An interactive environment for creating and validating syntactic rules

Panagiotis Bouros*, Aggeliki Fotopoulou, Nicholas Glaros
Institute for Language and Speech Processing (ILSP),
Artemidos 6 & Epidavrou,
GR-151 25, Athens, Greece
{pbour,afotop,nglaros}@ilsp.gr

Abstract

Syntactic analysis is a key component in many Natural Language Processing applications. This is especially true when considering advanced spelling checkers, where the usage of contextual rules at the syntax level can significantly increase the spelling error detection and correction capability of such systems. The advantage of the contextual approach over the isolated-word approach becomes more clear in morphologically rich languages, in which it is very likely that a spelling error free word can, in fact, represent a misspelled word within a given context. In such cases, even a minimal set of syntactic rules can be proved very effective in obtaining high spelling performance levels. However, determining a consistent set of rules for spelling checking purposes is not always a straightforward task. In this paper, we design and implement an interactive linguistic environment for managing the grammatical and syntactic resources of an advanced spelling checker system for Greek.

1 Introduction

Checking human free text has always been a very important and challenging issue to address. There is a lot of work already done for lexical analysis of text in order to identify and tag, using dictionaries, the lexical units contained in a text.

This word-by-word approach is quite efficient for the automatic check of spelling errors, which render a word totally invalid or non-existent. This type of spelling errors is most prominent in languages with poor morphology. However, in highly inflectional languages, it is very common that a spelling error in a lexical type produces another lexical type, which is valid on its own. For example, in the sentence: "I listens to the music.", there are no misspelled words on their own, yet, the syntax is still incorrect, because the verb type, according to the subject, should be "listen".

Clearly, the latter type of spelling errors is totally missed out by the word-by-word approach as well as by all spelling checkers that rely on it. On the contrary, this is precisely not the case when a rule-based syntactic analysis of every phrase of the text being checked (phrase-by-phrase approach) is employed. Resolving this kind of spelling errors takes more than simply going through a lexicon to match a given token. This leads us to advanced spelling systems, the design and implementation of which is still challenging and necessary for morphologically rich languages.

Building advanced spellers, based on statistical approaches, may require the use of a corpus in order to extract n-grams (Knight 99), (Beaujard & Jardino 99) (in most cases up to 3-grams) and then apply statistical models to compute the occurrence probability of the n-grams and of the corresponding parent sentence. Of course, if the lexical pattern of a sentence is correct, but never occurred before, there lies the problem of mischaracterizing it as incorrect. This problem is only partially addressed by smoothing techniques.

On the other hand, the fundamentals of a syntactic analysis framework are a morphological lexicon and a set of syntactic rules. Each rule of the set defines a number of word environments, i.e. grammatical patterns, which are formed by acceptable combinations of grammatical categories. In this manner, after tagging the words of a given sentence, the checking procedure attempts to verify the presence of the defined grammatical patterns on specific segments of the tagged sentence, thus concluding on possible rule-violations owning to spelling errors.

The work presented in this paper is directly connected with the syntactic analysis. In particular, we tackle the problem of creating, managing, monitoring and testing syntactic rules from within an easy and user-friendly interactive environment. For this purpose, we have designed and implemented a special tool for the graphical, most of all, creation of rules for the advanced

* Current affiliation is National and Kapodistrian University of Athens (NKUA), Department of Informatics and Telecommunications.

spelling checker of (ILSP) (Symfonia) (Stathis & Carayannis 99) and, moreover, for monitoring their application and interaction on existing text corpora. Symfonia employs a context-based spelling check technique, in addition to the isolated word-based approach. Cases where words sound similarly but are spelt differently, e.g. /δόsis/ "δόσης" noun feminine (nominative of plural or genitive of singular) : payment and /δόsis/ "δώσεις" verb (2nd person of singular in Future Simple or 2nd person of singular in Subjunctive) : give, and in which the spelling depends on the grammatical identity of the word, can be resolved.

The rest of the paper is organized as follows. Section 2 discusses the objectives of the proposed environment, while section 3 presents its architecture. Section 4 describes the working environment of the tool and lists its functional features. Section 5 demonstrates a real world scenario of using the tool. Finally, in section 6 some concluding remarks and prompts for further work are given.

2 Objectives - Specifications

The main purpose of the work presented in this paper is to provide a supportive environment for fast generating a consistent set of syntactic rules optimized for advanced spelling checking processes. Through a user-friendly interface, this tool allows language specialists to create, view, edit, real-time test monitor and validate syntactic rules, while leaving them out of the underlying computer programming technicalities.

