Digital Scholarship in the Humanities Advance Access published July 14, 2016

Design and construction of the Greek grammar checker

Panagiotis Gakis and Christos Panagiotakopoulos
Department of Primary Education, University of Patras, Greece

Kyriakos Sgarbas
Department of Electrical and Computer Engineering, University of Patras, Greece

Christos Tsalidis
Neurolingo Language Technology Applications, Athens, Greece

Vassilios Verykios
School of Science and Technology, Hellenic Open University, Patras, Greece

Abstract

The aim of this paper is to present both the main parts of the designing and the implementation of a useful and user-friendly electronic tool, the Greek grammar checker. This tool carries out the function of analyzing morphologically and syntactically sentences, phrases, and words in order to correct syntactic, grammatical, and stylistic errors (Iordanidou, 1999, 2004). Our premise in order to deal with all these issues is the settings of Grammar (adaptation of Little Modern Grammar of Manolis Triantafyllidis), which is the formal grammatical codification of Modern Greek, since 1976 (Triantafyllidis, 1991). This paper also presents the formalism used (the Mnemosyne), a formalism that handles with the particularities of the Greek language that hinder the computational processing. This formalism has already been used to identify multi-word terms and to phrase grammars, aiming to automatically extract information. We tested the Greek grammar checker by giving texts that were to be evaluated both to the grammar checker and to a person. In the majority of cases, the human corrector accuracy is almost equal to the grammar checker one. As far as mistakes that have to do with the coherence of the text or with meaning are concerned, the human corrector was the only accurate corrector, not the grammar checker one (Gakis, 2015).

Correspondence:
Panagiotis Gakis,
Department of Primary Education, University of Patras, Rio, Greece
E-mail:
gakis@sch.gr

1 Introduction

Having realized that such a tool in Greek language didn't exist, the development of the software is based on the exhausting record, on the analysis, and on the formulation of the errors of writing speech. Moreover, for its development, the right software was chosen in order to describe the grammatical errors.

This paper also presents the formalism used (the Mnemosyne), a formalism that handles with the particularities of the Greek language that hinder

doi:10.1093/llc/fqw025

the computational processing. This formalism has already been used to identify multi-word terms and to phrase grammars, aiming to automatically extract information. In this way, all speakers (native or foreign) will be able to better understand not only the function of various parts of the Greek language system but also the way the mechanisms of linguistic analysis operate in the conquest and more broadly in the linguistic realization.

The fundamental cornerstone for the implementation is the electronic morphological lexicon[1] (Gakis *et al.*, 2012), a 5-level lexicon which consists of at least 90,000 entries that produce ~1,200,000 inflection types. These types inform about: (1) spelling, (2) morpheme information, (3) morphosyntactic information, (4) stylistic information, and (5) jargon (Kriaras, 1995; Mpampiniotis, 1998). This electronic lexicon is the premise on which the grammar checker is developed. The morphological lexicon plays a key role in supporting the Greek grammar checker, as the first level in which the language is examined is the morphology level and the structural level is not only based but also depends on the morphology of the words.

A major problem in processing the natural language was the lexical ambiguity (Gakis *et al.*, 2013), a product of the sophisticated morphology of the Greek language. Given that the major problem of Modern Greek is the lexical ambiguity, we designed the Greek tagger based on linguistic criteria in cases where the lexical ambiguity impedes the imprint of the errors in Greek language.

2 Templates of Greek grammar checker

The standardization of grammatical errors in templates was the most important part in the design. Errors gathered through an authentic body of specialized corpus. The grammar checker failed to wholly analyze the sentence but analyzed only those categories that have been described in the templates. The errors described by the grammar checker did not include all the wrong types because the grammatical analyzer is designed to focus on those cases that are the most typical and frequent.

The main areas of the grammatical errors in which the grammar checker interferes are: (1) punctuation problems, (2) final -n, (3) stylistic issues, (4) standardization issues (stereotyped phrases, words of literary origin), (5) inclination issues (incorrect declension of names or verbs either through ignorance or because of confusion), (6) vocabulary issues (cases of conceptual confusion, Greek translation of foreign words, redundancy, and use of incorrect word or phrase), (7) orthographic confusion issues (homonymous words), (8) agreement issues (cases of elements of nominal or verbal phrase disagreement), (9) syntax issues (verbs), and (10) cases of errors that require more specialized management of the spelling correction.

3 Particularities of Modern Greek language

3.1 Highly inflectional and 'free word order' language

Natural language processing systems incorporate notoriously complex algorithmic processes which become even more complicated as far as Modern Greek language is concerned. The declinable parts of speech produce a huge set of morphological word forms, as Modern Greek is a highly inflectional language (Gakis *et al.*, 2012). The vocabulary of Modern Greek also includes words that are lent from other languages.

Moreover, Modern Greek is a 'free word order' language and allows the speaker to form phrases in various ways. These variations however are big challenges for computational linguistics (Orphanos, 2000).

3.2 Lexical ambiguity

Lyons describes the ambiguity that is noticed generally in language with the term lexical ambiguity (Lyons, 1977) and determines two different categories of lexical ambiguity: (1) the homonymy and (2) the multi-meanings ambiguity. Beyond the different meanings that are given in the term lexical ambiguity, for the computer, this has direct relation with the way of re-construction and set-up of the lexicological entries (Boguraev and Pustejovsky, 1990).

Lexical ambiguity is also an important phenomenon in Modern Greek. This happens when a word type has more than one corresponding lexical entry (lemmas) or when the word is used with a different meaning in a figurative sense. As a result, we have to deal with a great number of words with an ambiguous meaning, and unless their meaning is resolved by the context, this ambiguity may carry over to phrases or even whole sentences (Gakis *et al.*, 2013). During natural language processing by computational systems, syntactic structures are represented by phrase structure rules (Chomsky, 1965; McCord, 1987; Kermanidou, 2005). However, there are many cases in which ambiguity is introduced as a part of speech—which is a major feature (Pollard and Sag, 1987)—such as an individual morphological attribute that a lexical type may have. When the part of speech is ambiguous, the parser is forced to examine much more syntactic rules and, eventually, produce all the phrasal structures that these rules dictate, hoping that one analysis will finally prevail.

4 The 'Kanon' formalism

In order to construct the Greek grammar checker, we used the 'Mnemosyne' environment, a complete complicated natural language processing system used for information retrieval and information extraction in free text. This software has been developed in Java using a parallel and/or distributed architecture. The 'Kanon' formalism is used to describe complex syntactic structures, obeys the Unification Grammars and belongs to the level of context-sensitive grammars (Chomsky, 1965).

The Mnemosyne environment constitutes of a complete NLP system that incorporates advanced linguistic resources and computational tools aiming at the automatic extraction of structured information from unstructured electronic documents. It is mainly used for automatic processing of free-text documents. It ensures processing of big volumes of information, high precision in the recognition of named entities and events, and possibility of addition of new sources of information with low cost.

The advantage of Mnemosyne is that it incorporates linguistic information data. It has already been used in environments with large quantity items with very good results on the size of the input data, the processing speed, and the output precision. The language of the text was Greek but 'Mnemosyne' can handle all European languages.

In https://ws.neurolingo.gr/WebCleansing/GGC.html, we posted all templates of grammar checker in a friendly environment. The user types or copies the text at the lexical editor and sees the analysis of his text.

The text analysis includes (1) morphological label of words[2] and (2) note of text errors. Grammar rules are grouped into the following categories: (1) error, (2) info, and (3) warning. The 'error' category describes errors that deviate from the linguistic norm; the 'info' category includes rules concerning the style information; and the 'warning' category includes rules concerning user's semantic information about words with conceptual confusion. The user can select some or all of the above rules (colored differently) and has, respectively, the corresponding message and, in the majority of cases, the 'correct' type.

The following figure (Fig. 1) presents the architecture and the structure of Mnemosyne.

The basic course analysis of a text is as follows:

(1) the collection of documents, which is the setting of all input sources;
(2) the processing flow, which consists of the type of analysis devices which analyze the sources (Savranidis, 1998);
(3) the compendium analyzations, which are specialized analyzers to control the information flow, as well as to alert users to the process, the errors, and the warnings; and
(4) the XML Dumpers, specialized analyzers that ensure the transfer of extracted information to specific destinations and formats (e.g. XML, Database tables).

The levels of a text processing follow a particular sequence and the text follows the levels of analysis, when the previous process finishes. In each level, parallel rules are applied.

