SOFTWARE—PRACTICE AND EXPERIENCE
Softw. Pract. Exper. 2005; **35**:643–665
Published online 11 March 2005 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/spe.653

Arabic GramCheck: a grammar checker for Arabic

Khaled F. Shaalan*,†,‡

Institute of Informatics, The British University in Dubai (BUID), P.O. Box 502216, Dubai, United Arab Emirates

SUMMARY

Arabic is a Semitic language that is rich in its morphology and syntax. The very numerous and complex grammar rules of the language may be confusing for the average user of a word processor. In this paper, we report our attempt at developing a grammar checker program for Modern Standard Arabic, called Arabic GramCheck. Arabic GramCheck can help the average user by checking his/her writing for certain common grammatical errors; it describes the problem for him/her and offers suggestions for improvement. The use of the Arabic grammatical checker can increase productivity and improve the quality of the text for anyone who writes Arabic. Arabic GramCheck has been successfully implemented using SICStus Prolog on an IBM PC. The current implementation covers a well-formed subset of Arabic and focuses on people trying to write in a formal style. Successful tests have been performed using a set of Arabic sentences. It is concluded that the approach is promising by observing the results as compared to the output of a commercially available Arabic grammar checker. Copyright © 2005 John Wiley & Sons, Ltd.

KEY WORDS: Arabic natural language processing; grammatical checking; common Arabic grammar errors; grammar checkers

INTRODUCTION

Grammar-checking programs are now available for many languages. They promise to ease the burden of memorizing the rules of the grammar, style and punctuation [1]. A grammar checker program allows us to correct a mistake while the word or phrase is still fresh in our mind [2]. This software has many nice features. It offers to clarify an error and gives advice on how to avoid such an error in the future. In other words, the program not only corrects you, but it also offers informal lessons as you go—an easy and painless way to refresh your grammar knowledge [3].

*Correspondence to: Khaled F. Shaalan, Institute of Informatics, The British University in Dubai (BUID), P.O. Box 502216, Dubai, United Arab Emirates.
†E-mail: khaled.shaalan@buid.ac.ae
‡On leave of absence from Faculty of Computers and Information, Cairo University.

Received 28 April 2004
Revised 27 July 2004
Accepted 22 September 2004

In the long run, high-level language technologies will include the development of methods of syntactic and semantic computer processing (parsing as well as generation) as a necessary prerequisite for the development of natural language based industrial systems. At the current stage of development, grammar checkers constitute one of the best feasible applications for commercial usage of high-level language technology.

Word processing technology is the software application domain with the most immediate growth potential. The use of word processors leads to a whole class of writing errors [4]. Many popular word-processing programs have companion grammar checkers. The role of the grammar checker—whether integrated or standalone—is to try to intercept these errors.

The many different kinds of grammatical errors which may appear in written text can be categorized in several different ways. For the purpose of this paper, we propose the following two categories: *mechanic editing errors* and *cognitive errors*. Mechanic editing errors are due to cut-and-paste or insertion–deletion operations when using a word processor, which can be corrected by deleting an existing word or replacing it with a different one. The following list shows some grammatical errors that may occur when using a word processor [4].

- *Partially deleting old text when inserting new text*. This can result in grammatical nonsense: e.g. partially change 'allows you to read' to 'lets you read' and you may end up with 'lets you to read'; partially change 'on the other hand' to 'however' and you may end up with 'on the other however'.
- *Misspellings that accidentally produce real words*. For example, if you type 'go' when you mean 'to', or 'an' when you mean 'am' (press adjoining key on keyboard); or 'no' instead of 'on' (transposition of letters).
- *Selecting the wrong replacement text with a spell checker*. If you accidentally choose the wrong choice of several alternative words offered by a spell checker, every occurrence of the word in the document will be changed. This can be a very hard error to find.
- *Excessive letter or missed letter.* Since it is easier to write with a word processor, many people write more letters: e.g. if you type 'bee' when you mean 'be'. Also, they could write less letters: e.g. if you type 'red' when you mean 'read'.

Cognitive errors are more complicated. They occur due to lack of competence on the part of the language users to write a sentence that complies with the grammar rules, which can be corrected by replacing an existing word, inserting a new word, or moving one or more words. An analysis of grammatical errors in formal style Arabic is presented in the 'Analysis of common Arabic grammar errors' section.

Arabic grammar is a very complex subject of study; even Arabic-speaking people nowadays are not fully familiar with the grammar of their own language. Thus, Arabic grammatical checking is a difficult task. The difficulty comes from several sources [5]: (1) the length of the sentence and the complex Arabic syntax; (2) the omission of diacritics (vowels) in written Arabic 'at-taškīl'; (3) the free word order nature of Arabic sentence; and (4) the presence of an elliptic personal pronoun 'aḍ-ḍamīr al-mustatir'.

Logic programming plays an essential role in the natural language analysis process because it attempts to use logic to express grammar rules and to formalize the process of parsing. Logic grammars can be conveniently implemented in Prolog. Prolog-based grammars can be quite efficient in practice. A Prolog interpretation algorithm uses exactly the same search strategy as the depth-first top-down

parsing algorithm, so all that is needed is a way to reformulate grammar rules as clauses in Prolog. Parsing can make use of Prolog's built-in term unification, instead of the more expensive feature unification. For these reasons, Arabic GramCheck has been successfully implemented using SICStus Prolog on an IBM PC. It is based on deep syntactic analysis and relies on a feature relaxation approach for detection of an ill-formed Arabic sentence. The current implementation covers the basic grammar rules for the nominal sentence and the verbal sentence. Arabic GramCheck has some limitations, however.

- The grammar checker as described is targeted at a particularly well-formed subset of Arabic, rather than more colloquial dialects. Even standard newswire is likely to frequently include pre-verbal subjects and adverbials which are not considered in this paper. This restriction to a well-formed subset may be appropriate for people trying to write in a formal style.
- For practical reasons, the grammar checker resorts to default analyses for sentence structures that are expected to occur rarely. For example, starting with an indefinite inchoative, considered in traditional Arabic grammar as specified indefinites 'nakira muḵa'assasa', do exist, e.g. 'رجل مهذب أمين خير من ألف أمير خائن' 'rajulun muhaḏabun 'amīnun ḵa'ayrun min 'alfi 'amīrin ḵa'a'in', 'Better a well-bred trustworthy man than a thousand unfaithful Prince'. Nevertheless, the system considers it as a mistake.
- The grammar checker does not intercept punctuation errors that are related to incorrect use of spaces, commas and question marks.
- As Modern Standard Arabic text is usually written without vowels, the system does not detect incorrect diacritic signs.
- As the free word order nature of Arabic is usually dependent on semantics, grammatical errors due to incorrect word order are not always detected by the system. Moreover, the feature relaxation approach that we follow could be used, in some cases, to indirectly detect errors caused by wrong word orders. For example, consider the wrong noun–adjective order in the sentence 'اشتريت جميلة قطة' 'štaraitu jamīlatan qiṭṭatan', 'I bought a cat beautiful'. In this case, the system issues an error indicating wrong object category (the object should not be an adjective).

The rest of this paper is structured as follows. The next section presents a brief background about the aspects of the Arabic language. Then the focus turns to a review of the previous work on grammar checking for different languages. Our analysis of the common grammar errors for Arabic is then introduced, followed by a description of our proposed Arabic grammar checker. Next, we discuss how we evaluated our system and compare the results with a commercially available Arabic grammar checker. In a concluding section, we present some final remarks. Transliteration in this paper follows the convention explained in Appendix A. For abbreviations, see the list in Appendix B.