As far as rule generation and editing is concerned, the environment provides a graphical rule representation mechanism. We consider that a tree graphical representation is suitable for presenting the word environments, the decision and generally the context of a syntactic rule. Moreover, in order for the tool to be speller technology-independent, we provide an XML (xml)-based mechanism for storing the rule tree representations. Furthermore, the tool automatically transcribes the user-defined rules into ready-to execute speller code (according to the speller being targeted), thus, providing a test-bed for the fast generation of a robust syntax analyzer.

By means of rich enough monitoring information, the system enables the user to evaluate the application of rules either individually or in combination with other user-specified rules. Emphasis is given on the production of a detailed report depicting the lexical analysis of the text, as well as details on the application of the user selected subset of rules, in order to identify or handle potential misusage, conflicts etc.

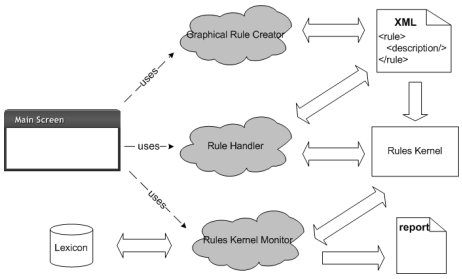

Figure 1: System architecture

3 Architecture

Figure 1 illustrates the architecture of the implemented tool. Each syntactic rule created by the Graphical Rule Creator is stored in an XML document and integrated in the Rules Kernel. The Rules Kernel is an extension of the kernel used by Symfonia speller with extra features for supporting insertion, handling and monitoring of the rules' application. Graphical Rule Creator is also used for editing and updating a syntactic rule. Furthermore, in order to provide additional handling functionality on the Rules Kernel, we have introduced the Rule Handle component.

Finally, the Rules Kernel Monitor is responsible for testing and reporting on the usage of a subset of the rules, integrated into the kernel, across real unformatted text. The monitor procedure relies on the speller's built-in lexicon for the lexical analysis and on the Rules Kernel for syntactic analysis, in order to generate a detailed report.

4 Working Environment - Functionalities

Figure 2 displays the main screenshot of the implemented tool, being the first window interacting with the user. This window consists of the list of rules that are integrated into the kernel. For every rule, its name, description and status are indicated. The status of a rule is either *enabled* or *disabled*, meaning that can be either taken into account by the monitor procedure or not. Moreover, all functionalities of the developed system are available through the menu options and the toolbar icons of this window. A detailed presen-

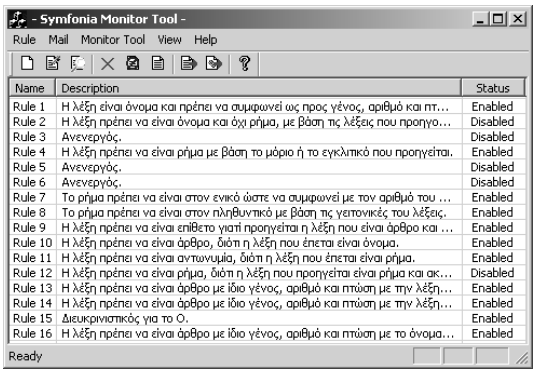

Figure 2: Main screen

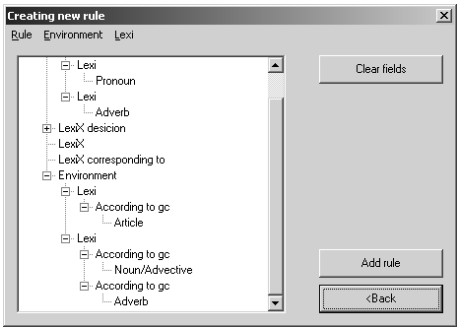

Figure 3: Rule tree

tation of the working environment and the implemented functionalities can be found in (Bouros 05).

4.1 Rule Handling

Rule handling mainly pertains to the management of the Rules Kernel component. Thus, it permits addition of new rules, editing of the definition and of the status of an existing rule or simply its removal from the kernel. All these changes are reflected in the list of Figure 2.

1. **Create a new rule**. In order to create a new syntactic rule the user takes advantage of the rule graphic tree representation presented in Figure 3. Each rule is focused on a single lexi[1] called LexiX.

[1] The term *lexi* (lexis in plural) is used in this paper to denote the set of grammatical characteristics of a word - On the other hand *words* are simply the tokens of a sentence.