A text is divided into paragraphs, based on the paragraph analyzer. The end of each paragraph is usually one or several blank lines. Consequently, each paragraph is divided into sentences. For Mnemosyne, sentence is the

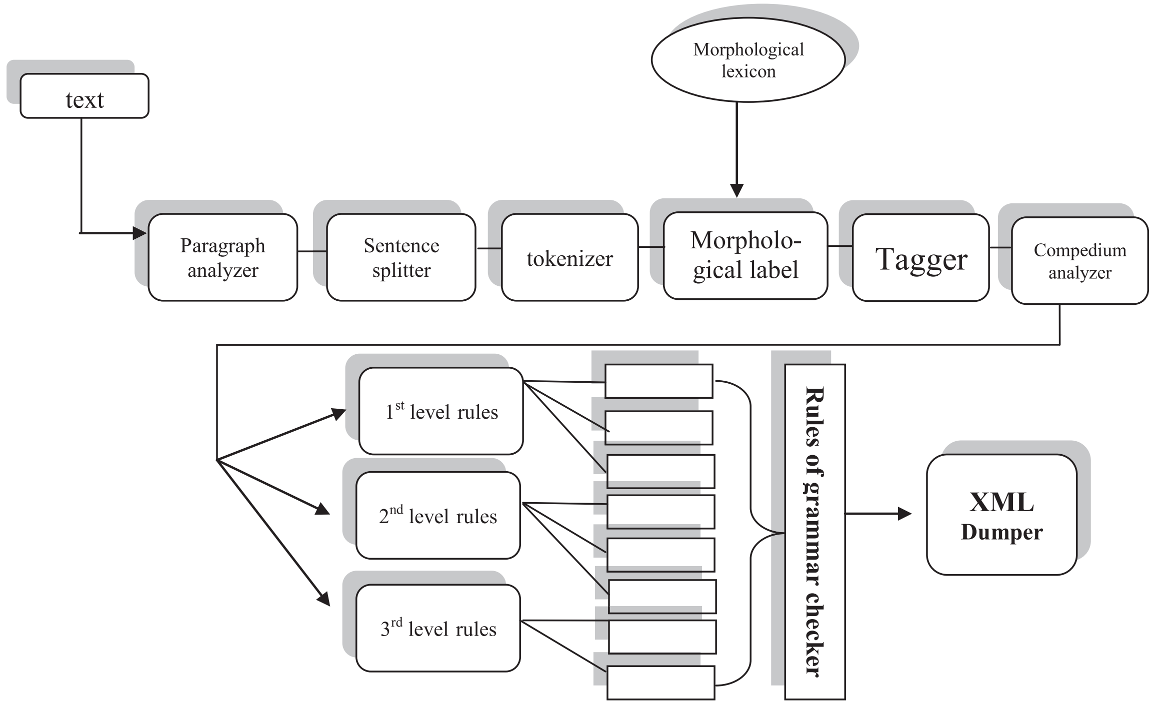


Fig. 1 Architecture and the structure of Mnemosyne

minimum semantic unit.[3] Their limitation is important to delineate, in order to retrieve the correct syntactic structures. Extra care is required when the part of speech is an abbreviation or acronym and the dot should not be seen as the end of the period.

The output in XML format is following:


```
<analysis name="sentence">
    <span length="23" offset="0">
      <contents>ο κ. Νίκος ήρθε σήμερα.</contents>
      <annotations>
        <tagname="FPARSEN" class="class java.lang.String">true</tag>
        <tagname="SSEQNO" class="class java.lang.String">0</tag>
      </annotations>
    </span>
```


In the next level of analysis, each component of the sentence is characterized (words, symbols, brackets, apostrophes, the quotes, etc.). Initially, spelling of each component of the sentence is recorded (Greek letters, numbers, uppercase first letter, etc.). In the next level, Mnemosyne handles with the tokens, followed by punctuation or other spelling points. For example, a token ending in comma or exclamation mark or question mark or semicolon separates into two words. Mnemosyne acts in a similar way in cases where the word begins or ends with quotes: separation into two or three words (Silberztein, 2003; Paumier, 2003). Mnemosyne recognizes as one token the acronyms or the pronoun [ό,τι].

5 The tagger of Grammar Checker

The Electronic Lexicon—in the following level—gives in each word the exact morphological and stylistic attributes necessary for parsing. For example, the word [*βωμού* = altar] has the following morphological attributes: N (MASC + GEN + SING), THEATER], which means that it has the morphological attribute noun (N), the morphological attributes: male (MASC) for gender, generai (GEN) for case, and singular (SING) for number. Additionally, it is characterized by a thematic area attribute (THEATER).

A tagger attaches the correct morphological attributes in words with lexical ambiguity. The clarification of lexical ambiguity is one of the most important issues in processing a text. The context will determine if the word [το] is an article or a pronoun, knowledge absolutely necessary at a later level of analysis in the grammatical errors. The tagger is oriented especially to the removal of lexical ambiguity in Greek. Van Eijck (Van Eijck and Jaspars, 1996) defines the lexical ambiguity as information shortage for the word meaning. It is not based on decision trees (Orphanos and Christodoulakis, 1999; Orphanos and Tsalidis, 1999), statistics methods (Tambouratzis and Carayannis, 2001), approaches based on machine learning (Papageorgiou *et al.*, 2000), the morphological model of two levels by Koskenniemi (Sgarbas et al., 2000a,b), but in context. The implementation of the tagger is geared to the needs of the rules of the levels of the grammar checker.

Grammar Checker supports the complete removal of lexical ambiguity exclusively using linguistic information. It consists of 70 rules. Most of the rules have to do with the removal of lexical ambiguity between article and pronoun. The categories of tagger are: (1) noun and adverb, (2) noun and preposition, (3) noun and pronoun, (4) noun and article, (5) noun and conjunction, (6) noun and verb, (7) noun and participle, (8) noun and interjection, (9) verb and noun, (10) preposition and pronoun, (11) adverb and conjunction, and (12) adjective and noun. Tagger defines words that do not exist in the morphological lexicon (unstressed words, words with misspelling) as well as the removal of lexical ambiguity in the gender and the case of the ambiguous word. For the removal of lexical ambiguity, Mnemosyne examines both previous words—up to 4 tokens—and/or the following word—up to 4 tokens. A rule has the form shown in Fig. 2.

6 Grammar checker rules

The parser is based on the templates and consists of approximately 2,600 "Kanon" rules. The Greek grammar checker in no case attempts to resolve extreme forms of the language problem that causes doubts.

At the beginning, the parser attributes each token. The result of this morphological analysis is: (1) the location of grammatical and semantic errors that have been described in rules flow, and (2) the location of the production tree that marks the part of the problematic parsing.

This formalism uses in a good way the morphological and stylistic characteristics of word types that have been described in electronic morphological lexicon. This formalism also recognizes incorrect polylectic terms.

It is worth mentioning that in all cases, any mistake is considered separately and a set of actions are defined in order to face it.

Every rule contains one or more head predicates and one or more body predicates. Head predicates are defined in terms of the body predicates and this means that if a sequence of symbols (text spans) matches the body predicates then we can reduce

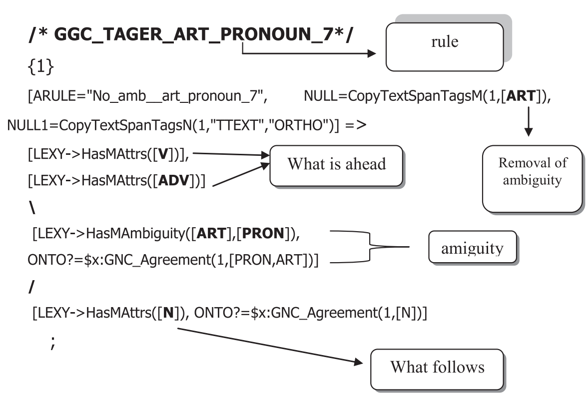


Fig. 2 The tagger formalism

these predicates to the one of the body. Rules are independent of each other. Their order does not matter the way they are evaluated. The system can use different heuristics about which rule to choose for reduction in case that multiple rules match an input sequence of symbols. The current applied technique chooses the longest (in terms of size of predicates in the body of a rule) rule. The symbols '\' and '/' specify the left and right context of a reduction. We can have a list of predicates at the left of the '\' symbol denoting the left context of the reduction. The meaning of the left context is that we expect to match all the predicates presented in the left context but we will not use them in the reduction. The same holds for the right context. Only the predicates presented between the '\' and '/' symbols will be reduced. Parentheses can also be used to group sequence of predicates. A body predicate or group can be right followed by a repeating operator of the '*', '+', {m,n}. The meaning of '*' is zero or more instances of the predicate or group existing in the left of the operator must be matched. The '+' operator is interpreted as one or more instances while the expression {m,n} means that we expect to match at least m and an most n instances.