ASPECTS OF THE ARABIC LANGUAGE

The modern form of Arabic is called Modern Standard Arabic (MSA). MSA is a simplified form of Classical Arabic, and follows the same grammar. The main differences between Classical and MSA are that MSA has a larger (more modern) vocabulary, and does not use some of the more complicated

forms of grammar found in Classical Arabic [6]. For example, vowels are omitted in MSA such that letters of the Arabic text are written without diacritic signs.

As Arabic is strongly structured and highly derivational, understanding Arabic requires the treatment of the language constituents at all levels: morphology, syntax, and semantics. Each component requires extensive study and exploitation of the associated linguistic characteristics [7,8].

Arabic words are generally classified into three main categories [9].

- *Noun.* A noun in Arabic is a name or a word that describes a person, thing, or idea. Traditionally, the noun class in Arabic is subdivided into derivatives (that is, nouns derived from verbs, nouns derived from other nouns, and nouns derived from particles) and primitives (nouns not so derived). These nouns could be further sub-categorized by number, gender, definition, and case. This noun class also includes participles, adverbs, circumstantial accusative, pronouns, relatives, interrogatives, and demonstratives.
- *Verb.* The verb is any word that indicates the occurrence of an action. The verb class in Arabic is subdivided according to the following criteria: tense (past, present and future), with respect to the object (intransitive, transitive), structure (sound, weak), mood (perfect, imperfect, imperative), and voice (active, passive). Further sub-categorization of the verb class is possible using number, person and gender.
- *Particle.* The particle is any word that has no meaning unless it is combined with one of the other two categories. Usually, it has fewer letters. It can be considered neither a verb nor a noun. In Arabic, particles are divided into three categories according to the type of word they can precede. They can either precede a noun, a verb, or both. The particle class includes prepositions, conjunctions, interrogative particles, exceptions, and interjections.

The inflection and conjugation of the Arabic word is so sophisticated that they yield a complex word form. For this reason, most of the contemporary work in the field has been at the word level [10].

An Arabic sentence has two forms [5]:

- *Nominal sentence.* A nominal sentence is composed basically of two constructions: inchoative[§] (مبتدأ) and enunciative (خـبر). A nominal sentence can embed a verbal/nominal sentence as its enunciative. A nominal sentence can start with Inna/Kan and its sisters, which change its case ending (الأعراب).
- *Verbal sentence.* A verbal sentence is composed basically of two constructions: verb and subject. If the verb is transitive, it needs to have an object(s). In its passive voice it comprises a verb and a proagent (نائب فاعل).

An Arabic compound sentence is formed from a simple sentence followed by a complementary, such as conjunction form (عطف), quasi-preposition (شبه جملة), and annexation form (تركيب اضافى). Because Arabic is a flexible language, constituent order may vary and the constructs may be curtailed (محذوف).

[§]Refer to reference [11] for a translation of the Arabic terminology.

PREVIOUS WORK

A grammar checker is a complex program which needs a lot of research and linguistic resources [12]. These days, grammar checkers, although still far from perfect, are much better and easier to use. In fact, it is hard to ignore them.

There are three main approaches to implementing a grammar checker, namely, syntax based, statistics based, and rule based.

Syntax-based checking is described in reference [13]. Using this approach, a text is completely analyzed morphologically and syntactically. It requires a lexical database, a morphological analyzer and a parser. The parser assigns a syntactic structure to each sentence. The text is considered incorrect if the parsing does not succeed. According to the level of the linguistic analysis to which the error belongs, syntax-based checking can be classified as either a deep syntactic analysis or a shallow syntactic analysis. The *feature relaxation* technique is employed mainly in syntax-based checking, which relies on positive knowledge for detection and diagnosis procedure [14].

The advantage of the syntax-based approach is that off-the-shelf NLP resources such as lexicons, morphological analyzers and parsers can be used to do the analysis. Unfortunately, the checker will only recognize that the sentence is incorrect, it will not be able to tell the user what the exact problem is. For this, extra rules are necessary in order to either parse ill-formed sentences or apply a technique to features associated with linguistic fragments. If a sentence cannot be parsed using such an extra rule, it is incorrect.

Statistics-based checking is described in reference [15]. The availability of a large amount of text (called corpus) has motivated researchers to innovate statistical models to extract valuable linguistic knowledge from such text. Among statistical language tools are part of speech (POS) taggers and statistical parsers. Some grammar checking systems use statistical tools to implement various tasks to detect grammar errors.

A POS-annotated corpus is used to build a list of POS tag sequences. Some sequences (called N-Grams) will be very common (for example, *determiner*, *adjective*, *noun* as in *the old man*), others will probably not occur at all (for example, *determiner*, *determiner*, *adjective*). Sequences which occur often in the corpus can be considered correct in other texts; uncommon sequences could be errors. Actually, even ungrammatical permutations of words are still probable.

Statistics-based parsers need to be trained over tagged text to infer a grammar that fits (describes) the structure of sentences. However, statistical parsers bear the risk that their results are difficult to interpret: if the system raises false errors, users will wonder why their input is considered incorrect when no specific error message is given. In statistics-based checking, it is hard to implement a pure statistical system due to inherited shortcomings in the approach. Such systems can be augmented with rule-based techniques for describing errors and proposing corrective actions.

Rule-based checking matches a set of rules against a text which has at least been POS tagged. This approach is similar to the statistics-based approach, but all the rules are developed manually. The *error anticipation* technique is employed mainly in rule-based checking, which relies on negative knowledge for detection and diagnosis.

The rule-based checker approach has many advantages. A sentence does not have to be complete to be checked; instead the software can check the text while it is being typed and give immediate feedback. It is easy to configure, as each rule has an expressive description and can be turned on and off individually. It can offer detailed error messages with helpful comments, even explaining

grammar rules. It is easily extendable by its users, as the rule system is easy to understand, at least for many simple but common error cases. It can be built incrementally, starting with just one rule and then extending it rule by rule.

In the following, we present some of the successful systems that were cited in this endeavor.

Grammatifix is a commercial grammatical checker for Finnish that provides an explanation of the error and a suggestion for correction if possible. Grammatifix uses a two-level morphological analyzer. Part of speech disambiguation is performed at the next level of analysis by the application of constrained grammar (CG) formalism. It uses a surface syntactic parser for sentence analysis. The errors are detected by partial parsing [16]. Grammatifix is part of the Swedish Microsoft Office Package [17,18].

Granska is a hybrid system that utilizes both probabilistic and rule-based methods in grammar checking for Swedish [19]. The morphological processing is performed using a Hidden Markov Model (HMM) tagger that was trained over a large tagged corpus (Stockholm-UmeaCorpus—SUC). Detection rules were written to identify grammatical errors in the tagged text, which are designed to match expected writing errors in the input text. The system produces error descriptions and proposes a correction. Another set of accepting rules handles correct grammatical parts in order to avoid false alarms.

Scarrie runs a spelling and grammatical checker for Danish at the same time. Scarrie produces a full analysis for both grammatical and ungrammatical sentences. Scarrie parses ungrammatical input by relaxation of the parsing rules and by additional error rules applied on parsing results. The system uses a bottom-up chart parser of syntactic analysis [20].

Bokmal is a grammar-based grammar checker for Norwegian (NGC) [21,22]. The grammar checking is applied to input text that has been grammatically tagged and morphologically disambiguated. It uses the CG formalism that has been used to develop a Swedish grammar checker [17,19]. NGC is part of Microsoft Word in the Office XP package released in 2001.