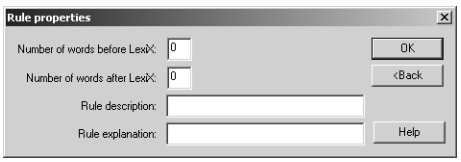

Figure 4: Specifying rule properties

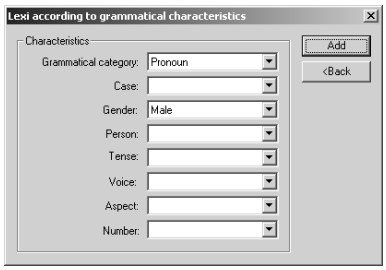

Figure 5: Specifying lexi's grammatical characteristics

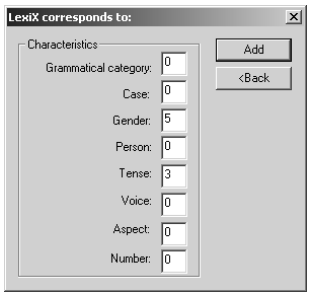

Figure 6: Specifying LexiX correspondence

First of all, the user should provide the description and the explanation of the rule. Explanation can contain parameters, denoted by $x for the LexiX or $+/- number, for a specific word of a sentence. These parameters are replaced by the corresponding words during the rule usage. User should also specify the number of words contained in rule environment before and after the LexiX position. The above rule properties are specified in the rule properties dialog depicted in Figure 4.

Next the user defines the valid combinations of grammatical characterizations, i.e. lexis, for LexiX, as well as the lexi which the new rule should conclude to. The definition of grammatical characteristics of each lexi is done through the dialog in Figure 5. The user can also restrict the application of the rule to a specific set of words. Through the dialog in Figure 6, the user can specify the adjacent words whose grammatical characteristics will be inherited to LexiX.

Finally, the user can specify the alternative environments of the new syntactic rule. Each environment is a set of lexis defined by their

grammatical characteristics (using Figure 5 dialog). The number of lexis contained in each environment must be equal to the total number of words specified in the rules properties dialog in Figure 4.

After completing the definition of the syntactic rule, the user integrates the new rule into the kernel. This is an automatic operation, which also constructs the XML rule file.

2. **Edit an existing rule**. The procedure of editing an existing rule is alike to the one of creating a new rule. Editing starts after the system has parsed the XML rule file and reproduced the tree representation of the rule (Figure 3). The user can modify the rule properties, characteristics and alternative environments and then choose to update the Rules Kernel and the corresponding XML file.

3. **Remove an existing rule**. Removal of an existing rule can be done through the respective menu option or toolbar icon located in the main screen (Figure 2).

4. **Disable/enable an existing rule**. By default, the status of a new rule is set to *enabled*. The status can be altered from the main screen in Figure 2 either to *disabled* or *enabled*.

5. **Export of existing rules**. Apart from XML format, a single or the entire set of the syntactic rules can be exported in a high level programming language code. The user has the option from within the environment to e-mail the resulted source code to the programmers group of the targeted syntactic speller.

4.2 Monitor

Efficient syntactic rules-based spell checking leads to the problem of generating and choosing syntactic rules that on the one hand optimize the performance of the spelling checker engine and on the other constitute a consistent set of rules. In trying to resolve this problem, there are many cases when a rule or a number of rules should be checked against a different set of rules, for identifying and minimizing potential rules conflicts and insufficiencies.

For this purpose, the system provides a monitor functionality for the evaluation of Rules Kernel

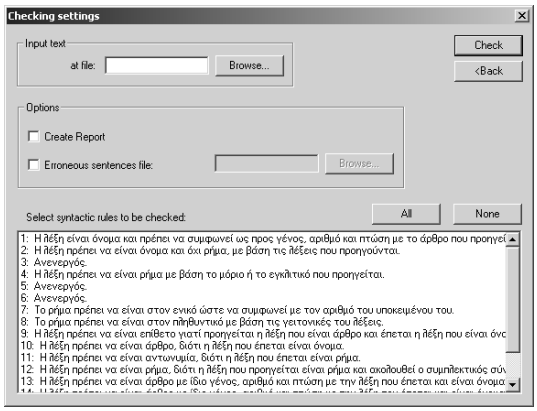

Figure 7: Checking procedure settings

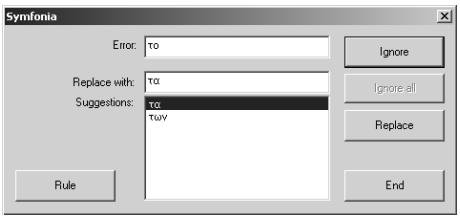

Figure 8: Interactive check dialog

while being on text documents. The system also takes advantage of the lexicon of (Symfonia) in order to perform the additional grammatical and lexical analysis required.

Rules checking can be done either interactively or automatically. In the first case, the user has to select one of the automatically generated system suggestions that attempt to correct the syntax error encountered. In the second case, the system by default adopts the first suggestion.