Specifically each rule has five elements:

(1) The head of a rule, left side of the symbol "=>", for example [VTEXT="*μήνας*"] *ή* [VTEXT="*ρήμα*"]. This determines the replacement of the identified expression of one or more virtual items or rules. Thus, the learned expression: *αυστηρώς ακατάλληλον* [strictly not appropriate] is replaced by the virtual text VTEXT="__ancient_phrase__". The rule description has the following form:

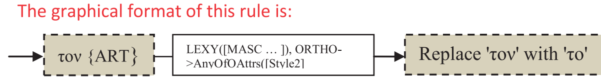
```
section
   /* GGC_pattern_ancient_1*/
   {1}
   [ARULE="pattern_ancient_1", VTEXT="__ancient_phrase__"] =>
           \
              (
                 [TTEXT->Match ("αυστηρώς")],
                 [TTEXT->Match ("ακατάλληλον")]
              )
           /
   ;
   end
```

The graphical format of this rule is:

Respectively, in the following rule, the corresponding message exists for the final -n:

```
section
   /* GGC_TELIKO_N_remove_n_3*/
   {3}
   [ARULE="GGC_TELIKO_N_remove_n_3", TTEXT=$x_1_3,
   EVTEXT=TagEvent("gevent.wrong","GEVENT","FINAL_N","%f","ERMSG","The
   word 'τον' hasn't the final-ν. Replace 'τον' with 'το'.")] =>
   \
      [TTEXT->$x_1_3:Match("τον"), LEXY->CanMatch("ο", [ART, ACC, MASC,
                                                        SING])]
   /
      [LEXY->HasMAttrs([MASC,SING,ACC]), ORTHO->AnyOfOAttrs([Style2])]
   ;
   end
```

The graphical format of this rule is:



(2) The corresponding word or phrase or lemma that is being considered (in this case the wrong type) exists between the symbols '\' and '/'. The first example is the phrase [*αυστηρώς ακατάλληλον* = most offensive] and the second is the type *τον* of the article.
(3) The left part of the expression before the symbol '\' (in the above examples is empty) is a set of words, phrases, and other tokens that is useful for expression determination but it is not replaced by the head of the rule.
(4) Similarly the right of the expression following the '\'. In the first example, there is no right-hand part, while the second is: [LEXY ->HasMAttrs ([MASC, SING, ACC]), ORTHO ->AnyOfOAttrs ([Style2])] and denotes any word having morphological characteristics (Male, Singular Accusative) and belonging to a group of words specified by the first letter [Style2]. The right side also remains unchanged in case of grammar rules application.
(5) The type of rules that lies within the symbol 'and', before the left panel. This part is not necessary, but there is in each rule and is used to categorize errors.

Each word is defined by a sequence of values surrounded by the symbols '[' and ']'. Thus, the symbol *ORTHO->AnyOfOAttrs ([WthSmbs])* defines a condition that must be accepted by the rule and notes that the token spellingWithSymbols.

The condition *ORTHO->AnyOfOAttrs([NrWrd])* means that the word must be normal Word (word with letters of the alphabet).

The condition *[LEXY->HasMAttrs([ADJ,... ACC])]* means any entry that has certain morphological attributes (in this case: adjective ...accusative). Respectively may include the condition *[LEXY->HasNoMAttrs([ADJ,.... ACC])]* means any entry that has not the certain morphological attributes.

The condition *[TTEXT->Match(«γκαρσόν»)]* defines words that have the specific word sequence (capital letters (*ΓΚΑΡΣΟΝ*) or/and small letter (*γκαρσόν/Γκαρσόν*), stressed or unstressed: *γκαρσόν*), while the more 'strict' form is defined by *[TTEXT == «επιστήμων»]* which includes only the specific type: *επιστήμων*. These rules have the opportunity to define the prefix *πρόθημα ([TTEXT->Prefix («εξεφώνησ»)])* which includes all the generated words with the prefix «*εξεφώνησ*». Under circumstances, the rules define the suffix of any word *[TTEXT ->Suffix («ουνα»)]* which includes all the generated words with the suffix -*ουνα*. Both cases support additional information by the definition of additional morphosyntactic attributes, to avoid identification of the type with another word with the same suffix and different morphological attributes. Thus, the rule *[TTEXT ->Suffix(«ουνα»), LEXY->HasMAttrs([V, A_P, SING*])] defines words with suffix–*ουνα* and morphosyntactic attributes [Verb, 1st person, singular].

The condition [*LEXY->HasLemma3* («*γράφω*»)] defines the specific lemma (all tenses, moods, persons, voices), while for more limited search is used the condition *[LEXY->CanMatch(«ψηλός», [FEM])]* in which is defined only the female of the adjective: *ψηλός*. These conditions can be additionally determined by agreement conditions `[ONTO?=$x:GNC_Agreement(1,[ADJ])]` in many levels:

(1) agreement in gender, number, and case `[ONTO?=$x: GNC_Agreement(1)]`;
(2) agreement in number and case `[ONTO?= $x: GNC_Agreement(2)]`;
(3) agreement only in number `[ONTO?=$x:GNC_Agreement(3)]`; or
(4) agreement only in case `[ONTO?=$x:GNC_Agreement(4)]`.

Mnemosyne has four levels of analysis. This function allows the phrases replacement by VTEXT. The necessity of levels becomes visible in the following analysis problem. Specifically, the particle [*μην*] is not converted into [*μη*] in a specific context (style1: the first letter is *κ*, *π*, *τ*, etc.), while in another context remains (style2: the first letter is *β*, *γ*, *δ*, etc.). This rule does not apply to learned participles still in use in spoken language. In the first level, therefore, when the particle [*μη*] is found in context consisting of learned participle is replaced by a virtual text, the [VTTEXT = "__archaiametochi__"]. Applying this approach, independent of the first letter of the learned participle, the rule concerning the final -n is not executed. The statement of the rule has the following form:

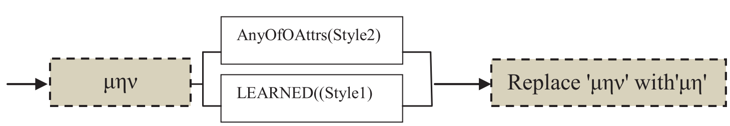
```
Section
/* GGC_TELIKO_N_remove_n_12 */
{1}
 [ARULE="GGC_TELIKO_N_remove_n_12", VTEXT="_learned_particle_"] =>
        \
                [LEXY->CanMatch("μη",[PARTICLE])]
        /
        (
                [LEXY->HasMAttrs([LEARNED])]  |
                [LEXY->HasMAttrs([PART,LEARNED])]
         )
 ;
end
```

The graphical format of this rule is:

At the second level, however, the rule applies to all cases that are analyzed. This rule determines as error more specific events, e.g. the particle [$\mu\eta\nu$] in front of learned participles and therefore is replaced by the particle [$\mu\eta$]. More specifically:

```
Section
/* GGC_TELIKO_N_remove_n_13*/
{2}
[ARULE="GGC_TELIKO_N_remove_n_13",
EVTEXT=TagEvent("gevent.wrong","GEVENT","FINAL_N","%f","ERMSG",
"The particle 'μην' hasn't the final -ν. Replace the particle 'μην' with'μη'.")] =>
      \
               [LEXY->CanMatch("μην",[PARTICLE])]
      /
       (
              [ORTHO->AnyOfOAttrs([Style2])] |
              [LEXY->HasMAttrs([LEARNED]), ORTHO->AnyOfOAttrs([Style1])]
      )
      ;
end.
```

The graphical format of this rule is:



At the end of this process, there is the final analysis, the dumper. The dumpers transfer the results of processing in detail and extract and record all the features (sentence analyzer, tagger, rules flow, rules) in the appropriate position. Export and analysis files are XML and Apache Lucene (http://lucene.apache.org).

All dumpers have a certain common formalism format: {$info} {analysis (options)}, where {} means 0 or more than one items enclosed in {}.

6.1 Stylistic rules (learned types)

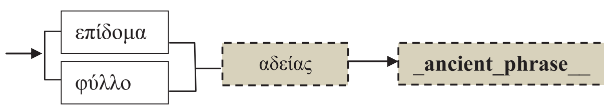
Learned type is either a type with learned morphological suffix or a lemma with similar attribute. The grammar checker handles with learned types in several levels. In the first level, the grammar checker handles with learned phrases (~359) that are still in use in written or spoken speech.

At this level, the grammar checker recognizes these phrases and replaces them by a [VTEXT =] = "__ ancient_phrase__". This is because these phrases consist of words that have the morphological characterization: learned types in the specific context are acceptable.

In this level, grammar is a context-sensitive grammar and handles with such phrases as these that are detailed below:

```
Section
/* GGC_pattern_ancient_1*/
{1}
[ARULE="pattern_ancient_1", VTEXT="__ancient_phrase__"] =>
            (
                [LEXY->HasLemma3("επίδομα")]          |
                [LEXY->HasLemma3("φύλλο")]
            ),
       \
                [TTEXT=="αδείας"]
       /
       ;
       end
```

The graphical format of this rule is:



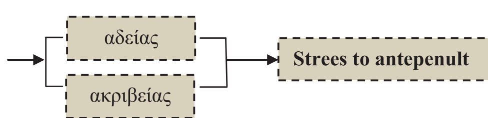
This means that the type [*αδείας* = license], which is a learned type, in the specific context (after lemma [*επίδομα* = benefit] ή [*φύλλο* = panel] will be replaced by VTEXT = "__ ancient_phrase__"].

In another context, which is described in the second level, the user is informed that the stylistic attribute of these words refers to formal speech and the opportunity to choose a word or type of Modern Greek lies in his distinguishing fluency.

Respectively grammar is a context-free grammar and handles with such learned types or lemmas as these detailed below:

```
Section
     /* GGC_pattern_2_ancient_2_2 */
       {2}
     [ARULE="GGC_pattern_ancient_2_2",
     EVTEXT=TagEvent("gevent.info","GEVENT","ANCIENT_TYPE","%f","ERMSG","Strees
     to antepenult")] =>
            \
                [TTEXT == "αδείας"]              |
                [TTEXT == "ακριβείας"]
            /
            ;
end
```

The graphical format of this rule is:



In another level, grammar checker uses a set of 113 rules, handles with potential errors in learned phrases (spelling errors, ungrammatical case, incorrect gender, voice, number or stress, redundancy, incorrect choice of lemma, etc.).

