A Robust Parser for Unrestricted Arabic Text

Riadh Ouersighni
Virtual Reality and Information Technology
Military Academy
Fondak Jedid, Tunisia
riadh.ouersighni@gmail.com

Abstract—Parsing systems able to analyze natural language text robustly would be of great value in computer applications such as Information Retrieval, Knowledge Extraction and automatic translation. In real world applications, the parser must deal efficiently with difficult input that cannot be parsed correctly according to the standard grammar rules in the system, whether it is an extra-grammatical sentence, ill-formed or unexpected input. Most systems use algorithmic approaches to robustness where parsing programs are extended to include heuristics to handle defect cases. In this work we adopt another solution based on a tolerant grammar-based approach for developing a robust parser for unrestricted Arabic sentences. It consists of introducing robust rules in the grammar itself and relaxing constraints if necessary. The implemented parser lends itself to applications where large scale text processing is involved and robust, reliable, and relatively accurate syntactic analysis is necessary. The system is based on a wide coverage linguistic knowledge from the DIINAR-MBC European project¹. The parser has been evaluated against real-world sentences and the results were very encouraging. The parser provides 95% coverage.

Keywords: Natural language processing; Robust parsing; Arabic; Formal grammar.

I. INTRODUCTION

Natural Language Processing (NLP) is a field of Computer Science and Artificial Intelligence, concerned with the interactions between computers and human natural languages. As such, NLP is related to the area of Human-Computer Interaction. Many challenges in NLP involve natural language understanding that is, enabling computers to derive meaning from human language input. An automatic syntactic parser is a key element of a wide range of NLP systems ranging from man-machine Interface and Information retrieval system to automatic translation and speech processing. A number of research groups worldwide are currently developing such systems, varying in the depth of analysis from lexical parsing or tagging, through shallow parsing, to full parsers. However, despite over two decades of research effort, no practical domain-independent parser of unrestricted Arabic text has been developed. This is due to challenging features of Arabic language such as high degree of ambiguity, high degree of syntactic flexibility and absence of regular punctuation. 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 intricacies of real-world texts. 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.

II. RELATED WORK

Designers of application-oriented text processing systems have adopted a number of strategies for robustness. Some of them incorporate a robustness method at the algorithmic level. In [2] 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 [31] 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 [8] consists of using probabilistic

¹ DIINAR-MBC is the acronym of “Dictionnaire INformatisé de l’ARabe, Multilingue et Basé sur Corpus” – project n° 961791 of the INCO-DC programme, European Commission [16]. A part of this system was realized within the DIINAR-MBC project [26], [27].
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 [24] and [29] 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.

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 declarative approach. This strategy has been successfully used in several systems [5], [6], [9], [17] and [22]. 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, many standard parsing systems have been reported [11], [12], [10], [14], [4], [18], [21], [23] and many others. In contrast, there were less works reported on robust parsing. In [22], Attia presents an Arabic robust parser using XLE (Xerox Linguistics Environment) which allows writing grammar rules that follow the LFG formalisms. For robustness, he used a grammar-based approach. The standard grammar is extended with some robust rules. When a complete parse is not found in the standard grammar, the “FRAGMENT” 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. [20] 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 [15] 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 [1], 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. [3] 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%.

III. ROBUST PARSING OF ARABIC IN PRACTICE

Our parser is organized in a sequential modular system presented in [25]: first the morphological analysis phase decomposes Arabic words into a set of stem and affixes and associates a set of morpho-syntactic features to each recognized lexical unit. 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.

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, 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.
IV. THE MORPHOLOGICAL MODULE

Our morphological analyzer presented in [26] and [28] uses a rule-based morphological segmentation algorithm and a large stem-based lexicon. A written word is considered as a suite of morphemes. The analyzer identifies these morphemes by decomposing them into proclitics, prefixes, stem, suffixes and enclitics and associates a set of information 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, definiteness, etc.

Our analyzer uses a large stem-based lexicon that we have generated from the DIINAR Lexical Data Base. DIINAR [16] encompasses 19,457 verbs, 70,702 verbal-derived 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. The total amount of minimal words (i.e. of stems with their prefix and suffixes) generated from the database is estimated at 7,774,938.