Nevertheless, in both cases the starting point is the same. Figure 7 presents the settings dialog of the checking procedure. In this dialog the user specifies the input text containing the sentences that should be checked and the set of syntactic rules that will be used, by picking them out from the rules list on the bottom of the dialog. The list contains all the rules integrated into the Rules Kernel except from the rule checking for simple spelling errors. This is a check that always takes place. Moreover the user can choose if the system will produce a report of the check and a document containing the erroneous sentences. In the latter option, the name of the output document should also be specified.

After having specified the settings, the rules checking begins. The procedure stops when an error is encountered and when in interactive mode.

The user is informed about the spelling mistake by Figure 8 dialog. This dialog is identical to the one used in the Symfonia advanced spelling checker. It denotes the misspelled word and proposes a number of alternative words. The user can either ignore this error or all of its subsequent occurrences, or replace the misspelled word or simply choose to end the checking procedure. In addition, the user can read the explanation of the rule used to detect the error.

A report regarding the checking of the document is produced at the end of the procedure if the user has requested so. The information contained in a report file is sentence-wise organized. In the beginning of the document, there is a list of the rules selected in Figure 7 to be taken into account. Then, for each sentence of the input document and for each error detected, a section is given that contains the grammatical analysis of the sentence words: lemma and grammatical category, the rules used in the checking of this sentence and the one that identified the error. In addition, the report lists the alternatives words proposed by the rule that detected the error, and also in case of an interactive check, it denotes the action of the user taken place in Figure's 8 dialog.

5 Real-World scenario

Let us assume that we wish to solve the ambiguity between the greek words for "more" and "which": "πιο" and "ποιο". Although these two words have the same phonetic transcription /pjo/, the first one is an adverb and the second is a pronoun. We create a syntactic rule with the following environment:

Lexi1 LexiX Lexi2

If LexiX is characterized by the ambiguity "πιο" - "ποιο" and Lexi1 is an article and Lexi2 is either an adjective or a noun or an adverb, then LexiX is an adverb, i.e. "πιο". Figure 3 illustrates the rule tree representing the created rule.

The previous rule resolves the ambiguity by rendering LexiX as an adverb. We can also define another rule for specifying that LexiX should be a pronoun, i.e. "ποιο". The environment of the required rule would be:

LexiX Lexi1 Lexi2 Lexi3 Lexi4 Lexi5

LexiX is "ποιο" if Lexi1 is an article, Lexi2 an adnoun, Lexi3 a noun, Lexi4 a particle and Lexi5 a verb. In addition, some or all of Lexi1, Lexi2, Lexi3 and Lexi4 can be missing.

6 Conclusion

Designing highly robust proofing tools for inflectional languages (Amaral *et al.*) is still an open issue. One fundamental approach to address this problem is to use grammar and syntax rules-based checking on a phrase-by-phrase basis. This, in turn, leads us to the problem of generating and choosing syntactic rules that not only optimize the performance of the spelling checker engine, but they also constitute a consistent set of rules. To this end, a purely linguistic tool was developed that lets language knowledgeable but computer programming unaware people to devise, build and test in real-time spelling checking processes whatever grammar and syntax rules they like, by means of graphical tree representations. At the same time plenty of monitoring information is provided by user-friendly interface in all phases of every syntactic rule life-cycle. Testing of the rules on large text corpora is also supported. The tool was implemented for the Greek language and for the Symfonia speller of ILSP. The environment has proven its value (e.g. rapid rule creation, efficient identification of potential rules conflicts etc.) after having thoroughly tested and evaluated by ILSP linguists group. Further work can be focused in converting the tool to a platform that can accommodate other spellers and support other morphologically rich languages.

References

(Amaral *et al.*) Carlos Amaral, Helena Figueira, Afonso Mendes, Pedro Mendes, and Claudia Pinto. A Workbench for Developing Natural Language Processing Tools.

(Beaujard & Jardino 99) Christel Beaujard and Michele Jardino. Classification of not labelled words by statistical methods. *Mathematics, Informatics and Social Science*, (147):7–23, 1999. in french.

(Bouros 05) Panagiotis Bouros. Technical report, Symfonia Monitor Tool manual, 2005. in greek.

(ILSP) ILSP. Institute of Language and Speech Processing, http://www.ilsp.gr.

(Knight 99) Kevin Knight. A Statistical MT Tutorial Workbook. prepared in connection with the JHU summer workshop, April 1999.

(Stathis & Carayannis 99) C. Stathis and G. Carayannis. title (in greek). In *2nd ELETO Conference: Hellenic Language and Terminology*, 1999. in greek.

(Symfonia) Symfonia. An intelligent spelling checker, http://www.ilsp.gr/correct.html.

(xml) xml. Extended Markup Language, http://www.w3.org/XML/.