This template encloses lemmas that have this attribute from the morphological electronic lexicon: learned and types with learned consonant. The following figure (Fig. 3) presents the structure of this template.

6.2 Stylistic rules (oral types)

The rules of this template inform the user of possible inconsistency in style which he wishes to have in his text by the use of words that are characterized by stylistic attribute: ORAL. This template, by the use of levels, includes (1) lemmas with the attribute: ORAL, (2) noun this attribute, (3) types with prefix that report to oral speech, and (4) specific verbs which in specific context are characterized as oral (e.g. the verb [*είμαι* = I am] followed by the noun [*λάθος* = wrong] is informal phrase).

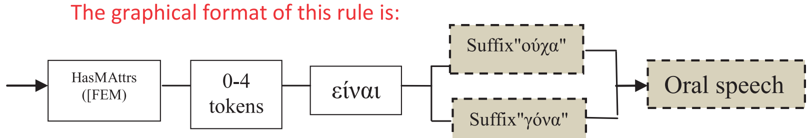
This template handles also with the oral types of adjectives and participles of passive voice (e.g. the oral suffix –*ούχα*/-*γόνα*) or the participles without reduplication.

We especially mention this category as adjectives and participles have to be syntactically analyzed. On the base that suffix [-*ούχα*] or [-*γόνα*] is female (oral attribute) or neutral (not oral type) parser has to check the gender. In these rules, analysis goes one level further and checks the agreement of the processed type. Here the tagger's support is necessary to define the gender of the type.

Below the way of dealing with oral adjectives or participles is presented:

```
Section
/* GGC_oral_adj_1_1*/
{1}
[ARULE="GGC_oral_adj_1_1",
EVTEXT=TagEvent("gevent.info","GEVENT","ORAL_ADJECTIVE","%f","ERMSG","The
suffix '-ούχα/ -γόνα' is used in oral speech. Replace with the suffix '-ούχος / -γόνος'.")]
=>
        [LEXY->HasMAttrs([FEM]), ONTO?=$x:GNC_Agreement(2,[FEM])],
        []{0,4},
        [TTEXT->Match("είναι")],
         []{0,3}
        \
                [LEXY->SuffixCanMatch("ούχος",[ADJ,FEM]),TTEXT->Suffix("ούχα"),
                  ONTO?=$x:GNC_Agreement(2,[ADJ])]                          |
                [LEXY->SuffixCanMatch("γόνος",[ADJ,FEM]),TTEXT->Suffix("γόνα"),
                  ONTO?=$x:GNC_Agreement(2,[ADJ])]
                /
                ;
        end
```

The graphical format of this rule is:



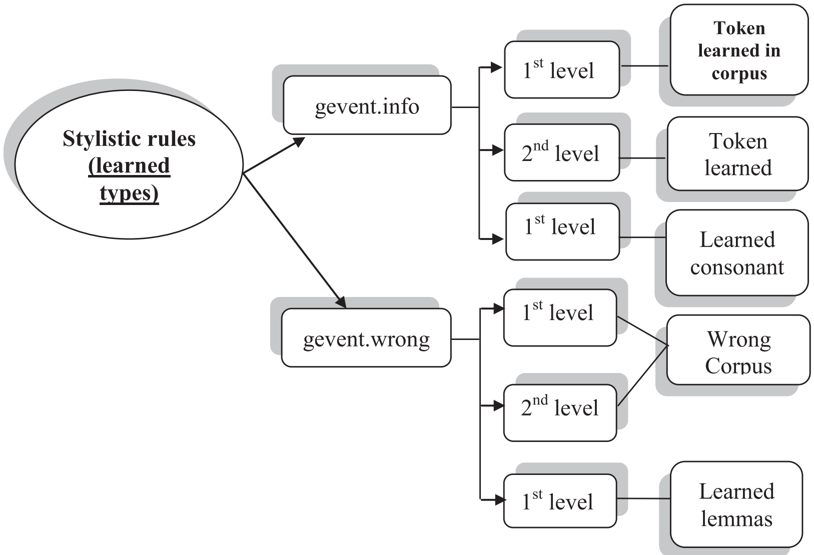
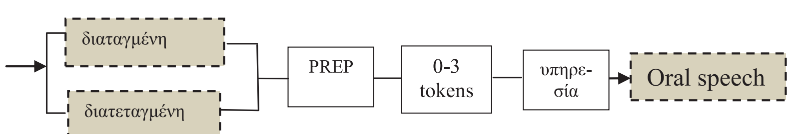


Fig. 3 Structure of template (learned types)

```
            Section
            /* GGC_oral_part_1_1*/
{1}
[ARULE="GGC_oral_part_1_1",
EVTEXT=TagEvent("gevent.info","GEVENT","ORAL_PARTICIPLE",       "%f","ERMSG","
These participle form is used in oral speech. Replace with the participle
'διατεταγμένη/ης'.")] =>
 \
            [TTEXT->Match("διαταγμένη"), LEXY->CanMatch("διατάζω",
            [PART, FEM, PASS]), ONTO?=$x:GNC_Agreement(4,[PART])] |
            [TTEXT->Match("διαταγμένης"), LEXY->CanMatch ("διατάζω",
                        [PART,FEM,PASS]), ONTO?=$x:GNC_Agreement(4,[PART])]
/
             (
                        [LEXY->HasMAttrs([PREP])],
                        []{0,3}
            )?,
 [LEXY->CanMatch("υπηρεσία",[N, SING]), ONTO?=$x:GNC_Agreement(4,[N])]
;
            end
```

The graphical format of this rule is:



The template includes rules for: (1) wrong abbreviations, (2) user's information for non-preferred style with abbreviations in formal texts, (3) oral consonants, (4) oral adverbs, and (5) phrases encountered mainly in oral speech.

The structure of this template (oral types) is given in Fig. 4.

6.3 Final -n

The most common error in written speech is described in this template and deals with almost all cases referring to the final -n.

Template includes rules that work at four levels, because of the number of cases and their difficulties. There are 20 rules describing the mistaken addition of the final -n and 20 where the absence of final -n is error. For these rules, the tagger support is crucial, while the art [*τον*] has different handling from the pronoun [*τον*]. Furthermore, the removal of gender and case ambiguity is criterion to add or remove the final -n.

The rules are defined in four levels in order to describe more complex linguistic situations (possessive adjectives, learned participles, prepositional, etc.). An example that highlights how useful the levels are is described below. In phrase {*το ένα* (*καλό*)}, the appearance of the final -n is not required in the word [*το*], but in phrase {*το ένα καλό άνθρωπο*}, final -n is needed to the word [*το*].

Originally letters are grouped into two groups: group (style 1) which contains the letters of the alphabet or symphonic complexes {*μπ*, *ντ*, etc.}, that demand the presence of the final –n, and the group (style 2) which contains the letters of the alphabet that emit the final -n.

The lemmas analyzed in this template are the articles {*τον*, *την*, *στον*, *στην*}, personal pronouns {*τον*, *την*}, the indefinite article {*έναν*, *μια*}, pronouns {*όποιος*, *κάποιος*, *όσος*, *πόσος*, *κάμποσος*, *τέτοιος*, *άλλος*}, particle {*δεν*, *μη*, *σαν*}, adjectives {*αρκετός*, *λίγος*, *πολύς*}, and numeric {*εκατό*}.

For the handling with rules concerning this template, parser checks in many cases both the previous words and the following (until six tokens). The structure of this template is presented more analytically later (Fig. 5).

6.4 Stress rules and other orthographic symbols

The wrong use of punctuation, stress, and orthographic symbols is described in this template. This parser handles with verb forms followed by two pronoun types written incorrectly in one word, with wrong particles stress [*να* and *θα*] and the wrong stress of two pronoun types followed by a verb in imperative. Moreover, grammar checker handles with monosyllabic types that in specific context are stressed, e.g. the types {*που* = where and *πως* = how, as interrogative adverbs} and wrong stressed words followed by.

This template includes rules for wrong hyphen presence or absence in specified context.

Much more complex is the way of dealing with comma. Grammar checker at a beginning level handles with the wrong use of comma in given words (*άρα* = so, *λοιπόν* = well, *ενδεχομένως* = perhaps, the word *ότι* = that/*ότι* = that = which). In a following level, it checks the use of comma before

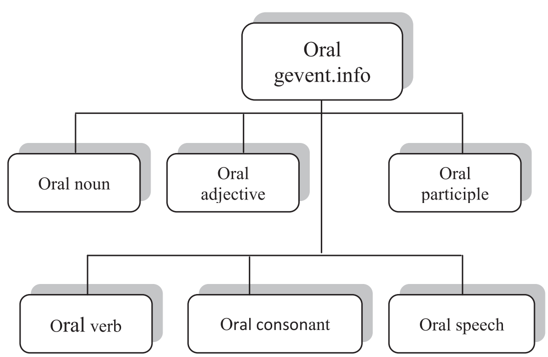


Fig. 4 Structure of template (oral types)

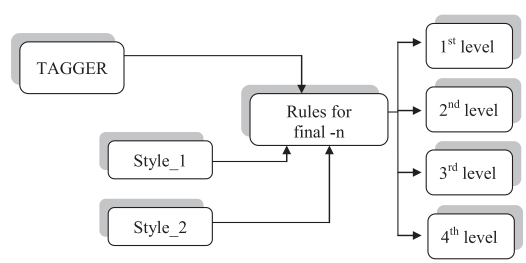


Fig. 5 Structure of template (final -n)

dependent sentences. When these sentences act as a modifying word, the use of comma is needed. Problems arise due to (1) the lexical ambiguity that arises when types act at the same time as conjunctions, adverbs or prepositions; and (2) depended sentences acting as clarification. The level function and the tagger support allow the handling of these problems.