GramCheck works in a detection–diagnosis–correction cycle and provides a grammar and style checker for Spanish and Greek [23,24]. A combined *feature relaxation* and *error anticipation* technique was adopted. It is based on a generalized use of Prolog extensions to highly typed unification-based grammars. These extensions, called constraint solvers (CSs), perform different Boolean and relational operations over feature values.

A prototype of a grammar-based grammar checker for Czech is described in reference [25]. The grammar checker is able to check errors in languages with a very high degree of word order freedom. The syntax analysis is applied to input text that is morphologically analyzed. If there is at least one syntactic inconsistency, the results are passed to the evaluation phase. Inconsistency is detected by the application of a grammar rule with relaxed constraints or an error anticipating rule. If there is a syntactic tree that contains a subtree with discontinuous coverage, the evaluation phase tries to decide if there should be an error message, a warning or nothing.

Several possibilities of using finite-state automata as a means for speeding up a grammar checker for Czech are discussed in [26]. This software is able to detect, by constraint relaxation, errors from a predefined set. The grammar allows feature violations and parsing of ungrammatical word sequences. The system does not employ a full analysis of the input sentence. The efficiency is gained by splitting the sentence (if possible) into clauses before the processing. It is possible to detect an error in one of the substrings (clauses) irrespective of the analysis results of the other one(s).

Most of the research on Arabic natural language processing is devoted to the morpho-syntactic analysis phase without paying any attention to the problem of grammar checking. However, the most recent version of Microsoft Office[¶] (2003) includes a grammar checker for Arabic in the bundle [27][‖]—it is the only Arabic grammar checker on the market. This grammar checker supports checking and correction of Arabic simple sentences, it is integrated with a spell checker and it has a unique feature that enables the errors to be corrected iteratively. This feature allows correction of multiple grammar errors in the same sentence. The punctuation correction in the Office 2003 grammar checker is a totally new feature for the Arabic language. This feature checks spaces, commas and question marks.

ANALYSIS OF COMMON ARABIC GRAMMAR ERRORS

In the literature, error analysis concerns only the most common Arabic grammatical errors without any indication of the frequency of occurrence of these errors; see, for example, [28]. This is intended to help Arabic writers to alleviate most of the grammatical problems that plague their writing. However, there is a need for a thorough study that answers questions like the following. Which errors are most frequent? Which errors for a particular language group (both native Arabic writers and learners of Arabic) are most frequent? Within a particular error type, are there differences in the kinds of errors produced by speakers of different languages? Unfortunately, we are not aware of any (either formal or informal) study that analyses the writing errors of either native Arabic speakers or learners. Moreover, we are not able to conduct an empirical study of Arabic as we do not have access to the hundreds of randomly chosen essays of students/learners of Arabic that would be required for such an analysis.

In order to investigate the possibility of developing a computational Arabic grammar checker, we analyzed and classified the common grammatical errors that occur when formulating an MSA sentence in a formal style. These errors were verified by Arabic specialists to be the most common Arabic grammatical errors. Tables I–III detail the possible grammatical errors as inspired by discussions with students who are native speakers of Arabic during an NLP course. These errors are representative of those encountered by the average word processor user when typing Arabic and are based upon a recent study [29]. For the sake of clarification, relevant Arabic error examples followed by their grammatical correct are given along with their classification. For each type of error, an erroneous example is explained within an ungrammatical Arabic sentence*. In addition, a morphological gloss is provided in square brackets.

[¶]Users of Microsoft Word, the most common word processor now in use, may already be reacting to the green wavy lines that underline potential errors in grammar and problems in style, as well as to the red ones that underline errors in spelling. Note, if you mistype a word but the result is not a misspelling (for example, typing 'from' instead of 'form' or 'there' instead of 'their'), the English spell checker will not flag the word. To catch such problems, we use the English grammar checker.

[‖]For more details, refer to http://www.microsoft.com/middleeast/arabicdev/office/office2003/Proofing.asp.

*The asterisk indicates an incorrect word or sentence.

Table I. Agreement errors.

Error type	Example	Correct version
Number and gender agreement between the inchoative and the enunciative	الجنود* يدافعان عن الوطن 'al-junūdu ydāfi'āni 'ani-l-waṭan [the-soldiers (pl) defend (dl) about the-country] The soldiers defend the country	الجنود يدافعون عن الوطن 'al-junūdu ydāfi'ūna 'ani-l-waṭan [the-soldiers (pl) defend (pl) about the-country] The soldiers defend the country
Number and gender agreement between the circumstantial accusative and the subject it modifies	جاءت بعض السيدات* تحمل أطفالهن jā'at ba'ḍu-s-sayyidāti taḥmilu 'ṭfālahunna [came some ladies carrying (sg) their-children] Some ladies came carrying their children	جاءت بعض السيدات يحملن أطفالهن jā'at ba'ḍu-s-sayyidāti taḥmilu 'ṭfālahunna [came some ladies carrying (pl) their-children] Some ladies came carrying their children
Number, gender, definition, and case ending agreement between adjective and the noun it modifies	الرجال* الكريم يساعدون الناس 'ar-rijālu-l-karīmu yusā'idūna-n-nās [the-men the-generous (sg) help people] Generous men help people	الرجال الكرماء يساعدون الناس 'ar-rijālu-l-kuramā'u yusā'idūna-n-nās [the-men the-generous (pl) help people] Generous men help people
Number and gender agreement between the demonstrative adjective and the noun it modifies	ذهبنا إلى هؤلاء *المعلم ḏahabnā 'ilā hā'ulā'i-l-mu'allim [we-went to those teacher(sg)] We went to those teacher	ذهبنا إلى هؤلاء المعلمين ḏahabnā 'ilā hā'ulā'i-l-mu'allimīn [we-went to those teachers(pl)] We went to those teachers
Gender agreement between a verb and the subject	* شرب البنت عصير البرتقال šariba-l-bintu 'aṣīra-l-burtuqāl [drank (m) the-girl juice the-orange] The girl drank (m) orange juice	شربت البنت عصير البرتقال šaribati-l-bintu 'aṣīra-l-burtuqāl [drank (f) the-girl juice the-orange] The girl drank (f) orange juice
Agreement between a verb tense and the use of specific particles	الرجال لن* ذهبوا إلى القرية 'ar-rijālu lan ḏahabū 'ila-l-qaryati [the-men not went to the-village] The men will not go to the village	الرجال لن يذهبوا إلى القرية 'ar-rijālu lan yaḏhabū 'ila-l-qaryati [the-men not go to the-village] The men will not go to the village
Case ending agreement between a number and its following descriptor	الفلاح زرع فدانين* قمح 'al-fallaḥ zara'a faddānain qamḥ [the-farmer grew two-fedans wheat (NOM)] The farmer grew two fedans of wheat	الفلاح زرع فدانين قمحا 'al-fallaḥ zara'a faddānain qamḥan [the-farmer grew two-fedans wheat (ACC)] The farmer grew two fedans of wheat

Table II. Wrong constituent forms.

