processing in large-scale text retrieval tasks, in the first text retrieval conference (TREC-1), D.K. Harman, ed., U.S. Department of commerce, National Institute of Standards and Technology, Washington, DC, 173-187, NIST Special Publication 500-207.

[29] Tounsi, L. and Van Genabith, J. 2010, Arabic parsing using grammar transforms. In: LREC - 7th conference on International Language Resources and Evaluation, 17-23 May 2010, Valletta, Malta.

[30] Tounsi, L., Attia, M. and Genabith, J., 2009 PARSING ARABIC USING TREEBANK-BASED LFG RESOURCES, Proceedings of the LFG09 Conference, Miriam Butt and Tracy Holloway King (Editors) CSLI Publications.