The template describes errors related with typography (two gaps, the four dots instead of three, the question mark or comma presence the dot, etc.). The template also handles with cases of very frequent use but wrong written abbreviations (e.g. *κτλ.*). These rules need an analytic description in tokenizer, in order to properly integrate all punctuations within the word or as independent entity.

The structure of this template (Stress rules and other orthographic symbols) is presented in Fig. 6.

6.5 Conceptual confusion

In this template, grammar checker handles with words in which misunderstanding the correct meaning creates mistakes in written speech. Consequently, the word [*απλά*] with the sense of [no composite things] is used in sentences where the word [*απλώς* = only] must be used (Iordanidou 2013).

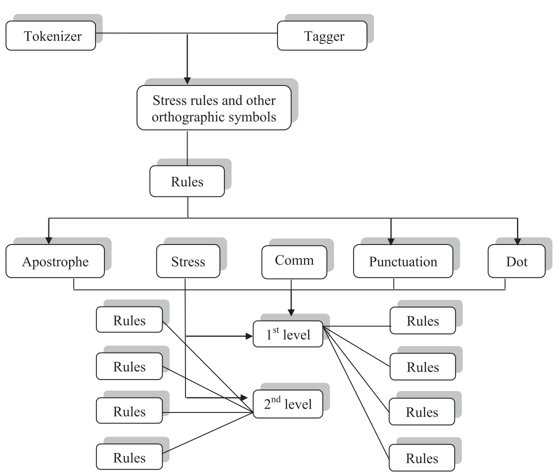


Fig. 6 Structure of template (stress rules and other orthographic symbols)

Homophones with different spelling words belong to this template (e.g. the [*λήμμα* = word] and [*λύμμα* = waste]). Depending on the part of speech, the rules that have been created refer to adjectives, adverbs, pronouns, verbs, and homophone word that belong to different parts of speech.

The rules of these templates inform ("gevent.-warning") the user for: (1) possible misplaced lemma choice, (2) the lemma to which is confused, as well as (3) the meaning of both lemmas.

Most of these lemmas are commonly used and, consequently, the user's information for possible mistaken choice would be unsuitable. For this reason, through a concordance of 870,000 newspaper articles and after searching these lemmas in search engines, the context of these lemmas was indexed.

So in the first level of analysis, the lemma that was indexed is replaced by a VTEXT and there is no information to the user about a possible mistaken choice. He is informed on the second level. For example the lemmas [*δεικτικός* = demonstrative] and [*δηκτικός* = critical] are homophones and are often used in a wrong way in the written speech.

We should stress out that the template belongs to a wider group of rules (rules.flow1) in which other templates belong (e.g. the rule with the final -n). A replacement by VTEXT causes problems to the function of other templates, in this case the rules of the final -n. That's why in computational processing and in the phrase *τη δεικτική αντωνυμία διαδέχεται ένα ουσιαστικό*], the tokens that are to be processed are [*τη*] [***VTEXT***] [*αντωνυμία*] [*διαδέχεται*] [*ένα*] [*ουσιαστικό*]. To avoid such failures, another group of rules (rules.flow2) which includes only the templates of the first level is created. Rules.flow1 includes all rules and templates both of the first and the VTEXT which do not affect the other.

The structure of this template is given in the following figure (Fig. 7).

6.6 Standardization issues

The rules of this template handle with lemmas of Modern Greek that are misspelled (one word instead of two or two instead of one). The wrong types are described by context-free grammar rules

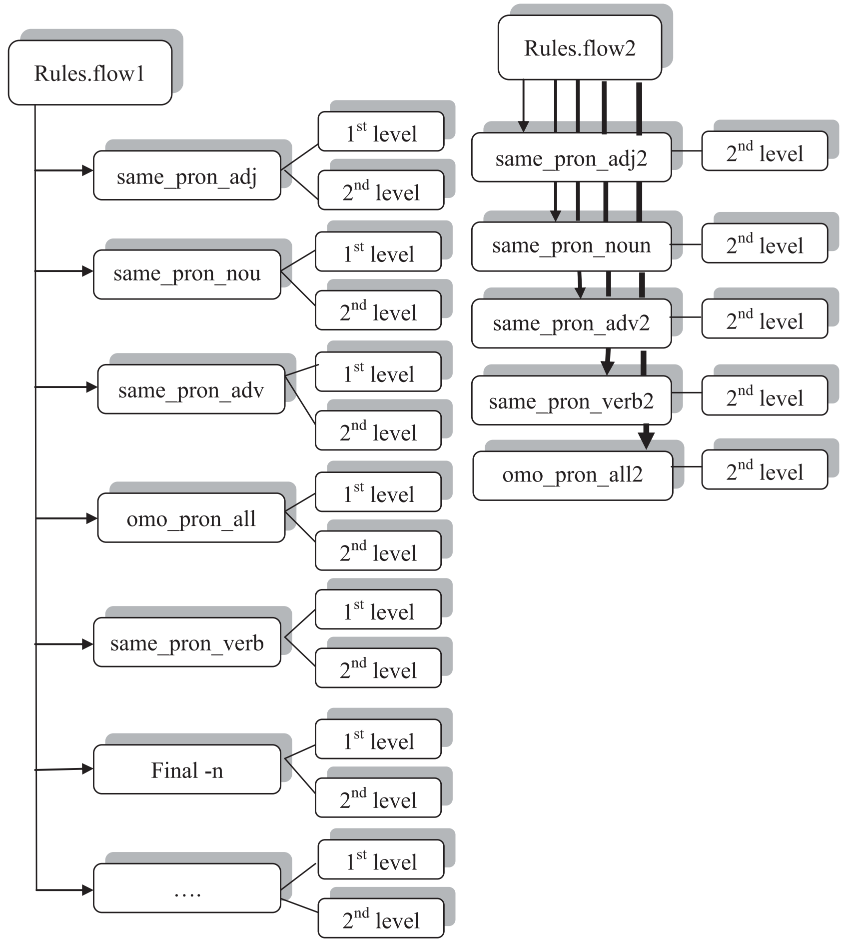


Fig. 7 Conceptual confusion

and the parser recommends to the user the correct types. This template includes foreign words written in one word depending on the stress in the Modern Greek, e.g. *αλερετούρ* (correct type) *αλέ ρετούρ* (wrong type). Foreign words written in wrong way, when corrected, are analyzed by other rules of another template (rules for foreign words) with the possibility that they may be replaced by a word of Modern Greek.

6.7 Vocabulary issues

Foreign words are included in this template (in the field of gevent.info rules). We should point out that this template does not handle with words which have been integrated into the Greek language in different periods without interruption until today. This category includes foreign words written in Greek characters and have been characterized by a morphological lexicon with stylistic attribute [FOREIGN WORD] (e.g. *Ίντερνετ* = internet).

Grammar checker recommends the corresponding Greek word or it informs that the used word is foreign, without recommending a corresponding Greek one (in cases that there is not a corresponding Greek word).

In the same template belong the Latin phrases still in use in oral and written speech that are handled in a similar way by the grammar checker.

In these phrases, the user is informed about their translation.

6.8 Redundancy

The errors included in this template are very frequent. First of all the parser handles with simple cases of redundancy (e.g. *ευθύς αμέσως* = right away, *από ανέκαθεν* = every time). The template includes (1) bad declaration of comparative or superlative (*πιο* = most + suffixes of comparative or superlative) and (2) more complex cases such as the verb's syntax, e.g. verbs with supplements that are unnecessary (*εμπεριέχω μέσα* = contained within, *επαναλαμβάνω πάλι* = I repeat again, etc.).

These cases become more problematic in processing, when the two tokens are not located in neighboring positions but one or more tokens are between them. So, for example, the phrase {*εμπεριέχω* [...] *μέσα*} might not be analyzed by the grammar checker if the token before *μέσα* is the article. Consequently, the possible forms of the object with the token [*μέσα*] are described in the first level: with article, with adjective, or participle. At the second level it nominates the token [*μέσα*] as an adverb. The punctuation at this level is important for the proper functioning of this rule.

The structure of this template is given in Fig. 8.

6.9 Nouns without plural

This template includes nouns with incorrect plural forming. Words with the incorrect plural are not defined (correctly) in a morphological lexicon. Consequently, these types of plural—used in oral speech but not in written speech—are described in this template.