Error type	Example	Correct version
Case ending of inchoative or enunciative	*المعلمين ضربا الولد 'al-mu'allimaini ḍaraba-l-walada [the-two-teachers (ACC) hit the-boy] The two teachers hit the boy	المعلمان ضربا الولد 'al-mu'allimāni ḍaraba-l-walada [the-two-teachers (NOM) hit the-boy] The two teachers hit the boy
Case ending of the noun in genitive	ذهبنا إلى* الحديقتان الجميلتان ḏahabnā 'ila-l-ḥadīqatāni-l-jamīlatāni [we-went to the-two-gardens (NOM) the-two-beautiful (NOM)] We went to the two beautiful gardens	ذهبنا إلى الحديقتين الجميلتين ḏahabnā 'ila-l-ḥadīqataini-l-jamīlataini [we-went to the-two-gardens (GEN) the-two-beautiful (GEN)] We went to the two beautiful gardens
Case ending of the circumstantial accusative, subject, or object	عادت الطائرتان* سالمتان 'ādati-l-ṭa'iratāni sālimatān [returned the-planes (dl) safe (dl, NOM)] The two planes returned safe	عادت الطائرتان سالمتين 'ādati-l-ṭa'iratāni sālimatain [returned the-planes (dl) safe (dl, ACC)] The two planes returned safe
Case ending the predicate of *Kana* or one of its sisters	كان المعلمون* مجتهدون kāna-l-mu'allimūna mujtahidūna [were the-teachers diligent (NOM)] The teachers were diligent	كان المعلمون مجتهدين kāna-l-mu'allimūna mujtahidīna were the-teachers diligent (ACC)] The teachers were diligent
Number and case ending of the noun that follows the interrogative particle *kam* (How many)	كم* تلاميذ الفصل؟ kam talāmīḏu-l-faṣl? [how-many students the-classroom] How many students are there in the classroom?	كم تلميذا في الفصل؟ kam tilmīḏan fi-l-faṣl? [how-many student in the-classroom] How many students are there in the classroom?
The verb should remain singular even though the subject is dual or plural	* يلعبون الأولاد في الحديقة yal'abūna-l-'awlādu fī-l-ḥadīqati [play (pl) the-boys in the-garden] The boys play in the garden	يلعب الأولاد في الحديقة yal'abu-l-'awlādu fī-l-ḥadīqati [play (sg) the-boys in the-garden] The boys play (sg) in the garden
Definition of inchoative	*رجل مهذب rajulun muhaḏabun [a-man polite] A man polite	الرجل مهذب 'ar-rajulu muhaḏabun [the-man polite] The man is polite
Declension of the simple and compound number	كتبت الرسالة *السادس *عشر لأختها في العراق katabati-r-risālata-s-sādisa 'ašara li'uḵtiha fi-l-'irāq [wrote-she the-message (f) the-sixteenth (m) to-her-sister in Iraq] She wrote the sixteenth message to her sister in Iraq	كتبت الرسالة السادسة عشرة لأختها في العراق katabati-r-risālata-s-sādisata 'ašarata li'uḵtiha fi-l-'irāq [wrote-she the-message (m) the-sixteenth (m) to-her-sister in Iraq] She wrote the sixteenth message to her sister in Iraq

Table III. Missing sentence fragments.

Error type	Example	Correct version
Missing the subject of a verbal sentence	* ذهب إلى الدار ḏahaba ’ilā-d-dāri [went to-the-house] Went to the house	ذهب الغلام إلى الدار ḏahaba-l-ġulāmu ’ilā-d-dāri [went the-boy to-the-house] The boy (or *any other animated masculine entity*) went to the house
Missing the object of a verbal sentence	* فتح الولد fataha-l-waladu [opened the-boy] The boy opened	فتح الولد الباب fataha-l-waladu-l-bāba [opened the-boy the-door] The boy (*or any other animated masculine entity*) opened the door

THE ARCHITECTURE OF THE ARABIC GRAMMAR CHECKER

Arabic GramCheck is a syntax-based grammar checker for modern standard Arabic. The system is based on deep syntactic analysis and relies on a feature relaxation approach for detection of ill-formed Arabic sentences.

Arabic GramCheck helps the user to write a sentence by analyzing each word and then only accepting the sentence if it is grammatically correct. The main features of Arabic GramCheck are that it (1) performs complete grammatical analysis of sentences, and (2) checks the sentence for common grammatical errors, describes the problem, and offers suggestions for improvement. The design of the whole system is shown in Figure 1. The grammar checker is basically composed of two parts: an Arabic morphological analyzer and a syntactic parser extended to include a grammatical checking handler. The system is implemented in SICStus Prolog† 3.9 that runs under Microsoft Windows.

Morphological analysis and the lexicon

In order to implement the parser, a morphological analysis is performed on the inflected Arabic words. In a previous work [10], we described a morphological analyzer for inflected Arabic words. An augmented transition network (ATN) [30] technique was successfully used to represent the context-sensitive knowledge about the relation between a stem and inflectional additions. The ATN consists of arcs, each of which is a link from a departure node to a destination node, called states; see Figure 2. An exhaustive search to traverse the ATN generates all the possible interpretations of an inflected Arabic word. The morphological analyzer is implemented in Prolog and integrated with the parser.

The morphological analyzer consists of three modules: analyzer module, a lexical disambiguation module and a features extraction module. Figure 3 shows an example of analyzing the inflected Arabic

†Copyrighted in 2001 by SICS (Swedish Institute of Computer Science), Sweden (http://www.sics.se).

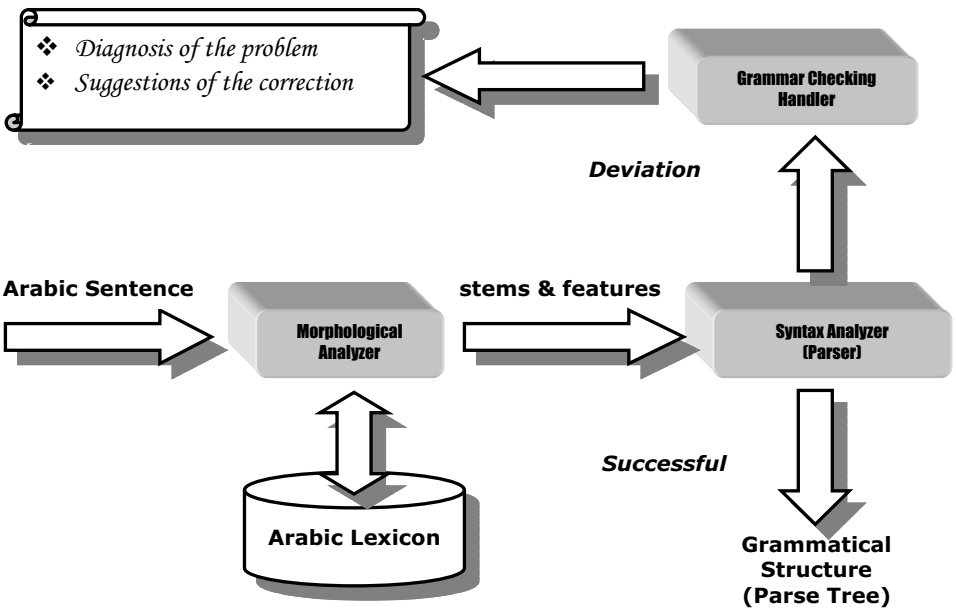

Figure 1. The architecture of the system.

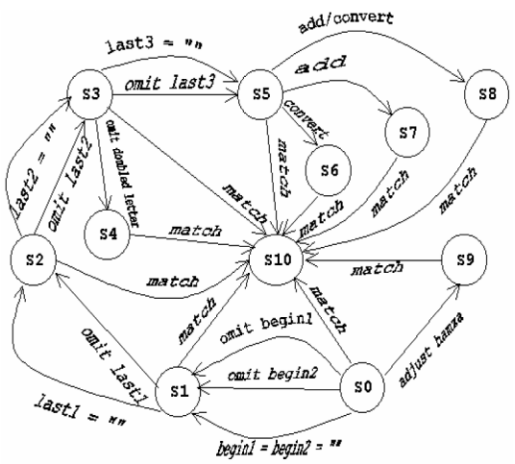

Figure 2. ATN representing the relation between the additions and the stem of an inflected Arabic word.