The stem-based lexicon used for parsing is a 13 Mb binary file, containing 170,000 unwovelled verbal and nominal stems. Each stem is accompanied by its vocalic schemes, all possible combinations of (prefixes, suffixes), and a set of morpho-syntactic features.

In a previous paper [28], we presented an evaluation experiment of the morphological analyzer on a corpus of 37952 words. The results showed a coverage rate of 89%.

V. PARSING

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 was chosen for implementing the process of Syntactic Analysis, because AGFL allows for compactly and intuitively written grammars [5]. 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 [6].

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 [5].

The parsing is based on the Recursive Back-up [6]. It is a generalization of Recursive Descent Parsing to ambiguous grammars, extended with on-the-fly computation of features. However, 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.

A. Main formal 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 [11], [13]. 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. For our system, we have translated this grammar in the AGFL formalism. The main grammar obtained encompasses some 850 syntactic rules.

B. Robust grammar

As mentioned in the previous section, we adopt 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.

The AGFL formalism offers a number of mechanisms that are suitable for developing robust grammar [6]. 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 SMATCH are used, with regular expression as parameter. This makes it possible to describe open classes of words with a simple structure.

A more important mechanism is the best-first parsing called “graceful degradation” [6]. 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.

Another important feature is the mechanism of stratification [6]. 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.
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 AraParse looks as follows:

```plaintext
NP (headreal, hum, DEF, gender, number, THIRD, case) :
  PREDART,
  HEAD (hum, DEF, gender1, number1, THIRD, case1),
  POM (DEF, gender2, number2, person, case2),
  agreement in ( 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 “ل” and ending in “ت” 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:

```plaintext
UNKNOWN NOUN (DEF, FEM, PLUR, case, hum) :
  $MATCH("[ُ]ل[ُ]تَ").
```

Since the suffix “ل” indicates plural, feminine and the prefix “ت” indicates definiteness in Arabic.

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

```plaintext
UNKNOWN NOUN (deftness, MASC, PLUR, NOM, hum) :
  $MATCH("[ُ]ن[ُ]تَ").
```

Word with prefix “ن” 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, …).

```plaintext
UNKNOWN VERB (INDIC, ACTIVE, 2, MASC|FEM, PLUR, compl) :
  $MATCH("[ُ]ن[ُ]تَ").
```

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 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.

```plaintext
UNKNOWN SEQUENCE : PHRASE PART .
  [UNKNOWN SEQUENCE].
PHRASE PART :
  NOUNPHRASE (deftness, 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.

```plaintext
UNKNOWN SEQUENCE (DEF, FEM, PLUR, case, hum) :
  $MATCH("[ُ]ت[ُ]*[ُ]ل[ُ]تَ").
```

Word with prefix “ت” and suffix “ل” that occur in the grammar rule of the **Verb** phrase can be parsed as unknown verb (Indicative tense, Active voice, second, plural, …).
VI. Parser Evaluation

The evaluation has been carried out on a set of 200 unrestricted 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 our experiment was to investigate whether the parser is sufficiently robust for Arabic real-world applications.

<table>
<thead>
<tr>
<th>Number of sentences</th>
<th>200</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parsed</td>
<td>141 (70.5%) 95%</td>
</tr>
<tr>
<td>Robustly parsed</td>
<td>49 (24.5%)</td>
</tr>
<tr>
<td>Not parsed</td>
<td>10 5%</td>
</tr>
<tr>
<td>Average number of valid analyses per sentence</td>
<td>23.12</td>
</tr>
</tbody>
</table>

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 our parser could be compared to the Attia’s Arabic LFG-based robust parser [22] 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.

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.

In this work, we have presented a large-scale robust parser for unrestricted Arabic sentences that 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, our system uses a robust grammar-based approach. We have explained how the main grammar of the parser can be extended with mechanisms which are suitable to produce robust parsing. The results observed in our 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.

For an input sentence, our parser provides an optimal coverage which is 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.

VII. Conclusion and Future Work

In this work, we have presented a large-scale robust parser for unrestricted Arabic sentences that 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, our system uses a robust grammar-based approach. We have explained how the main grammar of the parser can be extended with mechanisms which are suitable to produce robust parsing. The results observed in our 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 decreases. 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.

REFERENCES


---

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