This category includes the indeclinable loanwords from other languages that do not decline in the plural (Iordanidou, 2013).

However, there are words (e.g. *ζωή* = life) without plural apart from specific phrases (e.g. *έχει επτά ζωές* = has seven lives). Levels deal with these cases.

The template handles with words (e.g. *πολιτικές* = policies, *λογικές* = logics) whose plural is acceptable when they are adjectives, but wrong when they are nouns. In these cases, both the levels and the grouping the rules to larger groups deal with these cases (rules.flow1), so that VTEXT does not hinder the normal operation of the rules.

Nouns whose general plural is not commonly used have the same management.

6.10 Inclination issues (morphology)

This category includes: (1) the one-word or periphrastic type of verbs (e.g. the wrong formation of the present perfect and the past perfect of the verb [*άγω*] (simple or complex)), participle, or noun that has incorrect morphological formation; (2) the wrong voice selection in a specific context (e.g. deponent verbs used with suffixes of active voice); (3) types with wrong stress; and (4) the wrong formation of mood or tense (Iordanidou, 1999).

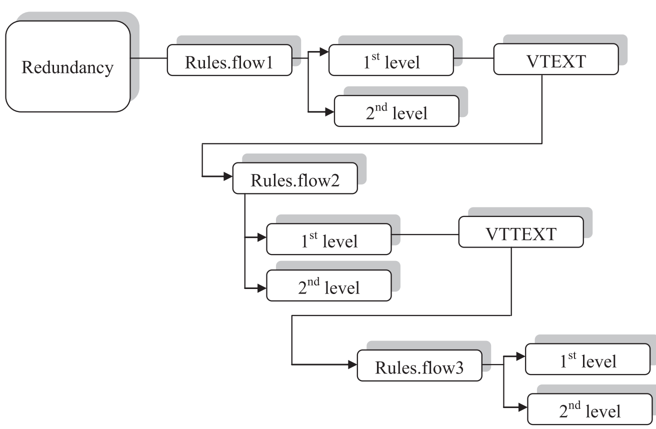


Fig. 8 Redundancy

6.11 Agreement (syntax)

The template handles with the agreement in a verb phrase or a noun phrase. Actually this template does not describe all possible forms (grammatically correct) of a noun or a verb phrase. This becomes prohibitive due to: (1) the variety of the possible forms of a phrase, (2) the huge lexical ambiguity which leads to many tree productions, and (3) the free-word order of the parts of a sentence. This template only describes the wrong formations of agreement of noun or verb phrase. Therefore, grammar checker checks the agreement between: (1) art with noun, adjective, or pronoun; (2) noun with noun; (3) particle or adjective with noun; (4) subject and verb; (5) parts of adverb phrase; (6) parts of preposition phrase; and (7) parts of verb phrase.

6.11.1 *Agreement in noun phrase*

The components of the noun phrase must agree to gender, number, and case.

The grammar checker allows the user to select the original type of the gender in some females ending in -os (e.g. [*ψήφος* = vote] [*μέθοδος* = method], [*δίφθογγος* = diphthong]). The electronic lexicon includes the morphological attribute: male but also the stylistic attribute (oral). The grammar checker suggests the user the type of female—as that is preferred in formal written speech. Moreover, the grammar checker acts in the same way when these nouns are identified by adjectives or predicate.

The template of agreement in a noun phrase has grouped these cases (e.g. Singular nominative of females with the suffix [-os]: [TTEXT-> Suffix (“os”), LEXY-> HasMAttrs ([N, FEM, SING, NOM])]) and checks the agreement with article, the predicate, or the adjective but only in the case and the number [ONTO? = $X: GNC_Agreement (2, [N])].

The levels’ function in this template is necessary, due to the same morphological formation of an article in nominative and general plural. So the first level describes the noun phrases in nominative and genitive plural which is replaced by a VTEXT.

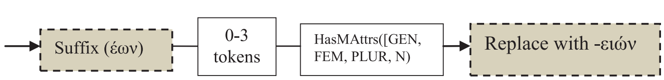
This template includes words (e.g. *μυς* = muscle, *δεσποινίς* = miss, males with suffix – *έας*) that are usually used incorrectly.

The wrong formation in adjectives with suffix [-υς, -*ιά*, -υ] is very common, especially in the genitive plural, while both the frequent use of genitive plural of males (the suffix –*έων*) in a female nouns’ context and the use of genitive plural of males with suffix *ειών* in a male nouns’ context (*των παχέων γυναικών* instead of the correct: *των παχειών γυναικών*) are common. The grammar checker in these cases checks the agreement between article and noun in gender and number.

Below the way of dealing with agreement between adjectives is presented:

```
Section
/* GGC_AGREEMENT_art_adj_1*/
{1}
[ARULE="GGC_AGREEMENT_art_adj_1",
EVTEXT=TagEvent("gevent.wrong","GEVENT",
"AGRREMENT_ART_ADJECTIVE","%f","ERMSG","Στη συγκεκριμένη ονοματική φράση
δεν υπάρχει συμφωνία. Αντικαταστήστε τον επίθετο με τον αντίστοιχο τύπο του
θηλυκού '-ειών'.")] =>
  \
        [LEXY->SuffixCanMatch("ύς",[ADJ]), TTEXT ->Suffix("έων")], "), LEXY-
        >HasMAttrs([ADJ,GEN, PLUR, MASC])],
  /
  ([0,3]
         [LEXY->HasMAttrs([GEN, FEM, PLUR, N])]
  ;
  end
```

The graphical format of this rule is:



Furthermore, the participles used as adjectives in female nouns have incorrect declaration (e.g. *ισχύοντες διατάξεις* instead of the correct: *ισχύουσες διατάξεις* = effective clauses | *πληγέντων περιοχών* instead of the correct: *πληγεισών περιοχών* = affected areas).

In this template, mistakes observed in wider noun phrases are described: (1) the adjective [*πολύς* = a lot], (2) the agreement between subject and (3) the wrong (semantically) selected adjective (e.g. *ραγδαία βελτίωση* instead of the right: *θεαματική βελτίωση*), (4) the use of adjectives instead of adverbs (e.g. *τόσοι πολλοί* instead of the correct: *τόσο πολλοί*).

The noun phrase analysis is presented in Fig. 9.

6.11.1 *Agreement in adverb and prepositional phrase*

The template in adverb and prepositional phrases handles with wrong preposition and adverb complements (e.g. the genitive plural [*εναντίων*] of the adjective [*ενάντιος* = opposed] which is homophone with the adverb [*εναντίον* = against]). Furthermore, grammar checker recognizes wrongly connected preposition phrases (Iordanidou, 2013) and recommends the preferable.

6.11.2 *Agreement in verb phrase*

This template describes the wrong syntax of verbs (Iordanidou, 2013). This description is not a single listing of the verb's syntactic structure. So, specific criteria (voice of verb, deponent verbs, possible object forms, etc.) are defined in order to describe exactly the "error". This template includes verbs that maintain—wrongly—the use of genitive in the object (grammar checker suggests the accusative). Furthermore, in this template, incorrect syntax of deponent verbs is described (Iordanidou, 1992).

The Greek grammar checker handles with the subject and verb agreement only in confusing circumstances (the suffix of the third singular of the middle voice of present tense [*εται*/-*αται*] and the suffix of the second plural active voice of present tense [-*ετε*/-*ατε*]). The template does not describe the variety of the possible structures of the two suffixes but for cases of wrongly suffix selection based on the most frequent cases observed in texts of concordance or student essays or other.

7 Evaluation

7.1 Comparing the Greek Grammar Checker

In evaluating grammar checkers, everyone needs to be aware of the kinds of errors they can possibly correct or not correct. Grammatical errors are described as wrong relation between words just as subject-verb disagreement or wrong sequence of words when for example plural noun is used instead of a single one.

Grammar checking phase starts after spell checking is finished (Ehsan & Faili, 2010). Grammar checking techniques can be categorized into three different types: syntax-based, statistical and rule-based. In syntax-based approach a text is completely parsed and if the parsing does not succeed the text is considered incorrect. The grammar checking is complete if the grammar itself is complete. The problem is that it cannot tell the user what exactly the problem is. Ill-formed sentences need extra rules for parsing. It requires a complete grammar which is obviously difficult to obtain for natural languages. In statistical approaches, a part-of-speech annotated corpus is used to build a list of POS tag sequences (Jensen et al., 1993).