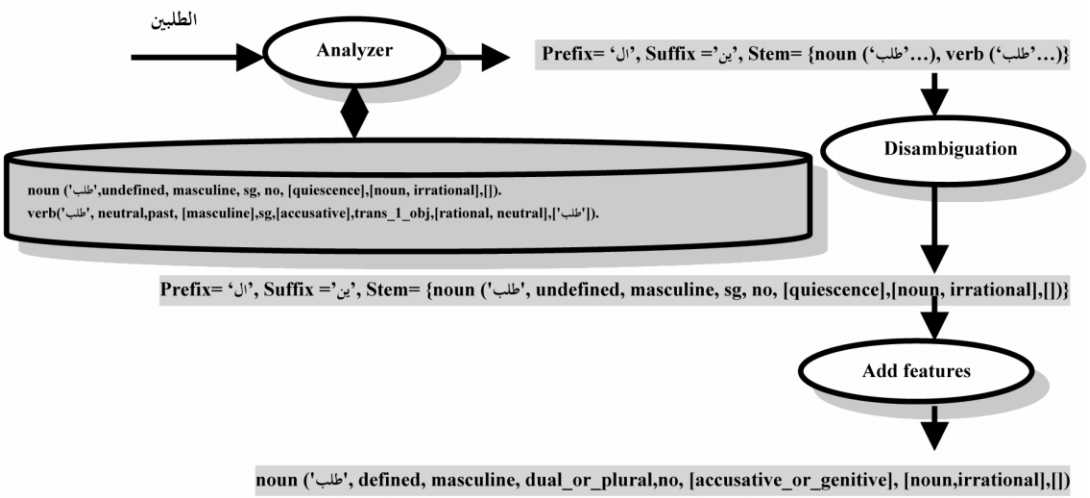

Figure 3. A morphological analysis example.

word 'الطلبين' ('al-talbaini). In this example, the word is analyzed into a verb and a noun. The former is discarded because the prefix is only used with nouns.

The lexicon

An Arabic monolingual lexicon was also needed to successfully implement the morphological analyzer. The lexicon is designed to reflect the word categories in Arabic. In our approach, we consider three basic morphological categories for Arabic—noun, verb, and particle—each with a different set of features. The system contains a dictionary of over 10 000 entries. Continued acquisition of lexicon entries is ongoing.

The lexicon features

There are two types of features in the lexicon: syntactic features that resolve syntactic ambiguity and lexical features that resolve lexical ambiguity. The default values of these features are stored in the lexicon and can be modified during the morphological analysis.

The lexicon entry is represented as a Prolog fact. The following list describes the forms of the lexicon entry.

1. **Verbs:** a verb has the following form:

```
verb (Stem, Voice, Tense, [Subject_Gender, Object_Gender],
Number, [End_case, Agent], Transitivity,
[Subject_rationality, Object_Rationality], Infinitive).
```


- *Syntactic features:*
  - **Voice:** passive/active.
  - **Tense:** past/present/future.
  - **[Subject_gender, Object_Gender]:** [male/female, male/female].
  - **Number:** singular/dual/plural.
  - **[End_Case, Agent]**: [accusative/nominative/genitive, subject/object/proagent].
  - **Transitivity:** intransitive/transitive_1_obj /transitive_2_obj; this feature is used to distinguish verbal sentence structures (commonly, verb–subject, verb–subject–object, and verb–subject–object1–object2).
- *Lexical features:*
  - **[Subject_Rationality, Object_rationality]:** [rational/irrational, rational/irrational]; this feature is used to distinguish subject from either object or proagent.
  - **Infinitive:** infinitive form; this feature is used to convert the weak letter of the verb in passive voice into its radical form in order to get the active voice of the verb.

2. **Nouns:** a noun has the following form:

```
noun(Stem ,Definition ,Gender ,Number ,Adjectivability ,
End_case ,[Category, Rationality] ,irregular_plural).
```

- *Syntactic features:*
  - **Definition:** defined/undefined/neutral.
  - **Gender:** masculine/feminine.
  - **Number:** singular/dual/plural.
  - **End_case:** [indeclinable/quiescence/accusative/nominative/genitive, without_noon: to indicate that the noun does not take suffix nūn ‘ن’ in case of dual or plural which means that the noun must be in a compound form].
  - **Irregular_plural:** broken plural form of the irregular noun/nil; this feature is used to link the singular noun entry with its irregular plural entry.
- *Lexical features:*
  - **Adjectivability:** yes/no; this feature takes yes if we can get the adjective form by adding the suffix yā’ ‘ى’; no otherwise.
  - **[Category, Rationality]:** [category is a noun type such as adjective, infinitive, demonstrative noun . . . etc., rational/irrational]; the category is needed because some noun types are not allowed grammatically to occur in a certain sentence position like the adjective in the position of subject.

3. **Particles:** a particle has the following form:

```
Particle (Stem, Category);
```

the only feature represented here is the **Category:** preposition, conjunction. . . etc.

Grammar checking: an extended variant of syntactic parsing

From a linguistic perspective, the current grammar can be characterized as unification-based grammar (UBG) formalism. With UBG, many grammatical errors can be described as violations of formal constraints between morpho-syntactic categories. The constraints may be intra-phrasal (e.g. phrase-internal agreement) or inter-phrasal (e.g. order between clausal elements). The central formal operation in UBG formalism is unification of feature structures. During the construction of the Arabic parser, feature structures are translated into Prolog terms. Because of this translation step, parsing can make use of Prolog's built-in term unification, instead of the more expensive feature unification. The current grammar covers the basic grammar rules for the nominal sentence and the verbal sentence. Each grammar rule has the form

```
rule(LHS,RHS):- constraints.
```

In our implementation, the error detection is embedded within the grammar rule and is based on the unification of the feature structures to determine the source of the grammar error. This is clarified by the following example:

```
rule(verb_phrase(Stem,Time,Gen,Num,Trans,Rat,Agent),[particle(Stem1,Cat),verb(Stem2,
Time,Tense,Gen,Num,[End_case|Agent],Trans,Rat,_)]):-
        (Tense==past->
                format('لايأتى الفعل الماضي(~w) بعد أداة النصب أو الجزم',[Stem2]),nl,nl,fail;true),
        (End_case==nominative->
                format('الفعل (~w) بعد أداة النصب أو الجزم لا يكون مرفوع',[Stem2]),nl,nl,fail;true),
        (\+var(Stem2),Cat==preposition->
                format('حرف الجر(~w) لا يسبق الأفعال',[Stem1]),nl,nl,fail;true),
        (\+var(Stem1),\+var(Stem2)->Stem=[Stem1,Stem2];true).
```

This rule says that in order to precede a verb with a particle (accusative or apocopative), some constraints must be satisfied:

1. the verb must not be in the past tense;
2. the verb must not be in the nominative case; and
3. the particle must not be a preposition.

If any of the above constraints is not satisfied, then the whole rule will fail and an error message reporting which type of error has occurred will be issued.

General search methods are not best for syntactic parsing because the same syntactic constituent may be re-derived many times as a part of different larger constituents. Chart parsing avoids re-parsing constituents by storing intermediate results in a data structure, called a 'chart'. So, for efficient implementation, we decided to implement the Arabic syntax analysis component as a chart parser [21]. We described our Arabic chart parser in [31]. The parser tries to analyze the Arabic sentence input. Similar to the work described in [25], there are three possible results of the analysis.

(a) The analysis is successful and no syntactic inconsistencies are found (at this stage of processing it is too early to use the term syntactic error, because in our terminology the term error is

reserved for something that is being announced to the user after the error detection)—in this case the sentence is considered to be correct and no message is issued. Syntactic ambiguity may arise. This ambiguity increases the range of possible interpretations of an Arabic sentence. In a recent study [32], we described our strategy for resolving ambiguities in understanding Arabic sentences. Syntactic ambiguity does not affect Arabic GramCheck capabilities because it detects errors that are related to ill-formed sentences.

(b) The analysis fails, the results contain at least one syntactic inconsistency. In this case it is necessary to pass the results to the grammar checking handler component. Then, the error message is issued to the user with suitable suggested corrections.

(c) The analysis fails and the handler cannot identify the error (probably due to the incompleteness of the grammar) and so cannot say anything about the input sentence. In such a case no error message is issued. Partial results are not used to indicate the possible source of an error. Partial results are misleading because often the error is buried somewhere inside the partial tree and no operations performed on partial trees can provide a correct error message. Besides, operations on (hundreds or thousands of) partial trees are very ineffective and they can also substantially slow down the processing of the given sentence.

A worked example

To explain the working of the system as a whole, we shall consider the following nominal sentence example:

التلميذات *مسرورة
’at-tilmīḏātu masrūrah
[the-students (pl, f) happy (sg, f)]
The students are happy

The following grammar rules are found relevant to the parsing of this sentence:

```
rule(simple_nominal_sentence(Stem,Gen1,Num1,Cat1),
        [inchoative(Stem1,Def,Gen1,Num1,_,Cat1),
        enunciative(Stem2,_,Gen2,Num2,_,_)]):-
(Def==undefined->format('يجب ان يكون معرفة (~w) المبتدأ', [Stem1]), nl,nl,fail;true),
(\+var(Gen1),\+var(Gen2)->
  ((Gen2==Gen1;Gen2==neutral;Gen1==neutral)->true
   ;format('اختلاف في الجنس بين المبتدأ (~w) و الخبر (~w)',
    [Stem1,Stem2]),nl,nl,fail);true),
(\+var(Num1),\+var(Num2)->
  ((Num2==Num1;Num2==neutral;Num1==neutral)->
    true
   ;format('اختلاف في العدد بين المبتدأ (~w) و الخبر (~w)',
    [Stem1,Stem2]),nl,nl,fail);true),
(\+var(Stem1),\+var(Stem2)->
    Stem=[Stem1|Stem2];true).
```


This rule states that the simple nominal sentence consists of an inchoative and enunciative and the following constraints must be satisfied:

1. The inchoative should not be undefined.
2. The inchoative and enunciative should neither disagree in number nor in gender.

```
rule(inchoative(Stem,defined,Gen,Num,End_case,Cat),
        [defined(Stem,Gen,Num,End_case,[Cat,_])]).

rule(defined(Stem,Gen,Num,End_case,[Cat,Rat]),
        [noun(Stem,defined,Gen,Num,_,[End_case|_], [Cat,Rat],_)]).
```

These two rules say that the inchoative should be a defined noun.

```
rule(enunciative(Stem,Def,Gen,Num,End_case,noun),
         [noun(Stem,Def,Gen,Num,_,[End_case|With_noon], [Cat,_],_)]):-
(Def==defined->
   format('الخبر(~w) لابد أن يكون نكرة ',[Stem]),nl,nl,fail;true),
   Cat\==annexation,
(Num==dual->
   With_noon\=[without_noon];true),
(End_case==accusative_or_genitive->
   format('الخبر(~w) لابد أن يكون مرفوع ',[Stem]),nl,nl,fail;true),
(End_case==accusative->
   format('الخبر(~w) لابد أن يكون مرفوع',[Stem]),nl,nl,fail;true).
```

This rule states that the enunciative is a noun and the following constraints must be satisfied:

1. The noun should not be defined.
2. The dual form should have the suffix nūn 'ن'.
3. The end case should be neither accusative nor genitive.

The lexicon entries of the words in the input sentence are

```
noun('تلميذ',undefined,male,sg,no,[quiescence],[noun,rational],['تلاميذ']).
noun('مسرور',undefined,male,sg,no,[quiescence],[adj,neutral],[]).
```

First, the morphological analysis is applied yielding the following structure:

```
[noun('تلميذ',defined,female,plural,no,[quiescence],[noun,rational],['تلاميذ' ]),
noun('مسرور',undefined,female,sg,no,[quiescence],[adj,neutral],[])]
```

Then, bottom-up chart parsing is applied. This is shown in Figure 4. Finally, an error message is issued indicating the disagreement in number between the inchoative and enunciative parts of the input nominal sentence.

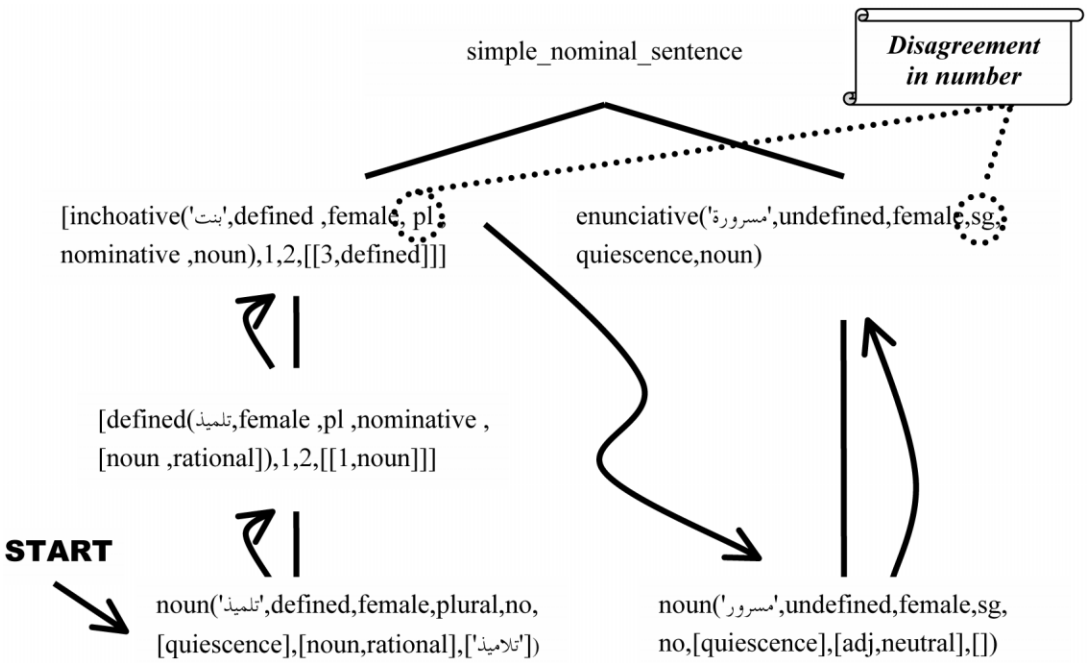

Figure 4. Bottom-up parsing with grammar checking. The parser discovers the nodes of the tree in the order shown by the arrows. The dashed lines show the source of the grammatical error.

ARABIC GRAMCHECK EVALUATION: COMPARATIVE RESULTS

The evaluation of NLP systems is classically divided into two main approaches: glass-box and black-box [33–35]. In black-box evaluation, the evaluator has access only to the input and output of the system under evaluation. In glass-box evaluation, the evaluator also has access to the various workings of the system and can thus assess each sub-part of the system. Component-based evaluation and detailed error analyses are also important types of evaluation [34,35].

In our work, we have chosen the black-box evaluation approach due to the fact that we want to compare our results with commercial systems, and, obviously, we do not have access to their inner workings. In such a setting, the evaluation may not be able to pinpoint the error source, however it will give an indication as to what subsystem is malfunctioning.