There have been grammar checkers, developed for other languages. Grammatifix (Arppe, 2000) is a grammar checker for Swedish, which has the same approach with the Greek Grammar cheker. In this grammar checker, the error types of the language are collected first and those which result in high precision are chosen for implementation with this argument that the precision is important for grammar checkers (Bernth, 1997). Other Swedish grammar checkers are SCARRIE3 and Granska (Domeij, 2000). LanguageTool (Naber, 2003) is another grammar checker which has separate rules for 14 languages. EasyEnglish (Bernth, 2000) is another grammar checker developed for people whose main language is not English. For example, for German speakers it checks overuse of progressive form. GramCheck (Bustamante and Leœn, 1996)

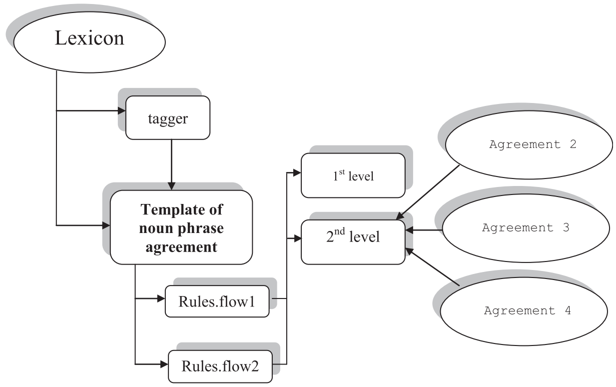
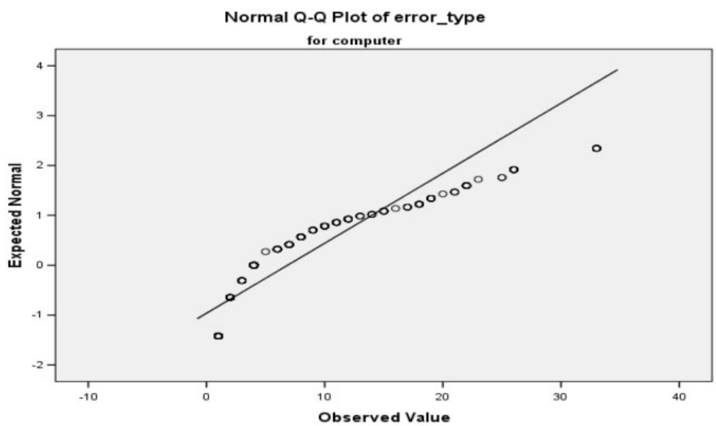
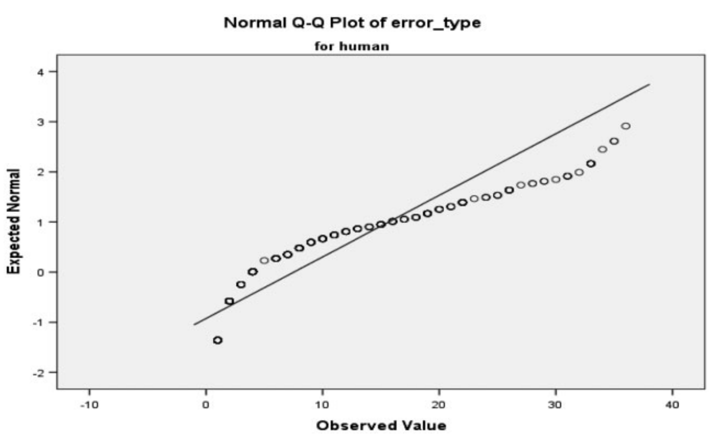


Fig. 9 Noun phrase agreement redundancy



Picture 1. Human correction



Picture 2. Grammar checker correction

is a grammar checker for Spanish and Greek, including rules only for agreement errors and certain head-argument relation issues. EasyEnglish, Critique and GramCheck and some other English grammar checkers are commercial and there is not enough documentation of the algorithms and rules which are used. According to other approaches, the developers of BonPatron, ‘‘a pedagogically-oriented grammar checker like BonPatron is at least as good as human correction for helping students improve their L2 writing skills’’ (Nadasdi & Sinclair, 2007). Other Grammar checkers (J. Park et al., 1997) have the same approach with the Greek Grammar Checker and do not attempt to handle all the possible grammar errors. The English Grammar checker handles with similar to the Greek Grammar Checker grammatical mistakes. The English Grammar Checker detects the following kinds of mistakes: Wrong capitalization (sentence initial, wrong lowercase/uppercase initial letter), missing fragments (subjects, objects, some prepositions, complements, articles, clauses, the, than, etc), some extra elements (e.g., the infinitive marker after auxiliary verbs), wrong agreement (number, case, etc), wrong verb form, and various mismatches (verb tense with adverbs, etc). It is beyond the scope of a grammar checker (Greek and other checkers) to identify mistakes along with missing fragments, run-on sentences, , wrong expressions, and wrong paragraph boundaries. Tense usage and pronominal reference are equally beyond their ability to correct. Grammar checkers identify certain specific types of grammatical mistakes in the proposed domain of application that are more regular than others. Existing grammar checking systems, such as those described in Thurmair (1990), (Bolioli et. al., 1992), (Genthial & Courtin, 1992; Bustamante & Leon, 1996), fall into this discipline, addressing the issue with a collection of heuristic rules that approximate a natural language grammar. CorrecText, from Houghton-Mifflin, is a significant advance in grammar checkers, because it uses a full parse of sentences in its analysis (Dobrin, 1990).

The Greek Grammar Checker addresses an implementation of grammar checking as an important application of computational linguistics, whose primary focus in syntax is on identifying what constitutes grammatical expressions. This grammar checker focuses on rule based approach and this is the first grammar checker for Greek. Of necessity, grammar checkers, on the whole, they operate on the premise that the underlying syntactic structure of a text is described as problematic. Because of the risk of misidentification of errors when the syntax of a text is faulty, they pass over in silence the entirety of structures they are unable to recognize. Greek Grammar checker has the same approach – for obtaining errors - with Latvian grammar checker (Deksne and Skadinš, 2011). As Greek and Latvian are highly inflected language with a high morphological ambiguity there are many long distance agreements between words and phrases in a sentence for which we need a deep syntactic analysis of phrases and sentence to find possible errors. The deference is that Greek Grammar Checker has only rules describing grammar errors when Latvian grammar has both rules for correct syntactic structures (G rules) and rules describing grammar errors (E rules).

7.2 Results

The construction of the grammar checker for the Modern Greek language is the first collection and coding effort of errors that occur in spoken and written language. The statistical processing and data analysis was performed using the statistical package for social sciences SPSS (v.21). For the statistical analysis non - parametric criteria have been applied, when the assumptions to implement parametric criteria were not met. When the observations were similar the statistical criterion $\chi2$ in pairs (McNemar test) was applied. We analyzed these data using the criterion McNemar, to detect possible differences in the correction of the two models (man and grammar checker). McNemar’s test is a statistical test used on paired nominal data. It is used to analyze studies, where the two treatments are given in matched subjects. This non-parametric (distribution-free) test assesses if a statistically significant change in proportions has occurred on a dichotomous trait at two time points on the same text. McNemar’s test determines whether paired proportions are different.

Table 1. Error processing by Greek grammar checker and human (statistics)

Error template	Grammar checker		Human corrector	
	Recognized errors	**Non -recognized errors**	**Recognized errors**	**Non - Recognized errors**
Comma removal	72	24	96	0
Comma addition	99	22	121	0
Apostrophe removal	14	2	14	2
Final -n removal	100	0	98	2
Dot removal	1	0	1	0
Final –n addition	16	2	16	2
n-phrase agreement	18	2	18	2
Oral type	32	2	34	0
Pre-phrase agreement	10	1	11	0
Foreign words	13	2	15	0
Learned types	7	4	11	0
Mistaken write	10	1	11	0
Stress addition	4	3	7	0
Apostrophe addition	4	2	4	2
Error in tense	10	2	10	2
Word alteration	1	6	7	0
Dot addition	4	2	4	2
Error in grammatical mood	7	0	6	1
Oral ending	12	2	12	2
Stress removal	1	5	6	0
Learned ending	4	2	4	2
Write in one word	11	3	11	3
Mistaken use of apostrophe	1	0	1	0
Verb absence	0	3	3	0
Incomplete phrase	2	3	3	2
Mistaken use of plural	9	1	10	0
Art absence	0	1	1	0
Mistaken phrase syntax	0	2	2	0
Quotation marks addition	0	2	2	0
Art addition	0	1	1	0
Change the word position	0	4	4	0
Word replacement	0	1	1	0
Adverb phrase agreement	8	2	8	2
change comma by dot	0	2	2	0
Preposition removal	0	1	1	0

The effective software evaluation was with the parallel correction of the same texts by the grammar checker and a human. The human correctors were four philologists, teachers in high school, with great experience in text correction. More than 100 texts were given for correction to the grammar and human checker.

The program was tested due to a subset of 90 student texts (compositions, articles. The compositions were written by students in their second year of high school with good grades (at least 70/100) in Greek language. Greek grammar checker approached in 90% the correction of human. In all, the texts contained 215 surface-level morpho-syntactic errors out of a total of ~3,500 words. These errors were randomly injected into the texts. The output results indicate that this approach is able to detect the grammatical errors that are described in the templates of Grammar Checker. Furthermore the results show that the Greek Grammar checker with rule-based approach is the most effective in detecting the defined errors.

In a very large percentage the grammar checker (picture 1) approximates the correction of a human (picture 2), because the electronic environment of Mnemosyne is closer to the way of thinking of human and the natural writing process (Daiute, 1985). Greek Grammar checker outdoes the speller, because it can recognize incorrectly (with the right spelling) homonyms, such as : [ότι: that] – [ό,τι: whatever], [δόσεις: dosages] – [δώσεις: you'll give] wrong forms of articles or pronouns such as: [η/οι, τον/των, της/τις], anagrams (the most common spelling errors), and the verb types of passive or active voice (διαβάζεται: its read – διαβάζετε: you read). It also includes stylistic information.