A set of 100 Arabic sentences was used to test Arabic GramCheck and evaluate its correctness. This set was prepared by an Arabic specialist, who is not a member of the Arabic GramCheck's team. The set included both grammatical and ungrammatical sentences, taking into consideration the coverage of both the grammar rules and the grammatical errors handled by Arabic GramCheck. The majority of these sentences were short and simple. We used short, simple sentences as they are easier to understand by the reader, they are easier to evaluate by the linguist, and they are suitable for comparison with the only commercially available grammar checker program.

Table IV. Correctness of Arabic GramCheck.

Sentence	Correct	Almost	Wrong	Total
Grammatically correct	10	0	0	10
Ungrammatical	86	1	3	90
Total (in percentage)	96%	1%	3%	100

Table V. Correctness of the commercially available Arabic grammar checker.

Sentence	Correct	Almost	Wrong	Total
Grammatically correct	5	0	5	10
Ungrammatical	34	17	39	90
Total (in percentage)	39%	17%	43%	100

The evaluation procedure was carried out by comparing the Arabic GramCheck results with those obtained on presenting the same sentences to an automatic grammar checking program available on the market. This comparison is a means of evaluating this Arabic GramCheck, rather than testing the commercially available Arabic grammar checker.

Of the 100 Arabic sentences, there were 10 grammatically correct sentences and 90 incorrect sentences. The average sentence length was four words and the longest sentence was 24 words long. The parser was capable of successfully parsing the longest sentence. The system includes 162 grammar rules.

A summary of the evaluation results is shown in Tables IV and V. The first column shows the category of the input sentences. A human reader rated the correctness of the output of both the Arabic GramCheck and commercially available Arabic grammar checker (correct, almost, wrong). These results are shown in columns 2–4 of both tables. The output was considered *correct* if the grammar checker gave a correct diagnosis of the ungrammatical sentence or accepted the grammatically correct sentence. The output was considered *almost correct* if the grammar checker detected inconsistencies in the ungrammatical sentence but did not give an explanation, the explanation was not correct, or the spelling checker flagged an error instead. The output was considered *wrong* if the grammar checker incorrectly detected an error in the grammatically correct sentence or did not detect the ungrammatical sentence.

The overall correctness is shown in the bottom row, which indicates the percentage of the input sentences marked as correct, almost, or wrong, in total. It shows 96% of the grammatical checking of Arabic GramCheck was correct compared with 39% of the commercially available Arabic grammar checker, and 3% of the grammatical checking of Arabic GramCheck was wrong compared with 43% of the commercially available Arabic grammar checker. Table VI shows the types of errors detected by Arabic GramCheck but missed by the commercially available Arabic grammar checker.

Table VI. Types of error detected by Arabic GramCheck, but missed by the commercially available Arabic grammar checker.

Type of error	No. of occurrences
Disagreement in gender or end case between the inchoative and enunciative اختلاف النوع أو الحالة الإعرابية بين المبتدأ و الخبر	8
Incorrect definition of the inchoative or gender of the enunciative عدم تعريف المبتدأ أو تنكير الخبر	6
Missing either the referent of the connected noun to the verb, the third person pronoun, or the construct replacing the subject عدم وجود ما يعود عليه الضمير المتصل بالفعل أو الضمير المستتر و القائمين محل الفاعل	5
Incorrect end case of the verb خطأ في الحالة الإعرابية للفعل	2
A verb in the past tense is incorrectly preceded by the accusative or apocopative particle الفعل الماضي مسبوق بأداة نصب أو جزم	3
A preposition incorrectly precedes the verb الفعل مسبوق بحرف جر	3
Disagreement in gender between either the verb and the subject or the verb and the pro-agent اختلاف النوع بين الفعل و الفاعل أو الفعل و نائب الفاعل	10
Disagreement in number, end case, or gender between the circumstantial accusative and the subject it modifies اختلاف العدد أو الحالة الإعرابية أو النوع بين الحال و صاحب الحال	3
Disagreement in number, end case, or gender between the adjective and the noun it modifies اختلاف العدد أو الحالة الإعرابية أو النوع بين الصفة و الموصوف	5
Incorrect case ending of the circumstantial accusative أخطاء في الحالة الإعرابية للحال	2
False alarm جمل صحيحة اعتبرها خطأ	5
Suffix nūn ‘ن’ is not omitted from either the irregular dual form or plural form in the case of annexation عدم حذف النون من نهاية المثنى أو جمع المذكر السالم عند الإضافة	3
Disagreement in number, end case, or gender between the permutative and the antecedent اختلاف العدد أو الحالة الإعرابية أو النوع بين المبدل و المبدل منه	2
Total	57

It can be concluded that Arabic GramCheck was shown to be superior to the commercially available Arabic grammar checker. The reason for this is that Arabic GramCheck is more accurate at detecting cognitive errors.

CONCLUSIONS

An Arabic grammatical checker is a complex program that needs extensive research and linguistic resources. In this paper, we reported our experiences gained from a project to develop Arabic GramCheck, a syntax-based grammar checker for modern standard Arabic. The system is based on deep syntactic analysis and relies on a feature relaxation approach for detection of ill-formed Arabic sentences. This useful tool is capable of detecting and suggesting improvements for certain common grammatical errors. Arabic GramCheck is basically composed of two parts: an Arabic morphological analyzer and a standard bottom-up chart parser including a grammatical checking handler. The system is implemented using SICStus Prolog on an IBM PC.

By reviewing the results obtained using Arabic GramCheck, it has been shown to be superior to a commercially available Arabic grammar checker. However, this experiment was limited to a set of simple Arabic sentences, manually prepared by an Arabic specialist.

It is hoped that the presented findings will be useful for development of Arabic grammar checkers, as well as for improving existing Arabic grammar checking software.

APPENDIX A. TRANSLITERATION OF ARABIC SOUNDS[‡]

Letter (E)	Transliteration	Letter (A)	Phonetic description
hamzah	’	أ	voiceless glottal stop
bā’	B	ب	voiced bilabial stop
tā’	T	ت	voiceless apico-dental stop
ṯā’	ṯ	ث	voiceless inter-dental fricative
jīm	J	ج	voiced lamino-alveolar palatal affricate
ḥa’	Š	ح	voiceless radico-pharyngeal fricative
ḵa’	ḵ	خ	voiceless dorso-uvular fricative
dāl	D	د	voiced apico-dental stop
ḏāl	ḏ	ذ	voiced inter-dental fricative
rā’	R	ر	voiced apical trill (roll)
zāy	Z	ز	voiced apico-alveolar fricative
sīn	S	س	voiceless apico-alveolar fricative
šīn	š	ش	voiced lamino-palatal fricative

[‡]Adopted from reference [36].

Letter (E)	Transliteration	Letter (A)	Phonetic description
ṣād	ṣ	ص	voiceless apico-alveolar emphatic fricative
ḍāḍ	Ǵ	ض	voiced apico-dental emphatic fricative
ṭā’	ṭ	ط	voiceless apico-dental emphatic stop
ẓā’	ẓ	ظ	voiced inter-dental emphatic fricative
‘ain	‘	ع	voiced radico-pharyngeal fricative
ḡain	ḡ	غ	voiced dorso-uvular fricative
fā’	F	ف	voiceless labio-dental fricative
qāf	Q	ق	voiced dorso-uvular stop
kāf	K	ك	voiceless velar stop
lām	L	ل	voiced apico-alveolar lateral
mīn	M	م	voiced bilabial nasal
nūn	N	ن	voiced apico-alveolar nasal
hā’	H	ه	voiced laryngeal fricative
wāw	W	و	voiced bilabial (rounded) velar glide
yā’	Y	ي	voiced palatal (unrounded) glide

Short vowels		
fatḥah	a	ـَ
Kasrah	i	ـِ
ḍammah	u	ـُ

Long vowels	Compound vowels
ā	au
ī	ai
ū	

APPENDIX B. LIST OF ABBREVIATIONS

Abbreviation	Full form
ACC	accusative (case)
Dl	dual
F	feminine
GEN	genitive (case)
M	masculine
NOM	nominative (case)
Pl	plural
Sg	singular

REFERENCES

1. Hartigan S. Grammar checker useful, but beware, hyperdispatch, computing and network services. University of Alberta, 1998. Available at http://www.ualberta.ca/CNS/PUBS/hyperDispatch/hyperDispatch19/grammar.html.
2. Johnson E. The ideal grammar and style checker. *TEXT Technology* 1992; **2.4**:3–4.
3. Harriehausen B. The computer as a 'teacher' for grammar and style errors. *Literary and Lingustic Computing* 1991; **6**(4):47–57.
4. Hahne H. Writing tools, in using a computer in biblical and theological studies. *Tyndale Seminary*, Toronto, 1999. Available at: http://www.balboa-software.com/hahne/harry.html.
5. Shaalan K, Farouk A, Rafea A. Towards an Arabic parser for modern scientific text. *Proceedings of the 2nd Conference on Language Engineering, Egyptian Society of Language Engineering (ELSE)*, Egypt, 1999; 103–114.
6. Khoja S. APT: Arabic Part-of-speech Tagger. *Proceedings of the Student Workshop at the Second Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL2001)*, Carnegie Mellon University, Pittsburgh, PA, 2001.
7. Khayat M. Understanding natural arabic. *Proceedings of the First KFUPM Workshop on Information & Computer Science*, Saudi Arabia, 1996.
8. Shaalan K. Machine translation of Arabic interrogative sentence into English. *Proceedings of the 8th International Conference on Artificial Intelligence Applications*, Egyptian Computer Society (EGS), Egypt, 2000; 473–483.
9. Mokhtar H. An automatic System for English–Arabic Translation of Scientific Text (SEATS). *Masters Thesis*, Computer Engineering Department, Faculty of Engineering, Cairo University, 2000.
10. Rafea A, Shaalan K. Lexical analysis of inflected Arabic words using exhaustive search of an augmented transition network. *Software Practice and Experience* 1993; **23**(6):567–588.
11. Cachia P. *The Monitor: A Dictionary of Arabic Grammatical Terms*. Librairie du Liban: Beirut; and Longman: London, associated companies, branches and representatives throughout the world, 1973.
12. Bustamante F, Declerck T, Leon F. Towards a theory of textual errors. *Proceedings of the Third International Workshop on Controlled Language Applications*, CLAW2000, Seattle, WA, April, 1999.
13. Jensen K, Heidorn G, Richardson S. *Natural Language Processing: The PLNLP Approach*. Kluwer Academic Publishers, 1993.
14. Sanchez-Leon F, Bustamante R, Declerck. Integrated set of tools for robust text processing. *Proceedings of the Vextal Conference*, 1999; 1–7. Available at: http://project.cgm.unive.it/events/papers/decl.pdf.
15. Atwell E, Elliott S. *Dealing with Ill-formed English Text in the Computational Analysis of English*. Longman, 1987.
16. Abney S. Partial parsing via finite-state cascades. *Workshop on Robust Parsing at The European Summer School in Logic, Language and Information, ESSLLI'96*, Prague, Czech Republic, 1996.
17. Arppe A. Developing a grammar checker for Swedish. *The 12th Nordic Conference on Computational Linguistics, NODALIDA'99*, Nordgard T (ed.), Department of Linguistics, Norwegian University of Science and Technology, Trondheim, 2000; 13–27.
18. Domeij R, Knutsson O, Larsson S, Eklundh K, Rex A. *Granskaprojektet 1996–1997*. IPLab-146, Royal Institute of Technology, Stockholm, 1998.
19. Birn J. Detecting grammar errors with Lingsoft's Swedish grammar checker. *The 12th Nordic Conference in Computational Linguistics, NODALIDA'99*, Nordgard T (ed.), Department of Linguistics, Norwegian University of Science and Technology, Trondheim, 2000; 28–40.
20. Povlsen C, Sagvall Hein A, de Smedt K. Final project report. *Reports from the SCARRIE Project*, 1999. Available at: http://fasting.hf.uib.no/~desmedt/scarrie/final-report.html.
21. Allen J. *Natural Language Understanding* (2nd edn). The Benjamin/Cummings Publishing Company, 1995.
22. Johannessen J, Hagen K, Lane P. The performance of a grammar checker with deviant language input. *Proceedings of the International Conference on Computational Linguistics (COLING)*, Taiwan, 2002.
23. Bustamante F, Leon F. Is linguistic information enough for grammar checking? *Proceedings of the First International Workshop on Controlled Language Applications CLAW96*, Leuven, 26–27 March 1996; 216–228.
24. Bustamante F, Leon F. GramCheck: A grammar and style checker. *Proceedings of the 16th International Conference on Computational Linguistics*, Copenhagen, 5–9 August 1996; 175–181.
25. Holan T, Kubon V, Platek M. A prototype of a grammar checker for Czech. *Proceedings of the 5th Conference on Applied Natural Language Processing*, Washington, DC, 1997.
26. Oliva K. Techniques for accelerating a grammar-checker. *Proceedings of the 5th Conference on Applied Natural Language Processing*, Washington, DC, 1997.
27. Microsoft Office. Arabic proofing tools in Office 2003, White Paper, 2003. Available at: http://www.microsoft.com/middleeast/arabicdev/office/office2003/Proofing.asp.
28. Jassem A. *Study on Second Language Learners of Arabic, An Error Analysis Approach*. A. S. Noordeen: Kuala Lumpur, Malaysia, 2000.

29. El_Tatawey A. *Al-siha Al-luğawīya*. Arabic Linguistics Department, Faculty of Arts, Cairo University, 2002.
30. Woods W. Transition network grammar for natural language analysis. *Communications of the ACM* 1970; **10**:591–606.
31. Othman E, Shaalan K, Rafea A. A chart parser for analyzing modern standard Arabic sentence. *Proceedings of the MT Summit IX Workshop on Machine Translation for Semitic Languages: Issues and Approaches*, New Orleans, LA, 2003. Available at: http://www-2.cs.cmu.edu/~alavie/semitic-MT-wshp.html.
32. Othman E, Shaalan K, Rafea A. Towards resolving ambiguity in understanding arabic sentence. *International Conference on Arabic Language Resources and Tools*, Network for Euro-Mediterranean LAnguage Resources (NEMLAR), Cairo, Egypt, 22–23 September 2004; 118–122.
33. Hutchins J, Somers HL. *An Introduction to Machine Translation*. Academic Press: New York, 1992.
34. Nyberg EH, Mitamura T, Carbonell JG. Evaluation metrics for knowledge-based machine translation. Center for Machine Translation, Carnegie Mellon University, PA, 1993. Available at http://www.lti.cs.cmu.edu/Research/Kant.
35. Arnold DJ. Evaluating MT systems, December, 1995. http://c1www.essex.ac.uk/~doug/book/node75.html.
36. Wehr H. *A Dictionary of Modern Written Arabic* (3rd edn), Milton Cowan J (ed.). Librairie du Liban: Beirut, 1980.