The error processing by Greek grammar checker and human showed the following (Table 1). The error processing showed that in a very large percentage the grammar checker approximates the correction of a human. Differentiation between grammar checker and human corrector is noticed in cases referring to the conceptual field which is not described in the grammar checker templates. The human corrector handled all the foreign words found in texts. Grammar checker doesn't describe these cases since the electronic lexicon manages only Greek words.

Finally we compared the templates of the Greek grammar checker with the templates of other grammar checker and we noticed that it includes their templates.

The size and completeness of the Greek electronic morphological lexicon and the tagger (the two basis of the Greek grammar checker) are innovative tools for Greek. These two applications can be the basis for the implementation for other computational tools (automatic summary, translation, etc.). The grammar checker is a standalone application. This product can be incorporated into any text editor and fill in the option of Ms-office.

Funding

Department of Primary Education of the University of Patras.

References

Arppe, A. (2000). *Developing a grammar checker for Swedish*. The 12th Nordic Conference of Computational Linguistics, 2000, pp. 13–27.

Bernth, A. (1997). EasyEnglish: a tool for improving document quality, *Proceedings of the fifth conference on Applied natural language processing*, pp. 159–165.

Bernth, A. (2000). EasyEnglish: Grammar checking for non-native speakers, *Proceedings of the Third International Workshop on Controlled Language Applications (CLAW00), Association for Computational Linguistics*, April, pp. 29–30.

Burston, J. (2008). ''Bon Patron: An Online Spelling, Grammar, and Expression Checker,'' *CALICO Journal*, vol. 25, no. 2, pp. 337–347, 2008.

Bustamante, F.R. & León, F.S. (1996a). GramCheck: A grammar and style checker, *Proceedings of the 16th conference on Computational linguistics-Volume 1*, pp. 175–181.

Bustamante, R., Flora & Sánchez-León Fernando. (1996b). ''GramCheck: A Grammar and Style Checker'', *Proceedings of the International Conference on Computational Linguistics* (COLING-96), pp. 175-181, Copenhagen, Denmark.

Boguraev, B. and Pustejovsky, J. (1990). *The Role of Knowledge Representation in Lexicon Design*, Proceedings of the 13th Conference on Computation Linguistics, Inland, pp. 36–41.

Boguraev, B. and Pustejovsky, J. (1990). *The Role of Knowledge Representation in Lexicon Design*, Proceedings of the 13th Conference on Computation Linguistics, Inland, pp. 36–41.

Boguraev, B. and Pustejovsky, J. (1990). *The Role of Knowledge Representation in Lexicon Design*, Proceedings of the 13th Conference on Computation Linguistics, Inland, pp. 36–41.

Chomsky, N. (1965). *Aspects of the Theory of Syntax*. Cambridge, MA: MIT, Press.

Daiute, C. (1985). *Writing and computers*. Reading, MA: Addison-Wesley.

Deksne, D., Skadiņš, R. (2011). CFG Based Grammar Checker for Latvian. In: *18th Nordic Conference in Computational Linguistics (NODALIDA 2011)*, pp. 275–278.

Dobrin, D. N. (1990). A new grammar checker. *Computers and the Humanities*, 24, pp. 67–80.

Domeij, R., Knutsson, O., Carlberger, J. & Kann, V. (2000). *Granska–an efficient hybrid system for Swedish grammar checking*, Nodalida, pp. 28–40.

Ehsan, N., & Faili, H. (2010). Towards grammar checker development for Persian language, in 6th IEEE International Conference on Natural Language Processing and Knowledge Engineering (IEEE NLP-KE'10), pp. 150–157.

Gakis, P. (2015). Design, Construction and Evaluation of the Greek Grammar Checker. Ph.D. thesis, University of Patras.

Gakis, P., Panagiotakopoulos, Ch., Sgarbas, K. and TsalidisCh., (2012). Design and implementation of an electronic lexicon for modern Greek. *Literary and Linguistic Computing*, 27(2): 155–70.

Gakis, P., Panagiotakopoulos, Ch., Sgarbas, K. and Tsalidis, Ch. (2013). Analysis of lexical ambiguity in Modern Greek using a computational lexicon. *Literary and Linguistic Computing*, 27(2): 20–38.

Genthial D. & Courtin J., (1992). ''Prom Detection/ Correction to Computer Aided Writing'', *Proceedings of the 15th International Conference on Computational Linguistics* (COLING- 92), pp. 1013-1018.

Iordanidou, A. (1992). *4500 Modern Greek Verbs in Greek* (2nd ed. 2003). Athens: Patakis.

Iordanidou, A. (1999). *The Guidebook of Modern Greek*. Athens: Patakis.

Iordanidou, A. (2004). *The Guidebook of Modern Greek*, 2nd ed. Athens: Patakis.

Iordanidou, A. (2013). *Is it Error or Not. Behold the question*. Athens: Metaixmio.

Jensen, K., Heidorn, G.E. & Richardson, S.D. (1993).*Natural language processing: the PLNLP approach*, Kluwer.

Kermanidou, K. (2005). Learning of Syntactic Dependencies and the Grammatical Development of the Greek Language, Ph.D. thesis, University of Patras.

Kriaras, E. (1995). *The Dictionary of Synchronous Civic Greek (LKN)*. Kriaras, Athens.

Lyons, J. (1977). *Semantics (Two volumes)*, Cambridge: Cambridge University Press.

Mpampiniotis, G. (1998). *The Dictionary for the Modern Greek*. Mpampiniotis, Athens.

McCord, M. (1987). *Natural Language Processing in Prolog, Knowledge Systems and Prolog*. Addison-Wesley, Benjamin, p. 291.

Naber, D. (2003). *A rule-based style and grammar checker*, Diplomarbeit Technische Fakultät, Universität Bielefeld, Germany.

Nadasdi, T., & Sinclair, S. (2007). Anything I can do, CPU can do better: A comparison of human and computer grammar correction for L2 writing using BonPatron.com. Available at http://bonpatron.com.

Orphanos, G. (1993). *Computational Morphosyntactic Analysis of Greek*. Ph.D. thesis, University of Patras.

Orphanos, G. and Christodoulakis, D. (1999). Part-of-speech Disambiguation and Unknown Word Guessing with Decision Trees, Proceedings of EACL, Bergen, Norway, pp. 134–41.

Orphanos, G., & Tsalidis, Ch. (1999). Combining handcrafted and corpus-acquired Lexical Knowledge into a Morphosyntactic Tagger, Proceedings of the 2ndCLUK Research Colloquium, Essex, UK.

Papageorgiou, G., Prokopidis, X.Giouli, B. and Piperidis, S. (2000). A Unified POS Tagging Architecture and its Application to Greek, Proceedings of the 2nd Language Resources and Evaluation Conference, Athens, pp. 1455–62.

Park, J.C., Palmer, M. & Washburn, G. (1997). ''An English grammar checker as a writing aid for students of English as a second language'', In *Proceedings of Conference on Applied Natural Language Process*, New Brunswick, NJ.

Paumier, S. (2003). *De la reconnaissance de formes linguistiques à l'analyse syntaxique*, Ph.D. Thesis, Université Marne-la-Vallée.

Pollard, C. and Sag, A. (1987). *Information-Based Syntax and Semantics*, Vol. 1. CSLI Publications, Stanford.

Savranidis, X. (1998). Teaching language with the support of computer. *Sentence Analyzer*. http://www.etpe.gr/custom/pdf/etpe264.pdf.

Sgarbas, K., Fakotakis, N. & Kokkinakis, G. (2000a). Two algorithms for incremental construction of directed acyclic word graphs. *International Journal on Artificial Intelligence Tools, World Scientific*, 4(3), 369–81.

Sgarbas, K., Fakotakis, N. and Kokkinakis, G. (2000b). A Straight forward Approach to Morphological Analysis and Synthesis, 2nd ed., Proceedings COMLEX 2000, Workshop on Computational Lexicography and Multimedia Dictionaries, Greece, pp. 31–34.

Silberztein, M. (2003). Finite-state description of the french determiner system. *French Language Studies*, 13, 221–46.

Tambouratzis, G. and Carayannis, G. (2001). Automatic Corpora – based Stemming in Greek. *Literary and Linguistic Computing*, 16(4), 445–66.

Triantafyllidis, M. (1991) [1941]). *Modern Greek Grammar*. 3rd rev. edn Thessaloniki: Manolis Triantafyllidis Foundation (in Greek).

Thurmair, G. (1990). Parsing for grammar and style checking. *Proceedings of the International Conference on Computational Linguistics* (COLING), pp. 365–370.

Van Eijck, J. and Jaspars, J. (1996). *Ambiguity and reasoning*. Technical Report CS-R9616. Dutch National Research Institute for Mathematics and Computer Science, Amsterdam.

Notes

1. http://www.neurolingo.gr/en/technology/lexica/morpholexicon.jsp
2. Adjectives, nouns, verbs, adverbs, etc.
3. 'Sentence' for Mnemosyne is what the traditional grammar called period.